Lower Bounds on Cross-Entropy Loss in the Presence of Test-time Adversaries

Arjun Nitin Bhagoji [*1] **Daniel Cullina** [*2] **Vikash Sehwag** [3] **Prateek Mittal** [3]

Abstract

Understanding the fundamental limits of robust supervised learning has emerged as a problem of immense interest, from both practical and theoretical standpoints. In particular, it is critical to determine classifier-agnostic bounds on the training loss to establish when learning is possible. In this paper, we determine optimal lower bounds on the cross-entropy loss in the presence of test-time adversaries, along with the corresponding optimal classification outputs. Our formulation of the bound as a solution to an optimization problem is general enough to encompass any loss function depending on soft classifier outputs. We also propose and provide a proof of correctness for a bespoke algorithm to compute this lower bound efficiently, allowing us to determine lower bounds for multiple practical datasets of interest. We use our lower bounds as a diagnostic tool to determine the effectiveness of current robust training methods and find a gap from optimality at larger budgets. Finally, we investigate the possibility of using of optimal classification outputs as soft labels to empirically improve robust training.

1. Introduction

The robustness of machine learning systems, particularly classifiers in the supervised setting, to adversarial perturbations (Szegedy et al., 2013; Goodfellow et al., 2015; Carlini & Wagner, 2017; Bhagoji et al., 2018; Madry et al., 2018) has become an important line of research owing to the critical role they play in society. While there is a tremendous amount of work on attacks and defenses (Papernot et al., 2016), a focus of recent research (Bhagoji et al., 2019; Dohmatob, 2019; Schmidt et al., 2018; Cullina et al., 2018; Mahloujifar et al., 2019; Diochnos et al., 2018) has been on establishing fundamental bounds on learning in the presence of test-time adversaries in various settings. One line of research (Bhagoji et al., 2019; Dohmatob, 2019; Pydi & Jog, 2020) into the limits of learning in the presence of test-time attackers has established classifier-agnostic lower bounds on adversarial robustness, i.e. the minimum $0-1$ loss that would be incurred by any classifier, when adversarial perturbations are added to the underlying data distribution. However, practical approaches to training classifiers such as neural networks usually use surrogate loss functions such as the cross-entropy loss that depend on the output confidence, and it is critical to establish bounds on these.

Thus, in this paper, we extend work on the information-theoretic limits of learning in the presence of test-time adversaries to *any* loss function that uses the output probabilities of a classifier, such as the cross-entropy loss. The key question this paper answers is:

What is the minimum possible cross-entropy loss that will be incurred by any classifier given a data distribution and adversary specification?

Answering this question enables us to quantitatively diagnose the effectiveness of practical defenses against adversarial examples (Madry et al., 2018; Zhang et al., 2019), and can inform the design of better learning algorithms. In particular, we can determine if current robust optimization techniques are able to recover these bounds as well as find regimes in which robust classification is not possible. A further goal is to investigate the findings from Bhagoji et al. (2019) which indicated a large gap between the theoretically determined lower bound for the $0-1$ loss and what was achievable in practice from robust training. We seek to determine if the gap arises solely from the use of a surrogate loss during training or if there is a more fundamental barrier to robustness.

To determine classifier-agnostic lower bounds on the cross-entropy loss, we focus on the interaction between data points when they are perturbed. We represent points from each class as the vertices of a graph, with edges existing between two vertices if the neighborhoods in which they can be perturbed overlap. We refer to this structure as a *conflict graph* (first implicitly defined by Bhagoji et al. (2019)). Data points connected by edges are then challenging to classify, even for the optimal classifier. The problem is then

[*]Equal contribution [1]Department of Computer Science, University of Chicago [2]Department of Electrical and Computer Engineering, Pennsylvania State University [3]Department of Electrical Engineering, Princeton University. Correspondence to: Arjun Nitin Bhagoji <abhagoji@uchicago.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

translated to one of finding the output probabilities of the optimal classifier over this graph. Minimizing the cross entropy loss over this graph determines these probabilities and provides a lower bound, which can be efficiently computed as the resulting optimization problem is convex. This quantity, known as the *graph entropy* (Körner, 1973), has independently appeared in information theory as the solution to a coding problem. We also determine an exact form for the lower bound on cross-entropy for a mixture of two Gaussians, along with the optimal classifer and adversarial strategy.

An efficient determination of these lower bounds is possible since the optimization problem is convex, but we find that existing solvers are prohibitively slow for the programs resulting from real-world distributions of interest. In light of this, we derive a custom algorithm that exploits the bipartite structure of the conflict graph and can determine bounds far faster than a generic convex solver for instantiations of interest. Our algorithm can find a solution in 10s of seconds for benchmark datasets such as MNIST (LeCun & Cortes, 1998), Fashion MNIST (Xiao et al., 2017) and CIFAR-10 (Krizhevsky & Hinton, 2009). We provide a proof of correctness and convergence for our algorithm.

We use our algorithm to find lower bounds on the cross-entropy loss for these benchmark datasets, as well as for synthetic Gaussian data. Comparing these bounds to the training loss obtained by state-of-the-art robust optimization techniques on commonly used deep neural networks, we find a gap in terms of convergence to the optimal loss. Interestingly, the gap is much larger for the $0-1$ loss than for the cross-entropy loss, indicating that the use of a surrogate loss does impact achievability but is not the sole reason for it. We examine the impact of model architectures and activation functions on this gap, finding that the former aids convergence while the latter has a negligible impact. Finally, for certain adversarial budgets, we find that the use of soft labels obtained from our framework during training can aid with both convergence and generalization. The code to reproduce all results in this paper is available at `https://github.com/arjunbhagoji/log-loss-lower-bounds`.

1.1. Summary of Contributions

General framework for lower bounds for all convex losses using output probabilities in the presence of test-time adversaries: Our problem formulation allows us to determine lower bounds on any loss function for a given dataset and adversary. In particular, we can compute lower bounds on the commonly used cross-entropy loss as well as the the optimal classification probabilities for all points.

Efficient determination of optimal log-loss: We propose a bespoke algorithm to compute these lower bounds and provide a proof of its correctness. For practical settings of interest, our algorithm provides a speedup of multiple orders of magnitude over a generic convex solver from CVXOPT (Andersen et al., 2013).

Analyzing the effectiveness of current robust training methods: Our framework enables us to determine regimes where robust classification is possible. In these regimes, we find that current robust training techniques are able to get close to, but not match, the lower bounds on cross-entropy loss. This gap is smaller than that for the $0-1$ loss observed in previous work, showing the impact of using surrogate losses. We also investigate the use of the optimal classification probabilities computed by our framework as soft-labels during training, and find that these aid in both convergence and generalization for certain adversarial budgets.

2. Lower Bounds on Cross-Entropy Loss

In this section, we derive lower bounds on the cross-entropy loss in the presence of a test-time attacker by demonstrating that it is the solution to a convex optimization problem. We show how this problem can be derived using a graphical interpretation of the classification problem. Our method applies to *all discrete two-class distributions as well as all adversaries perturbing points within a non-empty neighborhood*. We also extend our framework to the special case of a mixture of two Gaussians.

2.1. Problem formulation

We consider the following supervised classification problem. Data points x are drawn from a space $\mathcal{X}$, with labels $y \in \mathcal{Y} = \{-1, 1\}$. The joint probability distribution over this data is P [1]. The classification function (or classifier) $f : \mathcal{X} \to \mathcal{Y}$ maps data points to the space of labels. We also define a 'soft' classifier $h : \mathcal{X} \to [0,1]^{\mathcal{Y}}$ that maps data points to a metric of their confidence of being in a class. The index of the maximum element of h recovers f. This approach is followed in classification algorithms such as logistic regression and neural networks (Shalev-Shwartz & Ben-David, 2014).

Test-time adversary: We consider a test-time adversary that can modify any data point to generate an adversarial example (Goodfellow et al., 2015; Szegedy et al., 2013; Carlini & Wagner, 2017) within a neighborhood, i.e. $\tilde{x} = N(x)$, where $\tilde{x}$ is the adversarial example and $N(\cdot)$ is a non-empty neighborhood function. This general definition includes the ℓ_p family of constraints most widely used in previous work.

Loss functions and robust training: To obtain a classifier

[1] We note that some formalizations of the test-time adversary problem assume that the labels are determined by a ground-truth classifier.

robust to test-time adversaries, f must be trained to minimize the robust $0-1$ loss, defined as $\mathbb{E}\left[\tilde{\ell}_{0-1}(f,(x,y))\right] = \mathbb{E}\left[\sup_{\tilde{x}\in N(x)} \mathbf{1}\left(f(\tilde{x}) \neq y\right)\right]$. However, since the $0-1$ loss is non-differentiable, surrogate losses that are differentiable and upper bound it are used in practice. One of the most common is the cross-entropy or log loss, defined as $\ell^{\text{CE}}(h,v) = -\log h(x)_y$ for a 2-class problem, where $v=(x,y)$ and $h(x) \in [0,1]^{\mathcal{Y}}$ is the probability distribution over $\mathcal{Y}$ that the soft classifier h assigns to x. The robust classification problem using a surrogate loss ℓ is then

$$\inf_h \mathbb{E}_P\left[\sup_{\tilde{x}\in N(x)} \ell^{\text{CE}}(h,(\tilde{x},y))\right] = \inf_h \mathbb{E}_P\left[\tilde{\ell}^{\text{CE}}(h,(x,y))\right] \quad (1)$$

The robust cross-entropy loss is of particular interest in the robust training of neural networks (Madry et al., 2018).

Problem Statement: Our aim is to determine the value of $\inf_h \mathbb{E}_P[\tilde{\ell}^{\text{CE}}(h,(x,y))]$ over all measurable functions h, for a given discrete, two class distribution P and neighborhood function $N(\cdot)$.

2.2. Lower bound as the solution to a convex program

We first define a conflict graph in order to cast the problem of finding the lower bound as an optimization problem over the vertices of this graph. Then, we show that the feasible set of output probabilities is determined by the edge incidence matrix of the conflict graph. Finally, we determine the lower bound on the cross-entropy loss by minimizing over this feasible set.

Conflict graph: We define a conflict graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ that accounts for intersections between the neighborhoods of points from different classes. Each neighborhood represents the set of points reachable by the adversary from point x. Let $\mathcal{V} \subseteq \mathcal{X}\times\mathcal{Y}$ be the support of the distribution P. This means that each labeled data point (x,y) with strictly positive probability in P is represented as a vertex v. Since we consider a binary classification problem, the conflict graph is bipartite. Each part of the graph is $\mathcal{V}_c = \mathcal{V} \cap (\mathcal{X}\times\{c\})$, where $c \in \{-1,1\}$. The edge $((x,1),(x',-1))$ is present if and only if $N((x,1)) \cap N((x',-1))$ is nonempty. There are no edges between vertices in the same part of the graph.

Definition 1. *For a soft classifier h, the correct-classification probability q_v that it can achieve on an example $v=(x,y)$ in the presence of an adversary is*

$$q_v = \inf_{\tilde{x}\in N(x)} h(\tilde{x})_y.$$

Lemma 1 (Feasible output probabilities)**.** *Let $q \in \mathbb{R}^{\mathcal{V}}$ be the vector of correct-classification probabilities obtained by a classifier. The feasible set of such probabilities is*

$$\begin{aligned} q &\geq \mathbf{0} \\ Mq &\leq \mathbf{1}. \end{aligned} \quad (2)$$

where $M = \begin{pmatrix} E \\ I \end{pmatrix} \in \mathbb{R}^{(\mathcal{E}\sqcup\mathcal{V})\times\mathcal{V}}$ and $E \in \mathbb{R}^{\mathcal{E}\times\mathcal{V}}$ is the edge incidence matrix of the conflict graph.

Proof. Suppose that $(u,v) \in \mathcal{E}$. Then, there is some $\tilde{x} \in N(u) \cap N(v)$. We have $q_u \leq h(\tilde{x})_1$, $q_v \leq h(\tilde{x})_{-1}$, and $h(\tilde{x})_1 + h(\tilde{x})_{-1} = 1$. Combining these gives the constraint in (2) indexed by (u,v).

Now, we will show that each vector q in the polytope is achievable by some h. Let $h(\tilde{x})_1 = \sup_{u:\tilde{x}\in N(u)} q_u$ and $h(\tilde{x})_{-1} = 1 - h(\tilde{x})_1$. Then,

$\inf_{\tilde{x}\in N(u)} h(\tilde{x})_1 = \inf_{\tilde{x}\in N(u)} \sup_{u':\tilde{x}\in N(u')} q_{u'} \geq \inf_{\tilde{x}\in N(u)} q_u = q_u$

The output when the true example is v is $\inf_{\tilde{x}\in N(v)} h(\tilde{x})_{-1} = \inf_{\tilde{x}\in N(v)}(1 - \sup_{u:\tilde{x}\in N(u)} q_u) = \inf_{u:\exists\tilde{x}\in N(u)\cap N(v)}(1-q_u) \geq q_v$.

□

In the non-adversarial case, all constraints in (2) are of the forms $q_v \leq 1$ and $q_{(x,1)} + q_{(x,-1)} \leq 1$. Non-trivial adversaries lead to constraints between the probabilities achieved for distinct examples.

Having determined the feasible set of output probabilities, we can now determine the minimum possible cross-entropy loss by minimizing it over this feasible set.

Theorem 1 (Lower bound on cross-entropy loss)**.** *The discrete joint probability distribution P over data from two classes, and the neighborhood function $N(\cdot)$ define a bipartite conflict graph $\mathcal{G}$ with incidence matrix E. Let $p \in \mathbb{R}^{\mathcal{V}}$ with $p_v = P(\{v\})$. Let q^* be the minimizer of the following program:*

$$\begin{aligned} \min_q \sum_{v:p_v>0} -p_v \log q_v \\ \text{s.t. } q \geq \mathbf{0} \\ Mq \leq \mathbf{1}. \end{aligned} \quad (3)$$

Then, there is a classifier h^* that achieves the correct-classification probabilities q^* and for all h, $\mathbb{E}_P[\tilde{\ell}^{CE}(h^*,v)] \leq \mathbb{E}_P[\tilde{\ell}^{CE}(h,v)]$.

Proof. From Lemma 1, we know that the constraints in Eq.(2) represent the feasible set of all possible q. Further, there exists some h that achieves each q. The objective function must have a minimum in the feasible set. Additionally, the objective is convex and the constraints are linear, leading to a convex program. □

We note that a modification of the program above can be used to derive the minimum $0-1$ loss for discrete distributions by setting $\min_q \sum_v p^\intercal q$ as the objective function. This partially recovers results from Bhagoji et al. (2019), although that work considers a more general class of distributions including continuous distributions.

Lemma 2 (Properties of an optimal q)**.** *Suppose we have q and z such that*

$$q \geq \mathbf{0} \tag{4}$$
$$Mq \leq \mathbf{1} \tag{5}$$
$$z \geq \mathbf{0} \tag{6}$$
$$\mathrm{diag}(q)M^\top z \geq p \tag{7}$$
$$\mathbf{1}^\top z \leq \mathbf{1}^\top p. \tag{8}$$

Then, q is optimal in (3)*.*

Proof. From (7) we have $\mathbf{1}^\top p \leq q^\top M^\top z$ and from the (5) we have $z^\top Mq \leq z^\top \mathbf{1}$. Then (8) implies $\mathbf{1}^\top z = \mathbf{1}^\top p = q^\top M^\top z$. Furthermore $p_v = (\mathrm{diag}(q)M^\top z)_v = q_v(M^\top z)_v$.

There is always some feasible q that makes the objective function finite, so $q_v^* = 0$ implies $p_v = 0$. For $q_v^* > 0$, the upper bound on $\log q_v$ from the linear approximation at q_v^* is $\frac{q_v - q_v^*}{q_v^*} + \log q_v^*$. Thus

$$\begin{aligned}\sum_{v:p_v>0} p_v \log q_v &\leq \sum_{v:p_v>0} p_v \left(\frac{q_v - q_v^*}{q_v^*} + \log q_v^*\right) \\ &= \sum_{v:p_v>0} \frac{p_v}{q_v^*} q_v - \mathbf{1}^\top p + \sum_{v:p_v>0} p_v \log q_v^*.\end{aligned}$$

To prove $\sum_v -p_v \log q_v \geq \sum_v -p_v \log q_v^*$ for all q, we need $\sum_v \frac{p_v}{q_v^*} q_v \leq \mathbf{1}^\top p$. To show this, we note that $z^\top Mq \leq z^\top \mathbf{1}$ and $\mathbf{1}^\top z \leq \mathbf{1}^\top p$. Then, we only need that $(z^\top M)_v \geq \frac{p_v}{q_v^*}$, which follows from $\mathrm{diag}(q)M^\top z = p$. □

The vector z in Lemma 2 can be interpreted as the optimal strategy followed by the adversary.

2.3. Gaussian Case

We now consider the case when the data is generated from a mixture of two Gaussians with identical covariances and means that differ in their sign. Formally, we have $P = p_1\mathcal{N}(\mu, \Sigma) + p_{-1}\mathcal{N}(-\mu, \Sigma)$, where $p_1, p_{-1} \in [0, 1]$ and $p_1 + p_{-1} = 1$. $\mathcal{X}$ is then $\mathbb{R}^d$. We set the neighborhood function $N(x) = x + \epsilon\Delta$, where ϵ is the adversarial budget and $\Delta \in \mathbb{R}^d$ is a closed, convex, absorbing and origin-symmetric set.

Our first lemma proves that the optimal classifier is linear and the corresponding optimal adversarial strategy z^* [2] is just a translation of each component of the mixture. To show this, we just establish that these are identical to the solutions obtained in the $0-1$ loss case, allowing us to use Lemma 1 from (Bhagoji et al., 2019).

Lemma 3. *The optimal classifier h_y^* minimizing the cross-entropy loss is given by $\frac{1}{1+\exp(y(w^*)^\intercal x)}$ where $w^* = 2\Sigma^{-1}(\mu - z^*)$, and z^* is the optimal adversarial strategy given by Lemma 1 of (Bhagoji et al., 2019).*

The cross-entropy lower bound can then be directly computed.

Theorem 2. *The cross-entropy lower bound for a mixture of two Gaussians is*

$$\begin{aligned}&\inf_h \mathbb{E}_P[\tilde{\ell}^{CE}(h, v)] \\ &= p_1 \mathbb{E}_{N(\mu - z^*, \Sigma)}[\log(1 + \exp((w^*)^\intercal x))] \\ &+ p_{-1} \mathbb{E}_{N(\mu + z^*, \Sigma)}[\log(1 + \exp(-(w^*)^\intercal x))]\end{aligned} \tag{9}$$

We defer the proofs to Section A of the Supplementary.

3. Efficiently Computing Lower Bounds

In this section, we show that the convex program defined above can be efficiently solved by lower bounding its objective with a linear function and solving a recursive series of linear programs. We develop a specialized algorithm instead of using an off-the-shelf convex program solver in order to exploit the structure in the problem for faster computation.

3.1. Algorithm overview

Our algorithm (OptProb) executes the following strategy. It starts by guessing that there is a single correct-classification probability that should be assigned to all vertices from class 1 and a single probability for vertices from class -1. If this were the case, those probabilities should reflect the relative frequencies of the classes. The algorithm solves a linear program and either finds a dual certificate proving that the initial guess is correct or a partition of the vertices based on whether the optimal correct-classification probabilities are larger or smaller than the guess. In the latter case, the algorithm is applied recursively to the two subproblems and their solutions are assembled into a solution to the original problem. A precise description appears as Algorithm 1.

The computation of OptProb uses the function LinOpt at each stage of the recursion. The function $\mathsf{LinOpt}(\mathcal{A}, \mathcal{B}, \mathcal{E}, P)$ solves a dual pair of linear programs with variables $y \in \mathbb{R}^{\mathcal{A}\cup\mathcal{B}}$ and $z \in \mathbb{R}^{(\mathcal{A}\times\mathcal{B})\cup\mathcal{A}\cup\mathcal{B}}$:

[2] We note that there is a slight abuse of notation here since z in the previous section is a probability and is a perturbation here.

Algorithm 1 OptProb

Input: Bipartite graph $(\mathcal{A}, \mathcal{B}, \mathcal{E})$, vertex weights P
Output: Classifier probabilities q, adversarial strategy z

$(\mathcal{A}^+, \mathcal{A}^-, \mathcal{B}^+, \mathcal{B}^-, z^{\text{lin}}) = \mathsf{LinOpt}(\mathcal{A}, \mathcal{B}, \mathcal{E}, P)$
if $P(\mathcal{A}^+)P(\mathcal{B}^+) > P(\mathcal{A}^-)P(\mathcal{B}^-)$ **then**
  $\mathcal{E}' = \mathcal{E} \cap (\mathcal{A}^+ \times \mathcal{B}^-)$
  $\mathcal{E}'' = \mathcal{E} \cap (\mathcal{A}^- \times \mathcal{B}^+)$
  $(q', z') = \mathsf{OptProb}(\mathcal{A}^+, \mathcal{B}^-, \mathcal{E}', P)$
  $(q'', z'') = \mathsf{OptProb}(\mathcal{A}^-, \mathcal{B}^+, \mathcal{E}'', P)$
  $q = v \mapsto \begin{cases} q'_v & v \in \mathcal{A}^+ \cup \mathcal{B}^- \\ q''_v & v \in \mathcal{A}^- \cup \mathcal{B}^+ \end{cases}$
  $z = e \mapsto \begin{cases} z'_e & e \in (\mathcal{A}^+ \times \mathcal{B}^-) \cup \mathcal{A}^+ \cup \mathcal{B}^- \\ z''_e & e \in (\mathcal{A}^- \times \mathcal{B}^+) \cup \mathcal{A}^- \cup \mathcal{B}^+ \\ 0 & \text{otherwise} \end{cases}$
else
  $q = v \mapsto \begin{cases} P(\mathcal{A})/P(\mathcal{A} \cup \mathcal{B}) & v \in \mathcal{A} \\ P(\mathcal{B})/P(\mathcal{A} \cup \mathcal{B}) & v \in \mathcal{B} \end{cases}$
  $z = z^{\text{lin}}$
return (q, z)

$$\begin{aligned} \max r^\top y \quad & \qquad \min \mathbf{1}^\top z \\ y \geq \mathbf{0} \quad & \qquad z \geq \mathbf{0} \\ My \leq \mathbf{1} \quad & \qquad M^\top z \geq r \end{aligned}$$

where $r \in \mathbb{R}^{\mathcal{A} \cup \mathcal{B}}$ is defined as follows. If both $P(\mathcal{A}) > 0$ and $P(\mathcal{B}) > 0$, then

$$r_v = \begin{cases} P(\{v\})P(\mathcal{A} \cup \mathcal{B})/P(\mathcal{A}) & v \in \mathcal{A} \\ P(\{v\})P(\mathcal{A} \cup \mathcal{B})/P(\mathcal{B}) & v \in \mathcal{B} \end{cases}$$

and otherwise $r_v = P(\{v\})$.

The primal polytope is the vertex packing polytope of the bipartite graph $(\mathcal{A}, \mathcal{B}, \mathcal{E})$. This is integral, so there is some optimal $y \in \{0, 1\}^{\mathcal{A} \cup \mathcal{B}}$. The sets $\mathcal{A}^+, \mathcal{A}^-, \mathcal{B}^+, \mathcal{B}^-$ encode the support of y in a way that is convenient for expressing OptProb: $\mathcal{A}^+ = \{v \in \mathcal{A} : y_v = 1\}$, $\mathcal{A}^- = \mathcal{A} \setminus \mathcal{A}^+$, $\mathcal{B}^+ = \{v \in \mathcal{B} : y_v = 1\}$, and $\mathcal{B}^- = \mathcal{B} \setminus \mathcal{B}^+$. The support of y is an independent set: $(\mathcal{A}^+ \times \mathcal{B}^+) \cap \mathcal{E} = \varnothing$.

3.2. Proof sketch for algorithm optimality

Theorem 3 (Convergence to optimal for algorithm)**.** *The proposed Algorithm 1 returns the correct optimal classifier probability: the minimizer of* (3)*.*

The proof of Theorem 3 mirrors the recursive structure of OptProb and uses induction on the number of vertices. It relies on two technical lemmas (proofs deferred to Section B of the Supplementary). Lemma 4 establishes properties of the solutions to linear programs that are solved at each iteration. The proof uses standard duality and complementary slackness arguments for linear programs.

Lemma 4. *The function* LinOpt$(\mathcal{A}, \mathcal{B}, \mathcal{E}, P)$ *produces* $(\mathcal{A}^+, \mathcal{A}^-, \mathcal{B}^+, \mathcal{B}^-, z)$ *with the following properties.*

1. $P(\mathcal{A}^+)P(\mathcal{B}^+) \geq P(\mathcal{A}^-)P(\mathcal{B}^-)$
2. *If* $P(\mathcal{A}^+)P(\mathcal{B}^+) = P(\mathcal{A}^-)P(\mathcal{B}^-)$*, then*
   (a) $\mathbf{1}^\top z = P(\mathcal{A} \cup \mathcal{B})$,
   (b) $\frac{P(\mathcal{A})}{P(\mathcal{A} \cup \mathcal{B})}(M^\top z)_v = P(\{v\})$ *for all* $v \in \mathcal{A}$,
   (c) $\frac{P(\mathcal{B})}{P(\mathcal{A} \cup \mathcal{B})}(M^\top z)_v = P(\{v\})$ *for all* $v \in \mathcal{B}$.

Lemma 5 establishes that OptProb terminates and describes the structure of the optimal (q, z) in detail. Both vertex sets are split in partitions with paired parts and each pair of parts behaves similarly to a complete bipartite graph. Edges between pairs of parts are restricted by an order relation.

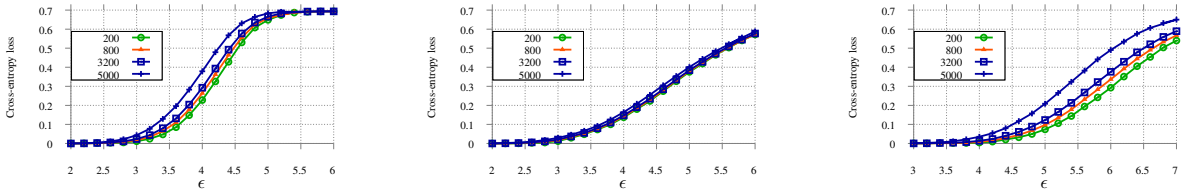
Lemma 5. *If* $P(\mathcal{A} \cup \mathcal{B}) > 0$*, the computation of* OptProb$(\mathcal{A}, \mathcal{B}, \mathcal{E}, P)$ *terminates and produces a pair* (q, z)*. For some* $[k] = \{0, 1, \ldots, k-1\}$*, there are functions* $a : \mathcal{A} \to [k]$ *and* $b : \mathcal{B} \to [k]$ *with the following properties.*

1. *If* $(u, v) \in \mathcal{E}$*,* $a(u) \leq b(v)$.
2. *We have* $z \geq 0$*,* $\mathbf{1}^\top z = P(\mathcal{A} \cup \mathcal{B})$*, and* $P(\{v\}) = q_v(M^\top z)_v$.
3. *Let* $\mathcal{A}_i = a^{-1}(i)$ *and* $\mathcal{B}_i = b^{-1}(i)$.
   For all i*,* $P(\mathcal{A}_i \cup \mathcal{B}_i) > 0$*. For all* $u \in \mathcal{A}$ *and* $v \in \mathcal{B}$*,* $q_u = \frac{P(\mathcal{A}_{a(u)})}{P(\mathcal{A}_{a(u)} \cup \mathcal{B}_{a(u)})}$ *and* $q_v = \frac{P(\mathcal{A}_{b(v)})}{P(\mathcal{A}_{b(v)} \cup \mathcal{B}_{b(v)})}$.
4. *For* $u, u' \in \mathcal{A}$*, if* $a(u) \leq a(u')$ *then* $q_u \leq q_{u'}$.

Proof of Theorem 3. From Lemma 5, we have that the computation of OptProb terminates and some information about $(q, z) = \mathsf{OptProb}(\mathcal{V}_1, \mathcal{V}_{-1}, \mathcal{E}, P)$. Properties 1,3, and 4 together imply (4) and (5) (i.e. that q is feasible in (3)): for any $(u, v) \in \mathcal{E}$, there is some u' such that $a(u') = b(v)$ and $q_u \leq q_{u'} = 1 - q_v$. Property 2 provides (6), (7), and (8). Lemma 2 establishes the optimality of q. □

3.3. Complexity analysis

In the worst case, at each step of the algorithm, only a single vertex will be removed from one part of the bipartite graph, and the algorithm will only terminate when only singleton parts of the graph remain. In this case, if there are $|\mathcal{V}|$ vertices in the graph, there will be $|\mathcal{V}|$ recursive steps, with each run taking $O(|\mathcal{V}||\mathcal{E}| \log(|\mathcal{V}|^2/|\mathcal{E}|))$ (Goldberg & Tarjan, 1988).



(a) MNIST (b) Fashion MNIST (c) CIFAR-10

Figure 1. Variation in minimum log-loss for an ℓ_2 adversary with adversarial budget ϵ and the number of samples from each class. The maximum possible log-loss is $\ln 2$, which is around 0.693. The total number of samples is 5000.

4. Experiments: Using Bounds as a Diagnostic Tool

In the previous section, we derived lower bounds on the cross-entropy loss that are applicable for all discrete distributions as well as for Gaussian data. In this section, we compute and use these bounds as a diagnostic tool to better understand limits of robust learning for practical datasets and algorithms. We determine lower bounds on the cross-entropy loss for practical datasets of interest. We analyze the runtime of Algorithm 1 and show its speedup over the generic non-linear convex solver from CVXOPT (Andersen et al., 2013). Finally, we uncover a gap between the loss obtained by several robust training methods and the lower bound, and investigate the use of 'soft-label' training with optimal classifier outputs to close this gap. All results are obtained on an Intel Xeon cluster with 8 P100 GPUs.

4.1. Lower bounds on robustness for real-world datasets

From Theorem 3 and Algorithm 1, we have an efficient method to compute the optimal log-loss for any empirical distribution. Here, we consider 3 benchmark computer vision datasets: MNIST (LeCun & Cortes, 1998), Fashion MNIST (Xiao et al., 2017) and CIFAR-10 (Krizhevsky & Hinton, 2009). Each of these datasets is originally a 10-class classification problem, and from each, without loss of generality, we choose the '3 vs. 7' classification task as a representative binary classification problem (results for other choices are in Section C.1. of the Supplementary). In each case, there are a total of $n = 5000$ training samples per class which can be used to compute the lower bound.

To derive a numerical bound, we need to specify the neighborhood function (adversarial constraints). While our bounds are valid for any non-empty neighborhood function, we pick the commonly used ℓ_2-norm ball constraint, parametrized by its radius ϵ. This has been used numerous times for both attacks (Carlini & Wagner, 2017) and defenses (Madry et al., 2018), and has well-established benchmarks (adv). Although ℓ_p-norm constraints have been critiqued (Gilmer et al., 2018a; Evtimov et al., 2020), we nonetheless choose to use them to provide a point of comparison with existing work.

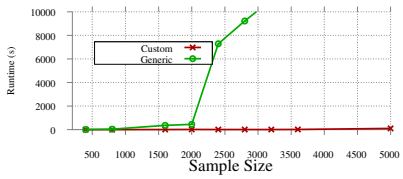
Algorithm implementation: We first create the conflict graph by checking for ℓ_2 ball intersections between all pairs of points from the two classes. The number of vertices $\mathcal{V}$ in the conflict graph $\mathcal{G}$ is $n_1 + n_{-1}$. We will generally consider the case when the total number of datapoints in each class is equal, giving $|\mathcal{V}| = 2n$. The total number of edges $\mathcal{E}$ is then $\hat{p}(n, \epsilon)n^2$, where $\hat{p}(n, \epsilon)$ is an estimate of the probability that the neighborhoods around points from the two classes have a non-empty intersection. We find that for the ℓ_2 norm, $\hat{p}(n, \epsilon)$ increases monotonically with ϵ and unlike the log-loss, is largely independent of the number of samples (Section C.2. of the Appendix).

Using this conflict graph, represented by a sparse matrix, we use Algorithm 1 to compute the lower bound. We use the maximum flow algorithm from Scipy (Virtanen, 2020) as LinOpt at the top level and for each recursively obtained split. This implementation uses the Edmonds-Karp (Edmonds & Karp, 1972) algorithm. We note that any linear program solver can be used and casting the problem as maximum flow is not canonical.

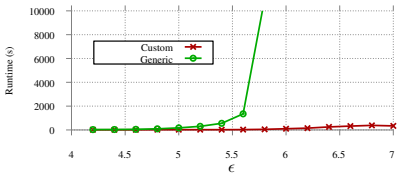
Numerical lower bounds: In Figure 1, we plot the variation in the minimum cross-entropy loss over the full set of 5000 training samples for all 3 datasets as the adversary's ℓ_2 budget is varied. The lower bound is only non-trivial after a budget of around 3.0 for the MNIST dataset and 4.0 for the CIFAR-10 dataset. At smaller budgets, the optimal classifier can achieve 0 loss even in the presence of an adversary. We note that this classifier may not generalize well to test data, since these bounds do not represent the population lower bound over the unknown underlying distribution.

Impact of subsampling: We also analyze the impact of subsampling from the complete set of samples to understand the dependence of the lower bound on the number of samples. We find that as the number of samples increases, the lower bound increases as well, indicating the presence of more intersections among samples, and thus more flexibility for the adversary.

Empirical runtime comparison: We compare the runtime



(a) Scaling with sample size at $\epsilon = 6.0$



(b) Scaling with ϵ for 5000 samples per class

Figure 2. Algorithm runtime comparisons for CIFAR-10

of Algorithm 1 using the max-flow solver from Scipy to that of the general purpose solver for convex programs with non-linear objective functions from CVXOPT (Andersen et al., 2013), which uses primal-dual interior point methods (Boyd et al., 2004).

The two parameters that determine the runtime of the algorithms to compute the minimum log-loss are the number of vertices $|\mathcal{V}|$ and the adversary's budget ϵ which controls the graph density. In Figure 2a, we show the variation in CPU time in seconds as the number of vertices in each class is varied. The mean and standard deviation over 10 runs is reported and the maximum time either algorithm is allowed to run is $10,000$ seconds after which it is terminated. It is clear that our custom algorithm runs significantly faster than the general purpose convex solver, with speed-ups of up to $3000\times$. The advantages are even starker as ϵ is varied in Figure 2b, with the general purpose solver taking in excess of $10,000$ seconds for any budget greater than 5.6. We can draw the same conclusions for the other two datasets from the runtime analysis presented in Section C.3. of the Supplementary.

4.2. Synthetic Gaussian data

From Section 2.3, we have a complete characterization of the robust learning problem for Gaussian distributions with respect to the cross-entropy loss. We use this to study the gap between population-level and sample-level cross-entropy lower bounds, finding that *this gap increases with the dimension of the data.* Thus, when the underlying distribution is unknown, sample-level lower bounds must be used carefully, especially with a small number of samples.

We use a diagonal covariance matrix Σ with Σ_{ii} sampled uniformly between 0 and 1, and set $\mu_i = \frac{C\Sigma_{ii}}{\sqrt{d}}$, where C is a constant determining the distance between the means.

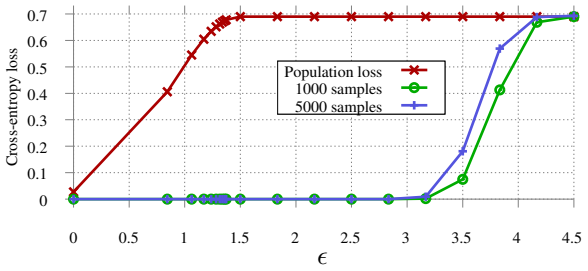


Figure 3. Comparing the population-level and sample-level lower bounds on cross-entropy loss for synthetic 2-class Gaussian data of dimension 100.

The two classes have identical covariances and means of opposite sign. In Figure 3, we compare the lower bound on cross-entropy loss directly obtained from Theorem 2 ('Population loss') and that over the empirical distributions resulting from sampling it ('k samples') for $d = 100$. In the latter case, the lower bounds are computed using Algorithm 1. The reason for the lack of intersections at lower budgets for the empirical distribution is that in high dimensions, even when the underlying distributions overlap, further perturbation is needed for intersections between the neighborhoods of sampled points. Results for other choices of d are in Section C.4. of the Supplementary.

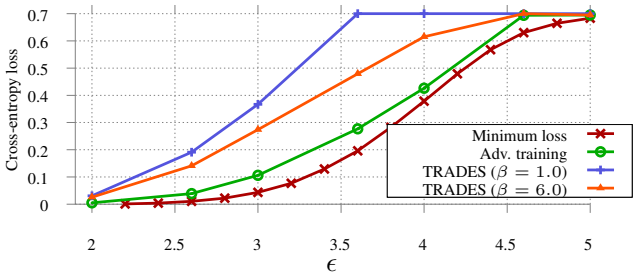
4.3. Evaluating the performance of robust training

We now compare the cross-entropy loss obtained by robust training techniques such as adversarial training (Madry et al., 2018) and TRADES (Zhang et al., 2019) to our lower bounds. We present results on the MNIST (LeCun & Cortes, 1998) and Fashion-MNIST (Xiao et al., 2017) datasets in the main body, and on CIFAR-10 in Section **??** of the Appendix.

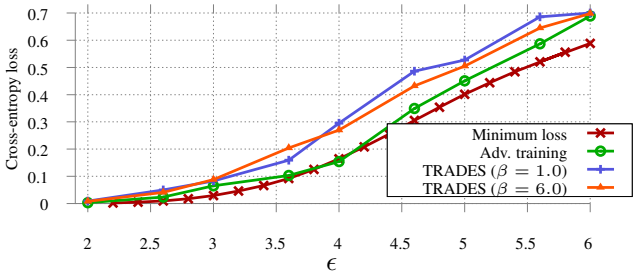
Our *key takeaways* are i) standard adversarial training can achieve close to the minimum cross-entropy loss with a sufficiently large architecture, but a gap still remains for the $0-1$ loss and, ii) soft label training with optimal probabilities obtained from our framework can help close this gap as well as aid in generalization in some cases.

Robust training setup. We train a ResNet-18 network using adversarial training and TRADES, these being the most effective robust training methods for an ℓ_2 adversary (Croce et al., 2020). The robust cross-entropy loss for these models is computed using the state-of-the-art AutoAttack (Croce & Hein, 2020). Adversarial training, referred to as 'hard labels' in Figure 4, utilizes one-hot labels, while TRADES uses the network's own prediction as soft-labels.

How close is the robust training loss of current techniques to optimal? In Figure 4, for both datasets, adversarial training achieves close to the minimum possible cross-entropy loss on the training data. However, TRADES is



(a) MNIST



(b) Fashion MNIST

Figure 4. Comparison on *training data* between the cross-entropy loss (computed using AutoAttack) obtained by different training methods versus the optimal loss.

outperformed by standard adversarial training with hard labels. This runs counter to earlier observations at lower adversarial budgets that TRADES was more robust. In the case of the $0-1$ loss for all datasets and the cross-entropy loss for CIFAR-10, the gap is far larger even for moderate budgets (Section D of the Supplementary). Nevertheless, since a gap exists even for the cross-entropy loss, we can rule out the possibility that the gap previously observed for the $0-1$ loss in Bhagoji et al. (2019) is only due to the use of a surrogate loss. Our experiments indicate that at larger adversarial budgets, especially for the MNIST and CIFAR-10 datasets, robust training does not converge to the optimal loss. We leave for future work to determine if closing this gap will lead to improved generalization.

We also conduct ablation studies with larger networks and smoother activation functions, techniques known to help with robust training. Resnet-101 for FMNIST reduces cross-entropy loss to 0.42, in comparison to 0.45 with ResNet-18, which is close to the optimal loss for Fashion MNIST at $\epsilon = 5.0$. Additionally, with over 15 different activation functions, we did not observe any significant drop in cross-entropy loss compared to the standard ReLU. Further details and results are in Section D.2. of the Supplementary.

Using soft labels. Training using soft labels is known to improve the performance of deep neural networks (Zheng et al., 2016). Since at higher values of ϵ, the optimal classifier may assign a higher probability to the opposite class as the true label, the obtained soft labels are noisy. To avoid introducing this label noise, while also extracting meaningful gradients, we impose a lower bound on the probability of the correct class (details in Section **??** of the Appendix). We find that training with these clipped soft-labels can reduce the cross-entropy loss by a significant margin (Table 1). Additionally, this method can also improve the $0-1$ loss for the MNIST datasets for a range of budgets (Section D.1. of the Supplementary). Overall, these results indicate that appropriately calibrated soft label training can help with robustness.

Table 1. Comparison of train and test set robust accuracy with different robust training techniques for the FMNIST dataset.

	FMNIST ($\epsilon = 4.6$)		FMNIST ($\epsilon = 5.0$)	
	Train	Test	Train	Test
Hard labels	0.349	0.348	0.451	0.451
Clipped soft labels	0.326	0.331	0.419	0.420
Optimal	0.305	–	0.401	–

5. Related Work

We only discuss the closest related work here on theoretical analysis of test-time adversaries and robust training. Extensive surveys (Papernot et al., 2016; Liu et al., 2018; Biggio & Roli, 2017; Li et al., 2020) provide a broader overview.

Information-theoretic limits on robust learning. All previous work on information-theoretic limits on robust learning has focused on the $0-1$ loss. (Dohmatob, 2019) and (Mahloujifar et al., 2019) use the 'blowup' property of specific data distributions to determine bounds on the robust loss, given some level of loss on benign data. (Bhagoji et al., 2019) and (Pydi & Jog, 2020) use optimal transport to provide lower bounds on the robust loss for a general class of distributions, without a dependence on the loss on benign data. While (Pydi & Jog, 2020) does consider convex losses, we are the first to provide an explicit method and framework, as well as numerical results, for the cross-entropy loss.

Generalization for adversarially robust learning. A number of papers analyze the sample complexity of robust learning for specific distributions of interest such as Gaussians (Schmidt et al., 2018; Javanmard et al., 2020; Dan et al., 2020), uniform (Diochnos et al., 2018) and spherical (Gilmer et al., 2018b). The sample complexity of PAC-learning (worst case over distributions) for robust classifiers has also been derived (Cullina et al., 2018; Yin et al., 2019; Montasser et al., 2019). However, this line of work does not analyze the minimum possible loss, only the gap between the minimum and learned.

Computational limits of robust learning. Computationally bounded adversaries (Garg et al., 2020) were consid-

ered to devise instances where there is a separation between their power and that of unbounded adversaries. Other work (Bubeck et al., 2018; Awasthi et al., 2019; Montasser et al., 2020) has focused on instances where computationally efficient robust learning is possible.

Robust training of neural networks. Adversarial training (Madry et al., 2018) with follow-up improvements in TRADES (Zhang et al., 2019), remains the most successful robust training technique. Its performance is further improved with larger networks (Gowal et al., 2020), smooth activations (Xie et al., 2020), early stopping (Rice et al., 2020), and careful tuning of weight decay (Pang et al., 2021). Some other works investigate the effect of weight perturbation (Wu et al., 2020), weight averaging (Gowal et al., 2020), sub-networks on robustness (Sehwag et al., 2020) and additional data (Carmon et al., 2019). Wang et al. (2020) further demonstrate minor improvements in robustness with sample-weighted adversarial training. Goibert & Dohmatob (2019) show that the use of smoothed labels obtained from the classifier can improve upon benign training, but performs worse than standard adversarial training. For a detailed comparison of state-of-the-art robust training techniques, we refer the reader to RobustBench (Croce et al., 2020).

6. Discussion

In this paper, we have provided a framework to compute optimal lower bounds on the cross-entropy loss for general discrete distributions as well as Gaussian mixtures. We showed how to leverage this framework to analyze current robust training methods.

While the results in this paper are restricted to the two-class case, here we sketch how they can be extended to the multi-class setting. There are several relationships between our approach to the two class problem and the analogous quantities for the k-class problem with $k \geq 3$. When $k \geq 3$, the *targeted* and *untargeted* adversarial classification problems become distinct. The untargeted version is the direct generalization of our formulation for the two class version. In the targeted version, in addition to a labeled example, the adversary received another random class label that serves as the target and the classifier incurs a loss based on the probability for the target class assigned to the adversarial example. Clearly, the untargeted version is more favorable for the adversary and the optimal loss for the untargeted version is at least as large as the optimal loss for the targeted version. Using the optimal losses for all of the one-vs-one and one-vs-rest two-class classification tasks, simple lower bounds on the optimal loss for both versions of the k-class problem can be derived.

Our characterization of the exact optimal loss extends to the untargeted version of the k-class problem in a reasonably straightforward manner. The bipartite conflict graph is replaced with a k-partite hypergraph: hyperedges contain up to one vertex from each class. A polytope derived from this hypergraph contains the achievable correct classification probabilities. Minimizing the cross-entropy over this polytope is still a convex problem, but our special purpose algorithm no longer applies. The situation in the targeted case is more complicated and we are currently extending our framework to handle this case.

On the empirical front, we aim to further investigate the convergence of robust training for complex datasets such as CIFAR-10, as well as to use our framework to guide the generation of more robust feature representations. The link between achieving a training loss close to optimal and the generalization gap for robust training is one of great interest for future work in this domain.

Acknowledgements

This work was supported in part by the National Science Foundation under grants CNS-1553437, CNS-1704105 and CNS-1949650, the DARPA GARD program, the Army Research Laboratory's Army Artificial Intelligence Innovation Institute (A2I2), the Office of Naval Research Young Investigator Award, the Army Research Office Young Investigator Prize, a faculty research award from Facebook, the Schmidt DataX award, and Princeton E-ffiliates Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any funding agencies.

References

Advbench: Adversarial robustness benchmark. `https://advbench.github.io/`. Accessed: 2020-09-27.

Andersen, M. S., Dahl, J., and Vandenberghe, L. Cvxopt: Python software for convex optimization, 2013.

Awasthi, P., Dutta, A., and Vijayaraghavan, A. On robustness to adversarial examples and polynomial optimization. In *Proceedings of Neural Information Processing Systems*, 2019.

Bhagoji, A. N., He, W., Li, B., and Song, D. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *European Conference on Computer Vision*, pp. 158–174. Springer, 2018.

Bhagoji, A. N., Cullina, D., and Mittal, P. Lower bounds on adversarial robustness from optimal transport. In *Advances in Neural Information Processing Systems*, pp. 7496–7508, 2019.

Biggio, B. and Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *arXiv preprint arXiv:1712.03141*, 2017.

Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Bubeck, S., Price, E., and Razenshteyn, I. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57. IEEE, 2017.

Carmon, Y., Raghunathan, A., Schmidt, L., Liang, P., and Duchi, J. C. Unlabeled data improves adversarial robustness. In *Neural Information Processing Systems*, 2019.

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

Croce, F., Andriushchenko, M., Sehwag, V., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

Cullina, D., Bhagoji, A. N., and Mittal, P. Pac-learning in the presence of adversaries. In *Advances in Neural Information Processing Systems*, pp. 230–241, 2018.

Dan, C., Wei, Y., and Ravikumar, P. Sharp statistical guarantees for adversarially robust Gaussian classification. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 2345–2355, 2020.

Diochnos, D., Mahloujifar, S., and Mahmoody, M. Adversarial risk and robustness: General definitions and implications for the uniform distribution. In *Advances in Neural Information Processing Systems*, pp. 10359–10368, 2018.

Dohmatob, E. Generalized no free lunch theorem for adversarial robustness. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 1646–1654, 2019.

Edmonds, J. and Karp, R. M. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

Evtimov, I., Cui, W., Kamar, E., Kiciman, E., Kohno, T., and Li, J. Security and machine learning in the real world. *arXiv preprint arXiv:2007.07205*, 2020.

Garg, S., Jha, S., Mahloujifar, S., and Mohammad, M. Adversarially robust learning could leverage computational hardness. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, pp. 364–385, 2020.

Gilmer, J., Adams, R. P., Goodfellow, I., Andersen, D., and Dahl, G. E. Motivating the rules of the game for adversarial example research. *arXiv preprint arXiv:1807.06732*, 2018a.

Gilmer, J., Metz, L., Faghri, F., Schoenholz, S. S., Raghu, M., Wattenberg, M., and Goodfellow, I. Adversarial spheres. In *ICLR*, 2018b.

Goibert, M. and Dohmatob, E. Adversarial robustness via label-smoothing. *arXiv preprint arXiv:1906.11567*, 2019.

Goldberg, A. V. and Tarjan, R. E. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, 35(4):921–940, 1988.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Gowal, S., Qin, C., Uesato, J., Mann, T., and Kohli, P. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

Javanmard, A., Soltanolkotabi, M., and Hassani, H. Precise tradeoffs in adversarial training for linear regression. In *Proceedings of Thirty Third Conference on Learning Theory*, pp. 2034–2078, 2020.

Körner, J. Coding of an information source having ambiguous alphabet and the entropy of graphs. In *6th Prague conference on information theory*, 1973.

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

LeCun, Y. and Cortes, C. The MNIST database of handwritten digits. 1998.

Li, L., Qi, X., Xie, T., and Li, B. Sok: Certified robustness for deep neural networks. *arXiv preprint arXiv:2009.04131*, 2020.

Liu, Q., Li, P., Zhao, W., Cai, W., Yu, S., and Leung, V. C. A survey on security threats and defensive techniques of machine learning: A data driven view. *IEEE access*, 6: 12103–12117, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Mahloujifar, S., Diochnos, D. I., and Mahmoody, M. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4536–4543, 2019.

Montasser, O., Hanneke, S., and Srebro, N. Vc classes are adversarially robustly learnable, but only improperly. *arXiv preprint arXiv:1902.04217*, 2019.

Montasser, O., Goel, S., Diakonikolas, I., and Srebro, N. Efficiently learning adversarially robust halfspaces with noise. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 7010–7021, 2020.

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

Papernot, N., McDaniel, P., Sinha, A., and Wellman, M. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.

Pydi, M. S. and Jog, V. Adversarial risk via optimal transport and optimal couplings. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 7814–7823, 2020.

Rice, L., Wong, E., and Kolter, Z. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pp. 8093–8104. PMLR, 2020.

Schmidt, L., Santurkar, S., Tsipras, D., Talwar, K., and Madry, A. Adversarially robust generalization requires more data. *arXiv preprint arXiv:1804.11285*, 2018.

Sehwag, V., Wang, S., Mittal, P., and Jana, S. Hydra: Pruning adversarially robust neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 7, 2020.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Virtanen, P. e. a. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17: 261–272, 2020. doi: 10.1038/s41592-019-0686-2.

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

Wu, D., Xia, S.-T., and Wang, Y. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Xie, C., Tan, M., Gong, B., Yuille, A., and Le, Q. V. Smooth adversarial training. *arXiv preprint arXiv:2006.14536*, 2020.

Yin, D., Ramchandran, K., and Bartlett, P. Rademacher complexity for adversarially robust generalization. In *ICML*, 2019.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

Zheng, S., Song, Y., Leung, T., and Goodfellow, I. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 4480–4488, 2016.