Concise Explanations of Neural Networks using Adversarial Training

Prasad Chalasani [1] **Jiefeng Chen** [2] **Amrita Roy Chowdhury** [2] **Somesh Jha** [1 2] **Xi Wu** [3]

Abstract

We show new connections between adversarial learning and explainability for deep neural networks (DNNs). One form of explanation of the output of a neural network model in terms of its input features, is a vector of feature-attributions. Two desirable characteristics of an attribution-based explanation are: (1) *sparseness*: the attributions of irrelevant or weakly relevant features should be negligible, thus resulting in *concise* explanations in terms of the significant features, and (2) *stability*: it should not vary significantly within a small local neighborhood of the input. Our first contribution is a theoretical exploration of how these two properties (when using attributions based on Integrated Gradients, or IG) are related to adversarial training, for a class of 1-layer networks (which includes logistic regression models for binary and multi-class classification); for these networks we show that (a) adversarial training using an ℓ_∞-bounded adversary produces models with sparse attribution vectors, and (b) natural model-training while encouraging stable explanations (via an extra term in the loss function), is equivalent to adversarial training. Our second contribution is an empirical verification of phenomenon (a), which we show, somewhat surprisingly, occurs *not only in 1-layer networks, but also DNNs trained on standard image datasets*, and extends beyond IG-based attributions, to those based on DeepSHAP: adversarial training with ℓ_∞-bounded perturbations yields significantly sparser attribution vectors, with little degradation in performance on natural test data, compared to natural training. Moreover, the sparseness of the attribution vectors is significantly better than that achievable via ℓ_1-regularized natural training.

1. Introduction

Despite the recent dramatic success of deep learning models in a variety of domains, two serious concerns have surfaced about these models:

Vulnerability to Adversarial Attacks: We can abstractly think of a neural network model as a function $F(\boldsymbol{x})$ of a d-dimensional input vector $\boldsymbol{x} \in \mathbb{R}^d$, and the range of F is either a discrete set of class-labels, or a continuous set of class probabilities. Many of these models can be foiled by an adversary who imperceptibly (to humans) alters the input $\boldsymbol{x}$ by adding a perturbation $\delta \in \mathbb{R}^d$ so that $F(\boldsymbol{x}+\delta)$ is very different from $F(\boldsymbol{x})$ (Szegedy et al., 2013; Goodfellow et al., 2014; Papernot et al., 2015; Biggio et al., 2013). *Adversarial training* (or *adversarial learning*) has recently been proposed as a method for training models that are robust to such attacks, by applying techniques from the area of Robust Optimization (Madry et al., 2017; Sinha et al., 2018). The core idea of adversarial training is simple: we define a set S of allowed perturbations $\delta \in \mathbb{R}^d$ that we want to "robustify" against (e.g. S could be the set of δ where $||\delta||_\infty \leq \varepsilon$), and perform model-training using Stochastic Gradient Descent (SGD) exactly as in natural training, except that each training example x is perturbed adversarially, i.e. replaced by $x + \delta^*$ where $\delta^* \in S$ maximizes the example's loss-contribution.

Explainability: One way to address the well-known lack of explainability of deep learning models is *feature attribution*, which aims to explain the output of a model $F(\boldsymbol{x})$ as an attribution vector $A^F(\boldsymbol{x})$ of the contributions from the *features* $\boldsymbol{x}$. There are several feature-attribution techniques in the literature, such as *Integrated Gradients* (IG) (Sundararajan et al., 2017), *DeepSHAP* (Lundberg & Lee, 2017), and *LIME* (Ribeiro et al., 2016). For such an explanation to be human-friendly, it is highly desirable (Molnar, 2019) that the attribution-vector is *sparse*, i.e., only the features that are truly predictive of the output $F(\boldsymbol{x})$ should have significant contributions, and irrelevant or weakly-relevant features should have negligible contributions. A sparse attribution makes it possible to produce a *concise* explanation, where only the input features with significant contributions are included. For instance, if the model F is used for a loan approval decision, then various stakeholders (like customers,

[1]XaiPient [2]University Of Wisconsin (Madison) [3]Google. Correspondence to: Prasad Chalasani <pchalasani@gmail.com>.

data-scientists and regulators) would like to know the reason for a specific decision in simple terms. In practice however, due to artifacts in the training data or process, the attribution vector is often not sparse and irrelevant or weakly-relevant features end up having significant contributions (Tan et al., 2013). Another desirable property of a good explanation is *stability*: the attribution vector should not vary significantly within a small local neighborhood of the input x. Similar to the lack of concise explainability, natural training often results in explanations that lack stability (Alvarez-Melis & Jaakkola, 2018).

Our paper shows new connections between adversarial robustness and the above-mentioned desirable properties of explanations, namely conciseness and stability. Specifically, let $\tilde{F}$ be an adversarially trained version of a classifier F, and for a given input vector $\boldsymbol{x}$ and attribution method A, let $A^F(\boldsymbol{x})$ and $A^{\tilde{F}}(\boldsymbol{x})$ denote the corresponding attribution vectors. The central research question this paper addresses is:

$$\text{Is } A^{\tilde{F}}(\boldsymbol{x}) \text{ sparser and more stable than } A^F(\boldsymbol{x})?$$

The main contributions of our paper are as follows:

Theoretical Analysis of Adversarial Training: Our first set of results show via a *theoretical* analysis that $\ell_\infty(\varepsilon)$-adversarial training 1-layer networks tends to produce sparse attribution vectors for IG, which in turn leads to concise explanations. In particular, under some assumptions, we show (Theorems 3.1 and E.1) that for a general class of convex loss functions (which includes popular loss functions used in 1-layer networks, such as logistic and hinge loss, used for binary or multi-class classification), and adversarial perturbations δ satisfying $||\delta||_\infty \leq \varepsilon$, the weights of "weak" features are on average more aggressively shrunk toward zero than during natural training, and the rate of shrinkage is proportional to the amount by which ε exceeds a certain measure of the "strength" of the feature. This shows that $\ell_\infty(\varepsilon)$-adversarial training tends to produce sparse *weight vectors* in popular 1-layer models. In Section 4 we show (Lemma 4.1) a closed form formula for the IG vector of 1-layer models, that makes it clear that in these models, sparseness of the *weight vector* directly implies sparseness of the *IG vector*.

Empirically Demonstrate Attribution Sparseness: [1] In Section 6 we *empirically* demonstrate that this "sparsification" effect of $\ell_\infty(\varepsilon)$-adversarial training holds not only for 1-layer networks (e.g. logistic regression models), but also for Deep Convolutional Networks used for image classification, and extends beyond IG-based attributions, to those based on DeepSHAP. Specifically, we show this phenomenon via experiments applying $\ell_\infty(\varepsilon)$-adversarial training to (a) Convolutional Neural Networks on public benchmark image datasets MNIST (LeCun & Cortes, 2010) and Fashion-MNIST (Xiao et al., 2017), and (b) logistic regression models on the Mushroom and Spambase tabular datasets from the UCI Data Repository (Dheeru & Karra Taniskidou, 2017). In all of our experiments, we find that it is possible to choose an ℓ_∞ bound ε so that adversarial learning under this bound produces attribution vectors that are sparse on average, *with little or no drop in performance on natural test data*. A visually striking example of this effect is shown in Figure 1 (the Gini Index, introduced in Section 6, measures the sparseness of the map).

It is natural to wonder whether a traditional *weight-regularization* technique such as ℓ_1-regularization can produce models with sparse attribution vectors. In fact, our experiments show that for logistic regression models, ℓ_1-regularized training does yield attribution vectors that are on average significantly sparser compared to attribution vectors from natural (un-regularized) model-training, and the sparseness improvement is almost as good as that obtained with $\ell_\infty(\varepsilon)$-adversarial training. This is not too surprising given our result (Lemma 4.1) that implies a direct link between sparseness of *weights* and sparseness of *IG vectors*, *for 1-layer models*. Intriguingly, this does *not* carry over to DNNs: for multi-layer models (such as the ConvNets we trained for the image datasets mentioned above) we find that with ℓ_1-regularization, the sparseness improvement is significantly inferior to that obtainable from $\ell_\infty(\varepsilon)$-adversarial training (when controlling for model accuracy on natural test data), as we show in Table 1, Figure 2 and Figure 3. Thus it appears that for DNNs, adversarial training confers properties that go beyond merely inducing sparseness of weights.

Connection between Adversarial Training and Attribution Stability: We also show theoretically (Section 5) that training 1-layer networks naturally, while encouraging stability of explanations (via a suitable term added to the loss function), is in fact equivalent to adversarial training.

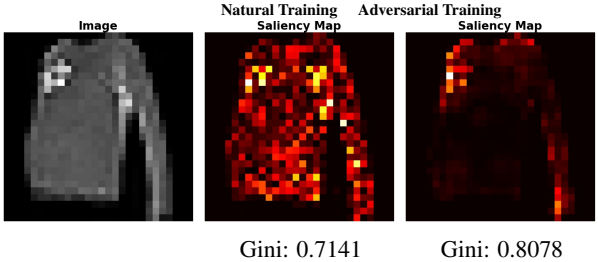


Figure 1: Both models correctly predict "Pullover", but the IG-based saliency map of the adversarially trained model is much sparser than that of the naturally trained model.

[1] Code for all experiments is available at `https://github.com/jfc43/advex`

2. Setup and Assumptions

For ease of understanding, we consider the case of binary classification for the rest of our discussion in the main paper. We assume there is a distribution $\mathcal{D}$ of data points $(\boldsymbol{x}, y)$ where $\boldsymbol{x} \in \mathbb{R}^d$ is an input feature vector, and $y \in \{\pm 1\}$ is its true label[2]. For each $i \in [d]$, the i'th component of $\boldsymbol{x}$ represents an input feature, and is denoted by x_i. The model is assumed to have learnable parameters ("weights") $\boldsymbol{w} \in \mathbb{R}^d$, and for a given data point $(\boldsymbol{x}, y)$, the *loss* is given by some function $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$. *Natural model training* [3] consists of minimizing the expected loss, known as *empirical risk*:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})]. \tag{1}$$

We sometimes assume the existence of an $\ell_\infty(\varepsilon)$-*adversary* who may perturb the input example $\boldsymbol{x}$ by adding a vector $\delta \in \mathbb{R}^d$ whose ℓ_∞-norm is bounded by ε; such a perturbation δ is referred to as an $\ell_\infty(\varepsilon)$-perturbation. For a given data point $(\boldsymbol{x}, y)$ and a given loss function $\mathcal{L}(.)$, an $\ell_\infty(\varepsilon)$-*adversarial* perturbation is a δ^* that maximizes the *adversarial loss* $\mathcal{L}(\boldsymbol{x}+\delta^*, y; \boldsymbol{w})$.

Given a function $F : \mathbb{R}^d \to [0, 1]$ representing a neural network, an input vector $\boldsymbol{x} \in \mathbb{R}^d$, and a suitable baseline vector $\boldsymbol{u} \in \mathbb{R}^d$, an *attribution* of the prediction of F at input $\boldsymbol{x}$ relative to $\boldsymbol{u}$ is a vector $A^F(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^d$ whose i'th component $A^F_i(\boldsymbol{x}, \boldsymbol{u})$ represents the "contribution" of x_i to the prediction $F(\boldsymbol{x})$. A variety of *attribution methods* have been proposed in the literature (see (Arya et al., 2019) for a survey), but in this paper we will focus on two of the most popular ones: Integrated Gradients (Sundararajan et al., 2017), and DeepSHAP (Lundberg & Lee, 2017). When discussing a specific attribution method, we will denote the IG-based attribution vector as $\text{IG}^F(\boldsymbol{x}, \boldsymbol{u})$, and the DeepSHAP-based attribution vector as $\text{SH}^F(\boldsymbol{x}, \boldsymbol{u})$. In all cases we will drop the superscript F and/or the baseline vector $\boldsymbol{u}$ when those are clear from the context.

The aim of *adversarial training*(Madry et al., 2017) is to train a model that is *robust* to an $\ell_\infty(\varepsilon)$-adversary (i.e. performs well in the presence of such an adversary), and consists of minimizing the expected $\ell_\infty(\varepsilon)$-adversarial loss:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\max_{||\delta||_\infty \le \varepsilon} \mathcal{L}(\boldsymbol{x}+\delta, y; \boldsymbol{w})]. \tag{2}$$

In the expectations (1) and (2) we often drop the subscript under $\mathbb{E}$ when it is clear that the expectation is over $(\boldsymbol{x}, y) \sim \mathcal{D}$.

Some of our theoretical results make assumptions regarding the form and properties of the loss function $\mathcal{L}$, the properties of its first derivative, and the data distribution $\mathcal{D}$. For the sake of clarify, we highlight these assumptions (with mnemonic names) here for ease of future reference.

[2]It is trivial to convert -1/1 labels to 0/1 labels and vice versa

[3]Also referred to as *standard training* by (Madry et al., 2017)

Assumption LOSS-INC. *The loss function is of the form* $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = g(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ *where* g *is a non-decreasing function.*

Assumption LOSS-CVX. *The loss function is of the form* $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = g(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ *where* g *is non-decreasing, almost-everywhere differentiable, and convex.*

Section B.1 in the Supplement shows that these Assumptions are satisfied by popular loss functions such as logistic and hinge loss. Incidentally, note that for any differentiable function g, g is convex if and only if its first-derivative g' is non-decreasing, and we will use this property in some of the proofs.

Assumption FEAT-INDEP. *The features* $\boldsymbol{x}$ *are conditionally independent given the label* y*, i.e. for any two distinct indices* i, j*,* x_i *is independent of* x_j *given* y*, or more compactly,* $(x_i \perp x_j) \mid y$*.*

This Assumption is used to prove Theorem 3.1 in the next Section, and is not as restrictive as it may first appear: one can imagine *clustering* features into groups that are conditionally independent of each other (given the label y), and extend our results to such clustered features; we leave this for future work.

Assumption FEAT-EXP. *For each feature* x_i*,* $i \in [d]$*,* $\mathbb{E}(x_i|y) = a_i y$ *for some constant* a_i*.*

In Section B.2 (Supplement) we show that Assumption FEAT-EXP is without loss of generality, and it is worth observing that this Assumption implies

$$\mathbb{E}(yx_i) = \mathbb{E}[\mathbb{E}(yx_i|y)] = \mathbb{E}[y^2 a_i] = a_i. \tag{3}$$

$$\mathbb{E}(yx_i|y) = y\mathbb{E}[x_i|y] = y^2 a_i = a_i. \tag{4}$$

Note that for any j, the expectation $\mathbb{E}(yx_j)$ can be thought of as representing the *degree of association*[4] between feature x_j and label y. When the data distribution satisfies Assumption FEAT-EXP, $\mathbb{E}(yx_j) = a_j$, so we refer to a_j as the *directed strength* [5] of feature x_j, and $|a_j|$ is referred to as the *absolute strength* of x_j. In particular when $|a_j|$ is large (small) we say that x_j is a *strong* (*weak*) feature.

3. Analysis of SGD Updates in Adversarial Training

One way to understand the characteristics of the weights in an adversarially-trained neural network model, is to analyze how the weights evolve during adversarial training under Stochastic Gradient Descent (SGD) optimization. One of

[4]When the features are standardized to have mean 0, $\mathbb{E}(yx_j)$ is in fact the covariance of y and x_j.

[5]This is related to the feature "robustness" notion introduced in (Ilyas et al., 2019)

the main results of this work is a theoretical characterization of the weight updates during a single SGD step, when applied to a randomly drawn data point $(\boldsymbol{x}, y) \sim \mathcal{D}$ that is subjected to an $\ell_\infty(\varepsilon)$-adversarial perturbation.

Although the holy grail would be to do this for general DNNs (and we expect this will be quite difficult) we take a first step in this direction by analyzing *single-layer networks* for binary or multi-class classification, where each weight is associated with an input feature. Intriguingly, our results (Theorem 3.1 for binary classification and E.1 for multi-class classification in the Supplement) show that for these models, $\ell_\infty(\varepsilon)$-adversarial training tends to selectively reduce the weight-magnitude of *weakly relevant* or *irrelevant* features, and does so much more aggressively than natural training. In other words, natural training can result in models where many weak features have significant weights, whereas adversarial training would tend to push most of these weights close to zero. The resulting model weights would thus be more sparse, and the corresponding IG-based attribution vectors would on average be more sparse as well (since in linear models, sparse weights imply sparse IG vectors; this is a consequence of Lemma 4.1) compared to naturally-trained models.

Our experiments (Sec. 6) show that indeed for logistic regression models (which satisfy the conditions of Theorem 3.1), adversarial training leads to sparse IG vectors. Interestingly, our extensive experiments with Deep Convolutional Neural Networks on public image datasets demonstrate that this phenomenon extends to DNNs as well, and to attributions based on DeepSHAP, even though our theoretical results only apply to 1-layer networks and IG-based attributions.

As a preliminary, it is easy to show the following expressions related to the $\ell_\infty(\varepsilon)$-adversarial perturbation δ^* (See Lemmas 2 and 3 in Section C of the Supplement): For loss functions satisfying Assumption LOSS-INC, the $\ell_\infty(\varepsilon)$-adversarial perturbation δ^* is given by:

$$\delta^* = -y \operatorname{sgn}(\boldsymbol{w})\varepsilon, \tag{5}$$

the corresponding $\ell_\infty(\varepsilon)$-adversarial loss is

$$\mathcal{L}(\boldsymbol{x} + \delta^*, y;\ \boldsymbol{w}) = g(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle), \tag{6}$$

and the gradient of this loss w.r.t. a weight w_i is

$$\frac{\partial \mathcal{L}(\boldsymbol{x} + \delta^*, y;\ \boldsymbol{w})}{\partial w_i} = -g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ (yx_i - \operatorname{sgn}(w_i)\varepsilon). \tag{7}$$

In our main result, the expectation of the g' term in (7) plays an important role, so we will use the following notation:

$$\overline{g'} := \mathbb{E}\big[g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\big] \tag{8}$$

Ideally, we would like to understand the nature of the weight-vector $\boldsymbol{w}^*$ that minimizes the expected adversarial loss (2). This is quite challenging, so rather than analyzing the *final* optimum of (2), we instead analyze how an SGD-based optimizer for (2) *updates* the model weights $\boldsymbol{w}$. We assume an idealized SGD process: (a) a data point $(\boldsymbol{x}, y)$ is drawn from distribution $\mathcal{D}$, (b) $\boldsymbol{x}$ is replaced by $\boldsymbol{x}' = \boldsymbol{x} + \delta^*$ where δ^* is an $\ell_\infty(\varepsilon)$-adversarial perturbation with respect to the loss function $\mathcal{L}$, (c) each weight w_i is updated by an amount $\Delta w_i = -\partial \mathcal{L}(\boldsymbol{x}', y; \boldsymbol{w})/\partial w_i$ (assuming a unit learning rate to avoid notational clutter). We are interested in the *expectation* of Δw_i, in order to understand what happens to a weight w_i *on average* during a single SGD step. In fact it will be useful to analyze the **expected SGD update** $\overline{\Delta w_i}$ defined as follows:

$$\overline{\Delta w_i} := \begin{cases} -\mathbb{E}\frac{\partial \mathcal{L}}{\partial w_i}, & \text{when } w_i = 0, \\ -\operatorname{sgn}(w_i)\mathbb{E}\frac{\partial \mathcal{L}}{\partial w_i}, & \text{when } w_i \neq 0, \end{cases} \tag{9}$$

When $w_i \neq 0$, a **positive** (**negative**) $\overline{\Delta w_i}$ indicates that during an SGD update on a randomly chosen data-point, on average the weight w_i **expands** (**shrinks**), i.e. maintains its sign and increases (decreases) in absolute value.

The following result characterizes $\overline{\Delta w_i}$ (the proof is in Section D of the Supplement).

Theorem 3.1 (Expected SGD Update in Adversarial Training)**.** *For any loss function $\mathcal{L}$ satisfying Assumption LOSS-CVX and a data distribution $\mathcal{D}$ satisfying Assumptions FEAT-INDEP and FEAT-EXP, if a data point $(\boldsymbol{x}, y)$ is randomly drawn from $\mathcal{D}$, and $\boldsymbol{x}$ is perturbed to $\boldsymbol{x}' = \boldsymbol{x} + \delta^*$, where δ^* is an $\ell_\infty(\varepsilon)$-adversarial perturbation, then under the $\ell_\infty(\varepsilon)$-adversarial loss $\mathcal{L}(\boldsymbol{x}', y; \boldsymbol{w})$, the expected SGD-update of weight w_i, namely $\overline{\Delta w_i}$, satisfies the following properties:*

1. *If $w_i = 0$, then*
$$\overline{\Delta w_i} = \overline{g'}\ a_i. \tag{10}$$
2. *If $w_i \neq 0$, then*
$$\overline{\Delta w_i} \leq \overline{g'}\ [a_i \operatorname{sgn}(w_i) - \varepsilon], \tag{11}$$
and equality holds in the limit as w_i approaches zero,

where $a_i = \mathbb{E}(x_i y)$ is the directed strength of feature x_i from Assumption FEAT-EXP, and $\overline{g'}$ is the expectation in (8).

For space reasons, a detailed discussion of the implications of this result is presented in Sec. D.2 of the Supplement. For the purposes of this paper, a key implication is that during SGD updates of $\ell_\infty(\varepsilon)$-adversarially perturbed examples, the weights of features that are weakly correlated (or un-correlated) to the label tend to be aggressively pushed

toward zero magnitude, and this aggressiveness is proportional to the difference between ε and the feature's label-correlation.

A generalization of the above result (Theorem 3.1) for the multi-class setting is presented in Section E (Theorem E.1) of the Supplement.

4. Feature Attribution using Integrated Gradients

Theorem 3.1 showed that $\ell_\infty(\varepsilon)$-adversarial training tends to shrink the *weights* of features that are "weak" (relative to ε). We now show a link between weights and *explanations*, specifically explanations in the form of a vector of feature-attributions given by the *Integrated Gradients* (IG) method (Sundararajan et al., 2017), which is defined as follows: Suppose $F : \mathbb{R}^d \to \mathbb{R}$ is a real-valued function of an input vector. For example F could represent the output of a neural network, or even a loss function $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ when the label y and weights $\boldsymbol{w}$ are held fixed. Let $\boldsymbol{x} \in \mathbb{R}^d$ be a specific input, and $\boldsymbol{u} \in \mathbb{R}^d$ be a baseline input. The IG is defined as the path integral of the gradients along the straight-line path from the baseline $\boldsymbol{u}$ to the input $\boldsymbol{x}$. The IG along the i'th dimension for an input $\boldsymbol{x}$ and baseline $\boldsymbol{u}$ is defined as:

$$\mathrm{IG}_i^F(\boldsymbol{x}, \boldsymbol{u}) := (x_i - u_i) \times \int_{\alpha=0}^{1} \partial_i F(\boldsymbol{u} + \alpha(\boldsymbol{x} - \boldsymbol{u}))d\alpha, \tag{12}$$

where $\partial_i F(\boldsymbol{z})$ denotes the gradient of $F(\boldsymbol{v})$ along the i'th dimension, at $\boldsymbol{v} = \boldsymbol{z}$. The vector of all IG components $\mathrm{IG}_i^F(\boldsymbol{x}, \boldsymbol{u})$ is denoted as $\mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u})$. Although we do not show $\boldsymbol{w}$ explicitly as an argument in the notation $\mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u})$, it should be understood that the IG depends on the model weights $\boldsymbol{w}$ since the function F depends on $\boldsymbol{w}$.

The following Lemma (proved in Sec. F of the Supplement) shows a closed form exact expression for the $\mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u})$ when $F(\boldsymbol{x})$ is of the form

$$F(\boldsymbol{x}) = A(\langle \boldsymbol{w}, \boldsymbol{x} \rangle), \tag{13}$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is a vector of weights, A is a differentiable scalar-valued function, and $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ denotes the dot product of $\boldsymbol{w}$ and x. Note that this form of F could represent a single-layer neural network with any differentiable activation function (e.g., logistic (sigmoid) activation $A(\boldsymbol{z}) = 1/[1 + \exp(-\boldsymbol{z})]$ or Poisson activation $A(\boldsymbol{z}) = \exp(\boldsymbol{z})$), or a differentiable loss function, such as those that satisfy Assumption LOSS-INC for a fixed label y and weight-vector $\boldsymbol{w}$. For brevity, we will refer to a function of the form (13) as representing a "1-Layer Network", with the understanding that it could equally well represent a suitable loss function.

Lemma 4.1 (IG Attribution for 1-layer Networks)**.** *If $F(\boldsymbol{x})$ is computed by a 1-layer network (13) with weights vector $\boldsymbol{w}$, then the Integrated Gradients for all dimensions of $\boldsymbol{x}$ relative to a baseline $\boldsymbol{u}$ are given by:*

$$\mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u}) = [F(\boldsymbol{x}) - F(\boldsymbol{u})] \frac{(\boldsymbol{x} - \boldsymbol{u}) \odot \boldsymbol{w}}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle}, \tag{14}$$

where the $\odot$ operator denotes the entry-wise product of vectors.

Thus for 1-layer networks, the IG of each feature is essentially proportional to the feature's fractional contribution to the logit-change $\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle$. This makes it clear that in such models, if the weight-vector $\boldsymbol{w}$ is sparse, then the IG vector will also be correspondingly sparse.

5. Training with Explanation Stability is equivalent to Adversarial Training

Suppose we use the IG method described in Sec. 4 as an explanation for the output of a model $F(\boldsymbol{x})$ on a specific input $\boldsymbol{x}$. A desirable property of an explainable model is that the explanation for the value of $F(\boldsymbol{x})$ is *stable*(Alvarez-Melis & Jaakkola, 2018), i.e., does not change much under small perturbations of the input $\boldsymbol{x}$. One way to formalize this is to say the following *worst-case* ℓ_1-norm of the change in IG should be small:

$$\max_{\boldsymbol{x}' \in N(\boldsymbol{x}, \varepsilon)} ||\, \mathrm{IG}^F(\boldsymbol{x}', \boldsymbol{u}) - \mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u})||_1, \tag{15}$$

where $N(\boldsymbol{x}, \varepsilon)$ denotes a suitable ε-neighborhood of $\boldsymbol{x}$, and $\boldsymbol{u}$ is an appropriate baseline input vector. If the model F is a single-layer neural network, it would be a function of $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ for some weights $\boldsymbol{w}$, and typically when training such networks the loss is a function of $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ as well, so we would not change the essence of (15) much if instead of F in each IG, we use $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ for a fixed y; let us denote this function by $\mathcal{L}_y$. Also intuitively, $||\, \mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}', \boldsymbol{u}) - \mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}, \boldsymbol{u})||_1$ is not too different from $||\, \mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}', \boldsymbol{x})||_1$. These observations motivate the following definition of *Stable-IG Empirical Risk*, which is a modification of the usual empirical risk (1), with a regularizer to encourage stable IG explanations:

$$\mathbb{E}_{(\boldsymbol{x}, y) \sim \mathcal{D}} \Big[\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) + \max_{||\, \boldsymbol{x}' - \boldsymbol{x}\, ||_\infty \le \varepsilon} ||\, \mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}, \boldsymbol{x}')||_1 \Big]. \tag{16}$$

The following somewhat surprising result is proved in Section G of the Supplement.

Theorem 5.1 (Equivalence of Stable IG and Adversarial Robustness)**.** *For loss functions $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ satisfying Assumption LOSS-CVX, the augmented loss inside the expectation (16) equals the $\ell_\infty(\varepsilon)$-adversarial loss inside the*

expectation (2), i.e.

$$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) \; + \max_{||\boldsymbol{x}' - \boldsymbol{x}||_\infty \leq \varepsilon} ||\,\mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}, \boldsymbol{x}')||_1 \;\; = \max_{||\delta||_\infty \leq \varepsilon} \mathcal{L}(\boldsymbol{x} + \delta, y; \boldsymbol{w}) \quad (17)$$

This implies that for loss functions satisfying Assumption LOSS-CVX, minimizing the Stable-IG Empirical Risk (16) is equivalent to minimizing the expected $\ell_\infty(\varepsilon)$-adversarial loss. In other words, for this class of loss functions, *natural model training while encouraging IG stability is equivalent to $\ell_\infty(\varepsilon)$-adversarial training!* Combined with Theorem 3.1 and the corresponding experimental results in Sec 6, this equivalence implies that, for this class of loss functions, and data distributions satisfying Assumption FEAT-INDEP, the explanations for the models produced by $\ell_\infty(\varepsilon)$-adversarial training are both *concise* (due to the sparseness of the models), and *stable*.

6. Experiments

6.1. Hypotheses

Recall that one implication of Theorem 3.1 is the following: For 1-layer networks where the loss function satisfies Assumption LOSS-CVX, $\ell_\infty(\varepsilon)$-adversarial training tends to more-aggressively prune the *weight-magnitudes* of "weak" features compared to natural training. In Sec. 4 we observed that a consequence of Lemma 4.1 is that for 1-layer models the sparseness of the weight vector implies sparseness of the IG vector. Thus a reasonable conjecture is that, for 1-layer networks, $\ell_\infty(\varepsilon)$-adversarial training leads to models with sparse attribution vectors in general (whether using IG or a different method, such as DeepSHAP). We further conjecture that this sparsification phenomenon extends to practical multi-layer Deep Neural Networks, not just 1-layer networks, and that this benefit can be realized without significantly impacting accuracy on natural test data. Finally, we hypothesize that the resulting sparseness of *attribution vectors* is better than what can be achieved by a traditional *weight regularization* technique such as L1-regularization, for a comparable level of natural test accuracy.

6.2. Measuring Sparseness of an Attribution Vector

For an attribution method A, we quantify the sparseness of the attribution vector $A^F(\boldsymbol{x}, \boldsymbol{u})$ using the *Gini Index* applied to the vector of absolute values $A^F(\boldsymbol{x}, \boldsymbol{u})$. For a vector $\boldsymbol{v}$ of non-negative values, the Gini Index, denoted $G(\boldsymbol{v})$ (defined formally in Sec. I in the Supplement), is a metric for sparseness of $\boldsymbol{v}$ that is known (Hurley & Rickard, 2009) to satisfy a number of desirable properties, and has been used to quantify sparseness of weights in a neural network (Guest & Love, 2017). The Gini Index by definition lies in [0,1], and a higher value indicates more sparseness.

Since the model F is clear from the context, and the baseline vector $\boldsymbol{u}$ are fixed for a given dataset, we will denote the attribution vector on input $\boldsymbol{x}$ simply as $A(\boldsymbol{x})$, and our measure of sparseness is $G(|A(\boldsymbol{x})|)$, which we denote for brevity as $G[A](\boldsymbol{x})$, and refer to informally as the *Gini of* A, where A can stand for IG (when using IG-based attributions) or SH (when using DeepSHAP for attribution). As mentioned above, one of our hypotheses is that the sparseness of attributions of models produced by $\ell_\infty(\varepsilon)$-adversarial training is much better than what can be achieved by natural training using ℓ_1-regularization, for a comparable level of accuracy. To verify this hypothesis we will compare the sparseness of attribution vectors resulting from three types of models: (a) n-model: *naturally-trained* model with no adversarial perturbations and no ℓ_1-regularization, (b) a-model: $\ell_\infty(\varepsilon)$-*adversarially trained* model, and (c) l-model: naturally trained model with ℓ_1-*regularization* strength $\lambda > 0$. For an attribution method A, we denote the Gini indices $G[A](\boldsymbol{x})$ resulting from these models respectively as $G^n[A](\boldsymbol{x})$, $G^a[A](\boldsymbol{x};\ \varepsilon)$ and $G^l[A](\boldsymbol{x};\ \lambda)$.

In several of our datasets, individual feature vectors are already quite sparse: for example in the MNIST dataset, most of the area consists of black pixels, and in the Mushroom dataset, after 1-hot encoding the 22 categorical features, the resulting 120-dimensional feature-vector is sparse. On such datasets, even an n-model can achieve a "good" level of sparseness of attributions in the absolute sense, i.e. $G^n[A](\boldsymbol{x})$ can be quite high. Therefore for all datasets we compare the *sparseness improvement* resulting from an a-model relative to an n-model, with that from an l-model relative to a n-model. Or more precisely, we will compare the two quantities defined below, for a given attribution method A:

$$dG^a[A](\boldsymbol{x};\ \varepsilon) := G^a[A](\boldsymbol{x};\ \varepsilon) - G^n[A](\boldsymbol{x}), \quad (18)$$

$$dG^l[A](\boldsymbol{x};\ \lambda) := G^l[A](\boldsymbol{x};\ \lambda) - G^n[A](\boldsymbol{x}). \quad (19)$$

The above quantities define the IG sparseness improvements for a *single* example $\boldsymbol{x}$. It will be convenient to define the overall sparseness improvement from a model, as measured on a test dataset, by averaging over all examples $\boldsymbol{x}$ in that dataset. We denote the corresponding *average* sparseness metrics by $G^a[A](\varepsilon)$, $G^l[A](\lambda)$ and $G^n[A]$ respectively. We then define the *average sparseness improvement* of an a-model and l-model as:

$$dG^a[A](\varepsilon) := G^a[A](\varepsilon) - G^n[A], \quad (20)$$

$$dG^l[A](\lambda) := G^l[A](\lambda) - G^n[A]. \quad (21)$$

We can thus re-state our hypotheses in terms of this notation: For each of the attribution methods $A \in \{IG, SH\}$, the average sparseness improvement $dG^a[A](\varepsilon)$ resulting from type-a models is high, and is significantly higher than the average sparseness improvement $dG^l[A](\lambda)$ resulting from type-l models.

6.3. Results

We ran experiments on four standard public benchmark datasets: two image datasets MNIST and Fashion-MNIST, and two tabular datasets from the UCI Data Repository: Mushroom and Spambase. Details of the datasets and training methodology are in Sec. J.1 of the Supplement.

For each of the two tabular datasets (where we train logistic regression models), for a given model-type (a, l or n), we found the average Gini index of the attribution vectors is virtually identical when using IG or DeepSHAP. This is not surprising: as pointed out in (Ancona et al., 2017), DeepSHAP is a variant of DeepLIFT, and for simple linear models, DeepLIFT gives a very close approximation of IG. To avoid clutter, we therefore omit DeepSHAP-based results on the tabular datasets. Table 1 shows a summary of some results on the above 4 datasets, and Fig. 2 and 3 display results graphically.

Table 1: Results on 4 datasets. For each dataset, "a" indicates an $\ell_\infty(\varepsilon)$-adversarially trained model with the indicated ε, and "l" indicates a naturally trained model with the indicated ℓ_1-regularization strength λ. The **attr** column indicates the feature attribution method (IG or DeepSHAP). Column **dG** shows the average sparseness improvements of the models relative to the baseline naturally trained model, as measured by the $dG^a[A](\varepsilon)$ and $dG^l[A](\lambda)$ defined in Eqs. (20, 21). Column **AcDrop** indicates the drop in accuracy relative to the baseline model.

dataset	attr	model	dG	AcDrop
MNIST	IG	a ($\varepsilon = 0.3$)	0.06	0.8%
	IG	l ($\lambda = 0.01$)	0.004	0.4%
	SHAP	a ($\varepsilon = 0.3$)	0.06	0.8%
	SHAP	l ($\lambda = 0.01$)	0.007	0.4%
Fashion-MNIST	IG	a ($\varepsilon = 0.1$)	0.06	4.7%
	IG	l ($\lambda = 0.01$)	0.008	3.4%
	SHAP	a ($\varepsilon = 0.1$)	0.08	4.7%
	SHAP	l ($\lambda = 0.01$)	0.003	3.4%
Mushroom	IG	a ($\varepsilon = 0.1$)	0.06	2.5%
	IG	l ($\lambda = 0.02$)	0.06	2.6%
Spambase	IG	a ($\varepsilon = 0.1$)	0.17	0.9%
	IG	l ($\lambda = 0.02$)	0.15	0.1%

These results make it clear that for comparable levels of accuracy, the sparseness of attribution vectors from $\ell_\infty(\varepsilon)$-adversarially trained models is much better than the sparseness from natural training with ℓ_1-regularization. The effect is especially pronounced in the two image datasets. The effect is less dramatic in the two tabular datasets, for which we train logistic regression models. Our discussion at the end of Sec. 4 suggests a possible explanation.

In the Introduction we gave an example of a *saliency map* (Simonyan et al., 2013; Baehrens et al., 2010) (Fig. 1) to dramatically highlight the sparseness induced by adversarial training. We show several more examples of saliency maps in the supplement (Section J.4).

7. Related Work

In contrast to the growing body of work on defenses against adversarial attacks (Yuan et al., 2017; Madry et al., 2017; Biggio et al., 2013) or explaining adversarial examples (Goodfellow et al., 2014; Tsipras et al., 2018), the focus of our paper is the connection between adversarial robustness and explainability. We view the process of adversarial training as a tool to produce more explainable models. A recent series of papers (Tsipras et al., 2018; Ilyas et al., 2019) essentially argues that adversarial examples exist because standard training produces models are heavily reliant on highly predictive but *non-robust features* (which is similar to our notion of "weak" features in Sec 3) which are vulnerable to an adversary who can "flip" them and cause performance to degrade. Indeed the authors of (Ilyas et al., 2019) touch upon some connections between explainability and robustness, and conclude, "*As such, producing human-meaningful explanations that remain faithful to underlying models cannot be pursued independently from the training of the models themselves*", by which they are implying that good explainability may require intervening in the model-training procedure itself; this is consistent with our findings. Another recent related paper (Kim et al., 2019) analyzes the effect of adversarial training on the interpretability of neural network loss gradients. We discuss other related work in the Supplement Section A.

8. Conclusion

We presented theoretical and experimental results that show a strong connection between adversarial robustness (under ℓ_∞-bounded perturbations) and two desirable properties of model explanations: conciseness and stability. Specifically, we considered model explanations in the form of feature-attributions based on the Integrated Gradients (IG) and DeepSHAP techniques. For 1-layer models using a popular family of loss functions, we theoretically showed that $\ell_\infty(\varepsilon)$-adversarial training tends to produce *sparse* and *stable* IG-based attribution vectors. With extensive experiments on benchmark tabular and image datasets, we demonstrated that the "attribution sparsification" effect extends to Deep Neural Networks, when using two popular attribution methods. Intriguingly, especially in DNN models for image classification, the attribution sparseness from natural training with ℓ_1-regularization is much inferior to that achievable via $\ell_\infty(\varepsilon)$-adversarial training. Our theoretical results are a first step in explaining some of these phenomena.

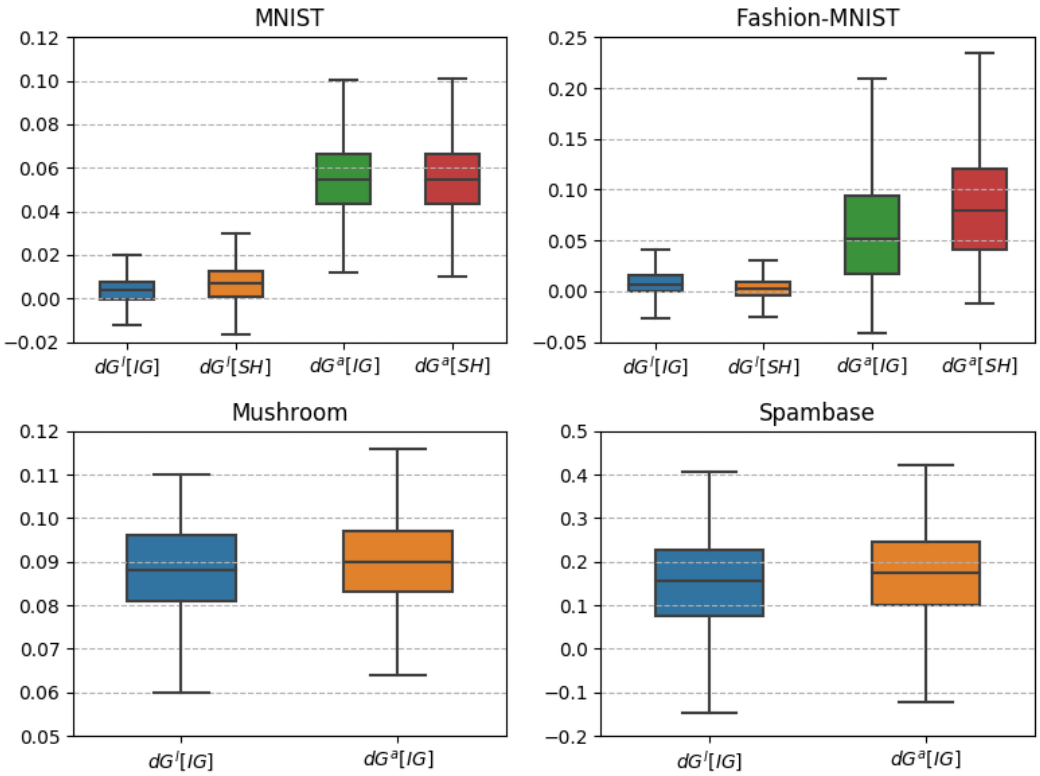


Figure 2: Boxplot of pointwise sparseness-improvements from adversarially trained models ($dG^a[A](\boldsymbol{x}, \varepsilon)$) and naturally trained models with ℓ_1-regularization ($dG^l[A](\boldsymbol{x}, \lambda)$), for attribution methods $A \in \{IG, SH\}$.

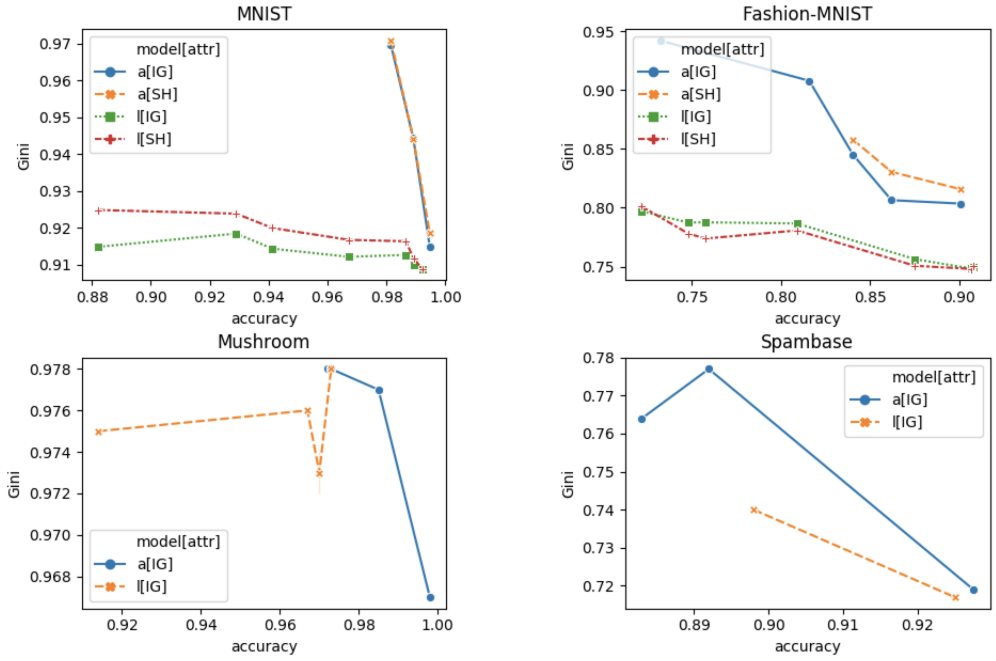


Figure 3: For each dataset, for each combination of model-type (l or a) and attribution method (IG or DeepSHAP), the Average Gini Index ($G^a[A](\varepsilon)$ and $G^l[A](\lambda)$) vs Accuracy achievable by choosing different values of ε or λ depending on the model type.

References

Abramovich, F. and Grinshtein, V. High-dimensional classification by sparse logistic regression. June 2017. URL http://arxiv.org/abs/1706.08344.

Alvarez-Melis, D. and Jaakkola, T. S. On the robustness of interpretability methods. *arXiv preprint arXiv:1806. 08049*, 2018. URL http://arxiv.org/abs/1806.08049.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. November 2017. URL http://arxiv.org/abs/1711.06104.

Arya, V., Bellamy, R. K. E., Chen, P.-Y., Dhurandhar, A., Hind, M., Hoffman, S. C., Houde, S., Vera Liao, Q., Luss, R., Mojsilović, A., Mourad, S., Pedemonte, P., Raghavendra, R., Richards, J., Sattigeri, P., Shanmugam, K., Singh, M., Varshney, K. R., Wei, D., and Zhang, Y. One explanation does not fit all: A toolkit and taxonomy of AI explainability techniques. September 2019. URL http://arxiv.org/abs/1909.03012.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. How to explain individual classification decisions. *J. Mach. Learn. Res.*, 11(Jun):1803–1831, 2010. URL http://www.jmlr.org/papers/volume11/baehrens10a/baehrens10a.pdf.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases*, pp. 387–402. Springer Berlin Heidelberg, 2013. URL http://dx.doi.org/10.1007/978-3-642-40994-3_25.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. December 2014. URL http://arxiv.org/abs/1412.6572.

Guest, O. and Love, B. C. What the success of brain imaging implies about the neural code. *Elife*, 6, January 2017. URL http://dx.doi.org/10.7554/eLife.21397.

Hurley, N. and Rickard, S. Comparing measures of sparsity. *IEEE Trans. Inf. Theory*, 55(10):4723–4741, October 2009. URL http://dx.doi.org/10.1109/TIT.2009.2027527.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. May 2019. URL http://arxiv.org/abs/1905.02175.

Kim, B., Seo, J., and Jeon, T. Bridging adversarial robustness and gradient interpretability. *Safe Machine Learning workshop at ICLR*, 2019.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pp. 4765–4774, 2017.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. June 2017.

Molnar, C. *Interpretable Machine Learning*. 2019. https://christophm.github.io/interpretable-ml-book/.

Noack, A., Ahern, I., Dou, D., and Li, B. Does interpretability of neural networks imply adversarial robustness? *ArXiv*, abs/1912.03430, 2019.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Berkay Celik, Z., and Swami, A. The limitations of deep learning in adversarial settings. November 2015. URL http://arxiv.org/abs/1511.07528.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should I trust you?": Explaining the predictions of any classifier. February 2016. URL http://arxiv.org/abs/1602.04938.

Sankaranarayanan, S., Jain, A., Chellappa, R., and Lim, S. N. Regularizing deep networks using efficient layerwise adversarial training. May 2017. URL http://arxiv.org/abs/1705.07819.

Shaham, U., Yamada, Y., and Negahban, S. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. November 2015. URL http://arxiv.org/abs/1511.05432.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

Sinha, A., Namkoong, H., and Duchi, J. Certifying some distributional robustness with principled adversarial training. February 2018. URL https://openreview.net/pdf?id=Hk6kPgZA-.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. March 2017.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. December 2013. URL http://arxiv.org/abs/1312.6199.

Tan, M., Tsang, I. W., and Wang, L. Minimax sparse logistic regression for very high-dimensional feature selection. *IEEE Trans Neural Netw Learn Syst*, 24(10):1609–1622, October 2013. URL http://dx.doi.org/10.1109/TNNLS.2013.2263427.

Tan, M., Tsang, I. W., and Wang, L. Towards ultrahigh dimensional feature selection for big data. *J. Mach. Learn. Res.*, 15(1):1371–1429, 2014. URL http://www.jmlr.org/papers/volume15/tan14a/tan14a.pdf.

Tanay, T. and Griffin, L. D. A new angle on l2 regularization. *CoRR*, abs/1806.11186, 2018.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. May 2018. URL http://arxiv.org/abs/1805.12152.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. August 2017. URL http://arxiv.org/abs/1708.07747.

Xu, H., Caramanis, C., and Mannor, S. Robustness and regularization of support vector machines. *J. Mach. Learn. Res.*, 10(Jul):1485–1510, 2009. URL http://www.jmlr.org/papers/volume10/xu09b/xu09b.pdf.

Yeh, C.-K., Hsieh, C.-Y., Suggala, A., Inouye, D. I., and Ravikumar, P. K. On the (in)fidelity and sensitivity of explanations. In Wallach, H., Larochelle, H., Beygelzimer, A., dAlché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 10967–10978. Curran Associates, Inc., 2019. URL https://arxiv.org/pdf/1901.09392.pdf.

Yuan, X., He, P., Zhu, Q., and Li, X. Adversarial examples: Attacks and defenses for deep learning. December 2017. URL http://arxiv.org/abs/1712.07107.

[**Code** for the experiments is available at `https://github.com/jfc43/advex`.]

A. Additional Related Work

Section 7 discussed some of the work most directly related to this paper. Here we describe some additional related work.

Adversarial Robustness and Interpretability. Through a very different analysis, (Yeh et al., 2019) show a result closely related to our Theorem 5.1: they show that adversarial training is analogous to making gradient-based explanations more "smooth", which lowers the sensitivity of gradient explanation. The paper of (Noack et al., 2019) considers a question that is the converse of the one we examine in our paper: They show evidence that models that are forced to have interpretable gradients are more robust to adversarial examples than models trained in a standard manner.

Relation to work on Regularization Benefits of AML. There has been prior work on the *regularization benefits* of adversarial training (Xu et al., 2009; Szegedy et al., 2013; Goodfellow et al., 2014; Shaham et al., 2015; Sankaranarayanan et al., 2017; Tanay & Griffin, 2018), primarily in image-classification applications: when a model is adversarially trained, its classification accuracy on natural (i.e. un-perturbed) test data can improve. All of this prior work has focused on the *performance-improvement* (on natural test data) aspect of regularization, but none have examined the *feature-pruning* benefits explicitly. In contrast to this work, our primary interest is in the explainability benefits of adversarial training, and specifically the ability of adversarial training to significantly improve feature-concentration while maintaining (and often improving) performance on natural test data.

Adversarial Training vs Feature-Selection. Since our results show that adversarial training can effectively shrink the weights of irrelevant or weakly-relevant features (while preserving weights on relevant features), a legitimate counter-proposal might be that one could weed out such features beforehand via a pre-processing step where features with negligible label-correlations can be "removed" from the training process. Besides the fact that this scheme has no guarantees whatsoever with regard to adversarial robustness, there are some practical reasons why correlation-based feature selection is not as effective as adversarial training, in producing pruned models: (a) With adversarial training, one needs to simply try different values of the adversarial strength parameter ε and find a level where accuracy (or other metric such as AUC-ROC) is not impacted much but model-weights are significantly more concentrated; on the other hand with the correlation-based feature-pruning method, one needs to set up an iterative loop with gradually increasing correlation thresholds, and each time the input pre-processing pipeline needs to be re-executed with a reduced set of features. (b) When there are categorical features with large cardinalities, where just some of the categorical values have negligible feature-correlations, it is not even clear how one can "remove" these specific feature *values*, since the feature itself must still be used; at the very least it would require a re-encoding of the categorical feature each time a subset of its values is "dropped" (for example if a one-hot encoding or hashing scheme is used). Thus correlation-based feature-pruning is a much more cumbersome and inefficient process compared to adversarial training.

Adversarial Training vs Other Methods to Train Sparse Logistic Regression Models. (Tan et al., 2013; 2014) propose an approach to train sparse logistic regression models based on a min-max optimization problem that can be solved by the cutting plane algorithm. This requires a specially implemented custom optimization procedure. By contrast, $\ell_\infty(\varepsilon)$-adversarial training can be implemented as a simple and efficient "bolt-on" layer on top of existing ML pipelines based on TensorFlow, PyTorch or SciKit-Learn, which makes it highly practical. Another paper (Abramovich & Grinshtein, 2017) proposes a feature selection procedure based on penalized maximum likelihood with a complexity penalty on the model size, but once again this requires special-purpose optimization code.

B. Discussion of Assumptions

B.1. Loss Functions Satisfying Assumption LOSS-CVX

We show here that several popular loss functions satisfy the Assumption LOSS-CVX.

Logistic NLL (Negative Log Likelihood) Loss.
$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = -\ln(\sigma(y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)) = \ln(1+\exp(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle))$, which can be written as $g(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ where $g(z) = \ln(1+e^z)$ is a non-decreasing and convex function.

Hinge Loss
$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = (1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)^+$, which can be written as $g(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ where $g(z) = (1+z)^+$ is non-decreasing and convex.

Softplus Hinge Loss.
$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = \ln(1 + \exp(1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle))$, which can be written as $g(-y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ where $g(z) = \ln(1 + e^{1+z})$, and clearly g is non-decreasing. Moreover the first derivative of g, $g'(z) = 1/(1 + e^{-1-z})$ is non-decreasing, and therefore g is convex.

B.2. PropFEAT-EXP is without loss of generality

While Assumption FEAT-EXP may seem restrictive, we show that there is in fact no loss of generality in making this assumption, when there are only two possible values of the label.

Lemma 1 (Form of conditional expectation of X given Y when $Y \in \{\pm 1\}$)**.** *Given random variables X, Y where $Y \in \{\pm 1\}$, $\mathbb{E}(X|Y)$ must be of the form:*

$$\mathbb{E}(X|Y) = aY + c \tag{B.22}$$

where

$$a = [\mathbb{E}(X|Y=1) - \mathbb{E}(X|Y=-1)]/2 \tag{B.23}$$

and

$$c = [\mathbb{E}(X|Y=1) + \mathbb{E}(X|Y=-1)]/2 \tag{B.24}$$

This implies that the random variable $X' = X - c$ satisfies $\mathbb{E}(X'|Y) = aY$.

Proof. Consider the function $f(Y) = \mathbb{E}(X|Y)$, and let $b_0 := f(-1)$ and $b_1 := f(1)$. Since there are only *two* values of Y that are of interest, we can represent $f(Y)$ by a *linear* function $aY + c$, and it is trivial to verify that $a = (b_1 - b_0)/2$ and $c = (b_1 + b_0)/2$ are the unique values that are consistent with $f(-1) = b_0$ and $f(1) = b_1$. □

A consequence of this Lemma is that a dataset can be transformed into one satisfying Assumption FEAT-EXP by a simple translation of each x_i to a new feature

$$x'_i = x_i - 0.5 * [\mathbb{E}(x_i|y=1) + \mathbb{E}(x_i|y=-1)]. \tag{B.25}$$

C. Expressions for adversarial perturbation and loss-gradient

We show two simple preliminary results for loss functions that satisfy Assumption LOSS-INC: Lemma 2 shows a simple closed form expression for the $\ell_\infty(\varepsilon)$-adversarial perturbation, and we use this result to derive an expression for the *gradient* of the $\ell_\infty(\varepsilon)$-adversarial loss $\mathcal{L}(\boldsymbol{x} + \delta^*, y; \boldsymbol{w})$ with respect to a weight w_i (Lemma 3).

Lemma 2 (Closed form for $\ell_\infty(\varepsilon)$-adversarial perturbation)**.** *For a data point $(\boldsymbol{x}, y)$, given model weights $\boldsymbol{w}$, if the loss function $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ satisfies Assumption LOSS-INC, the $\ell_\infty(\varepsilon)$-adversarial perturbation δ^* is given by:*

$$\delta^* = -y \operatorname{sgn}(\boldsymbol{w})\varepsilon, \tag{5}$$

and the corresponding $\ell_\infty(\varepsilon)$-adversarial loss is

$$\mathcal{L}(\boldsymbol{x} + \delta^*, y;\ \boldsymbol{w}) = g(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle) \tag{6}$$

Proof. Assumption LOSS-INC implies that the loss is non-increasing in $y\langle \boldsymbol{w}, x\rangle$, and therefore the $\ell_\infty(\varepsilon)$-perturbation δ^* of x that maximizes the loss would be such that, for each $i \in [d]$, x_i is changed by an amount ε in the direction of $-y \operatorname{sgn}(w_i)$, and the result immediately follows. □

Lemma 3 (Gradient of adversarial loss)**.** *For any loss function satisfying Assumption LOSS-INC, for a given data point $(\boldsymbol{x}, y)$, the gradient of the $\ell_\infty(\varepsilon)$-adversarial loss is given by:*

$$\frac{\partial \mathcal{L}(\boldsymbol{x} + \delta^*, y;\ \boldsymbol{w})}{\partial w_i} = -g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ (yx_i - \operatorname{sgn}(w_i)\varepsilon) \tag{7}$$

Proof. This is straightforward by substituting the expression (5) for δ^* in $g(-y\langle \boldsymbol{w}, \boldsymbol{x} + \delta^*\rangle)$, and applying the chain rule. □

D. Expectation of SGD Weight Update

The following Lemma will be used to prove Theorem 3.1.

D.1. Upper bound on $\mathbb{E}[Zf(Z,V)]$

Lemma 4 (Upper Bound on expectation of $Zf(Z,V)$ when f is non-increasing in Z, $(Z \perp V)|Y$, and $\mathbb{E}(Z|Y) = \mathbb{E}(Z)$)**.** *For any random variables Z, V, if:*

- *$f(Z,V)$ is non-increasing in Z,*
- *Z, V are conditionally independent given a third r.v. Y, and*
- *$\mathbb{E}(Z|Y) = \mathbb{E}(Z)$,*

then

$$\mathbb{E}[Zf(Z,V)] \;\leq\; \mathbb{E}(Z)\mathbb{E}[f(Z,V)] \tag{D.26}$$

Proof. Let $\overline{z} = \mathbb{E}(Z) = \mathbb{E}(Z|Y)$ and note that

$$\begin{aligned}\mathbb{E}[Zf(Z,V)] - \mathbb{E}[Z]\mathbb{E}[f(Z,V)] &= \mathbb{E}[Zf(Z,V)] - \overline{z}\mathbb{E}[f(Z,V)] &\text{(D.27)}\\ &= \mathbb{E}[(Z-\overline{z})f(Z,V)] &\text{(D.28)}\end{aligned}$$

We want to now argue that $\mathbb{E}[(Z-\overline{z})f(\overline{z},V)] = 0$. To see this, apply the Law of Total Expectation by conditioning on Y:

$$\begin{aligned}\mathbb{E}[(Z-\overline{z})f(\overline{z},V)] &= \mathbb{E}\Big[\mathbb{E}\big[(Z-\overline{z})f(\overline{z},V)|Y\big]\Big]\\ &= \mathbb{E}\Big[\mathbb{E}\big[(Z-\overline{z})|Y\big]\mathbb{E}\big[f(\overline{z},V)|Y\big]\Big] && \text{(since } (Z\perp V)|Y\text{)} &\text{(D.29)}\\ &= 0. && \text{(since } \mathbb{E}(Z|Y)=\mathbb{E}(Z)=\overline{z}\text{)} &\text{(D.30)}\end{aligned}$$

Since $\mathbb{E}[(Z-\overline{z})f(\overline{z},V)] = 0$, we can subtract it from the last expectation in (D.28), and by linearity of expectations the RHS of (D.28) can be replaced by

$$\mathbb{E}\big[(Z-\overline{z})(f(Z,V) - f(\overline{z},V))\big]. \tag{D.31}$$

That fact that $f(Z,V)$ is non-increasing in Z implies that $(Z-\overline{z})(f(Z,V)-f(\overline{z},V)) \leq 0$ for any value of Z and V, with equality when $Z = \overline{z}$. Therefore the expectation (D.31) is bounded above by zero, which implies the desired result. □

Theorem 3.1 (Expected SGD Update in Adversarial Training)**.** *For any loss function $\mathcal{L}$ satisfying Assumption LOSS-CVX and a data distribution $\mathcal{D}$ satisfying Assumptions FEAT-INDEP and FEAT-EXP, if a data point $(\boldsymbol{x}, y)$ is randomly drawn from $\mathcal{D}$, and $\boldsymbol{x}$ is perturbed to $\boldsymbol{x}' = \boldsymbol{x} + \delta^*$, where δ^* is an $\ell_\infty(\varepsilon)$-adversarial perturbation, then under the $\ell_\infty(\varepsilon)$-adversarial loss $\mathcal{L}(\boldsymbol{x}', y; \boldsymbol{w})$, the expected SGD-update of weight w_i, namely $\overline{\Delta w_i}$, satisfies the following properties:*

1. *If $w_i = 0$, then*
$$\overline{\Delta w_i} = \overline{g'}\; a_i. \tag{10}$$
2. *If $w_i \neq 0$, then*
$$\overline{\Delta w_i} \leq \overline{g'}\;[a_i \operatorname{sgn}(w_i) - \varepsilon], \tag{11}$$
and equality holds in the limit as w_i approaches zero,

where $a_i = \mathbb{E}(x_i y)$ is the directed strength of feature x_i from Assumption FEAT-EXP, and $\overline{g'}$ is the expectation in (8).

Proof. Consider the adversarial loss gradient expression (7) from Lemma 3. For the case $w_i = 0$, the negative expectation of the adversarial loss gradient is

$$\begin{aligned}\overline{\Delta w_i} &= \mathbb{E}\big[yx_i\, g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\big] \\ &= \mathbb{E}\Big[\mathbb{E}\big[yx_i\, g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ |y\big]\Big] \\ &= \mathbb{E}\Big[y\,\mathbb{E}\big[x_i\, g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ |y\big]\Big],\end{aligned}$$

and in the last expectation above, we note that since $w_i = 0$ the argument of g' does not depend on x_i, and by Assumption FEAT-INDEP the features are conditionally independent given y, so the inner conditional expectation can be factored as a product of conditional expectations, which gives

$$\begin{aligned}\overline{\Delta w_i} &= \mathbb{E}\Big[y\mathbb{E}(x_i|y)\mathbb{E}\big[g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ |y\big]\Big] \\ &= \mathbb{E}\Big[y^2 a_i \mathbb{E}\big[g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ |y\big]\Big] && \text{(due to Assumption FEAT-EXP)} \\ &= a_i \mathbb{E}\Big[\mathbb{E}\big[g'(\varepsilon||\boldsymbol{w}||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\ |y\big]\Big] && \text{(since } y = \pm 1\text{)} \\ &= a_i \overline{g'}, && \text{(D.32)}\end{aligned}$$

which establishes the first result.

Now consider the case $w_i \neq 0$. Starting with the adversarial loss gradient expression (7) from Lemma 3, multiplying throughout by $-\operatorname{sgn}(w_i)$ and taking expectations results in

$$\overline{\Delta w_i} = \mathbb{E}\Big[\big[yx_i \operatorname{sgn}(w_i) - \varepsilon\big]\ g'(\varepsilon||w||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\Big] \tag{D.33}$$

where the expectation is with respect to a random choice of data-point $(\boldsymbol{x}, y)$. The argument of g' can be written as

$$\varepsilon||w||_1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = -\sum_{j=1}^{d} |w_j|(yx_j \operatorname{sgn}(w_j) - \varepsilon).$$

For $j \in \{1, 2, \ldots, d\}$ if we let Z_j denote the random variable corresponding to $yx_j sgn(w_j) - \varepsilon$, then the expectation (D.34) can be written as

$$\mathbb{E}\left[Z_i\, g'\left(-\sum_{j=1}^{d} |w_j| Z_j\right)\right] = \mathbb{E}\left[Z_i\, g'\left(V - |w_i| Z_i\right)\right], \tag{D.34}$$

where the random variable V denotes the negative sum of the $|w_j|Z_j$ terms over all j *except* $j = i$. Note that the last expectation above is with respect to the distribution of three random variables: Z_i, V, and the random variable Y corresponding to the label y of the data point. Since Z_i is a function of feature x_i and Y, and V is a function of the remaining features and Y, Assumption FEAT-INDEP (the features x_j are conditionally independent given the label Y) implies $(V \perp Z_i)|Y$. Moreover (3) and (4) imply that

$$\mathbb{E}(Z_i) = \mathbb{E}(Z_i|Y) = a_i \operatorname{sgn}(w_i) - \varepsilon. \tag{D.35}$$

Since by Assumption LOSS-CVX, g' is a non-decreasing function, $g'(V - |w_i|Z_i)$ is *non-increasing* in Z_i. Thus all three conditions of Lemma 4 are satisfied, with the random variables Z, V, Y and function f in the Lemma corresponding to random variables Z_i, V, Y and function g' respectively in the present Theorem. It then follows from Lemma 4 that

$$\begin{aligned}\overline{\Delta w_i} &\leq E(Z_i)\overline{g'} && \text{(D.36)} \\ &= \overline{g'}[a_i \operatorname{sgn}(w_i) - \varepsilon], && \text{(D.37)}\end{aligned}$$

which establishes the upper bound (11) for any $w_i \neq 0$. Now consider a $w_i \neq 0$ that is infinitesimally close to 0. In this case $\overline{\Delta w_i}$ equals the RHS of (D.34), and if we let $|w_i|$ approach zero,

$$\overline{\Delta w_i} = \mathbb{E}[Z_i g'(V)] = \mathbb{E}\big[\mathbb{E}[Z_i g'(V)|Y]\big] = \mathbb{E}\big[\mathbb{E}[Z_i|Y]\mathbb{E}[g'(V)|Y]\big], \tag{D.38}$$

where the last equality follows from the conditional independence of Z_i and V given Y. Using (D.35) the last expectation above simplifies to $(a_i \operatorname{sgn}(w_i) - \varepsilon)\overline{g'}$, which establishes the equality result for non-zero infinitesimally small w_i. □

D.2. Implications of Theorem 3.1

Note that $a_j \operatorname{sgn}(w_j) > 0$ signifies that the sign of the weight w_j agrees with the *directed strength* a_j of feature x_j, and for brevity we simply say that *weight* w_j *is aligned*. Conversely when $a_j \operatorname{sgn}(w_j) < 0$ we say that *weight* w_j *is mis-aligned*. With these observations in mind Theorem 3.1 implies the following:

Weights grow from zero in the correct direction. When $w_i = 0$, the expected SGD update $\overline{\Delta w_i}$ is proportional to the directed strength a_i of feature x_i, and if $\overline{g'} \neq 0$, this means that on average the SGD update causes the weight w_i to grow from zero in the *correct direction*. This is what one would expect from an SGD training procedure.

Mis-aligned weights w_i shrink at a rate proportional to $\varepsilon + |a_i|$. When $w_i \neq 0$, if w_i is *mis-aligned*, i.e. $a_i \operatorname{sgn}(w_i) < 0$, the upper bound in (11) is non-positive, and this means the expected SGD update is non-positive, and the weight w_i *shrinks* on average. Thus, mis-aligned weights shrink on average, at a rate proportional to the ℓ_∞ adversarial bound ε plus the absolute feature strength. In other words, all other factors remaining the same, adversarial training (i.e. with $\varepsilon > 0$) shrinks mis-aligned faster than natural training (i.e. with $\varepsilon = 0$).

If w_i is aligned, and $\varepsilon > |a_i|$ then it shrinks at a rate proportional to $\varepsilon - |a_i|$. What happens when a non-zero weight w_i is aligned, i.e. $a_i \operatorname{sgn}(w_i) > 0$? In this case if $\varepsilon > |a_i|$, then the upper-bound (11) on the expected SGD update is once again negative (assuming $\overline{g'} \neq 0$), which means adversarial training with a *sufficiently large* ε that dominates the absolute strength of a feature will cause the weight of that feature to shrink on average. This observation is key to explaining the "feature-pruning" behavior of adversarial training: "weak" features (relative to ε) are weeded out by the SGD updates.

If w_i is aligned, and $\varepsilon < |a_i|$, then w_i expands up to a certain point. If a non-zero weight w_i is aligned and $\varepsilon < |a_i|$, then the upper bound (11) on $\overline{\Delta w_i}$ is non-negative. Since the Theorem states that equality holds in the limit as w_i approaches zero, this means if $|w_i|$ is sufficiently small, the expected SGD update $\overline{\Delta w_i}$ is non-negative, i.e., the weight w_i expands on average. In other words, weights of aligned features expand on average up to a certain point, if ε does not dominate their strength.

Note that Assumption LOSS-CVX implies that $\overline{g'} \geq 0$, and when the model $\boldsymbol{w}$ is "far" from the optimum, the values of $-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ will tend to be large, and since g' is a non-decreasing function (Assumption LOSS-CVX), $\overline{g'}$ will be large as well. So we can interpret $\overline{g'}$ as being a proxy for "average model error". Thus during the initial iterations of SGD, this quantity will tend to be large and positive, and shrinks toward zero as the model approaches optimality. Since $\overline{g'}$ appears as a factor in (10) and (11), we can conclude that the above effects will be more pronounced in the initial stages of SGD and less so in the later stages. The experimental results described in Section 6 are consistent with several of the above effects.

E. Generalization of Theorem 3.1 for the multi-class setting

E.1. Setting and Assumptions

Let there be $k \geq 3$ classes. For a given data point $\mathbf{x} \in \mathbb{R}^d$, its true label, $i \in [k]$, is represented by a vector $\mathbf{y} = [\underbrace{-1 \cdots -1}_{\text{i-1}}\, 1\, \underbrace{-1 \cdots -1}_{\text{k-i}}]$. We assume that the input $(\mathbf{x}, \mathbf{y})$ is drawn from the distribution $\mathcal{D}$. For this multi-class classification problem, we assume the usage of the standard one-vs-all classifiers, i.e., there are k different classifiers with the i-th classifier (ideally) predicting $+1$ iff the true label of $\mathbf{x}$ is i, else it predicts -1. Let $\mathbf{w}$ represent the $k \times d$ weight matrix where $\mathbf{w_i}$ represents the $1 \times d$ weight vector for the i-th classifier. w_{ij} represents the j-th entry of $\mathbf{w_i}$. Let y_i represent the i-th entry of $\mathbf{y}$.

The assumptions presented in the main paper (Sec. 2) are slightly tweaked as follows and hold true for each of the k one-vs-all classifiers:

Assumption LOSS-INC: *The loss function for each of the one-vs-all classifier is of the form* $\mathcal{L}(\mathbf{x}, y_i; \mathbf{w_i}) = g(-y_i \langle \mathbf{w_i}, \mathbf{x} \rangle)$ *where* g *is a non-decreasing function.*

Assumption LOSS-CVX: *The loss function for each of the one-vs-all classifier is of the form* $\mathcal{L}(\mathbf{x}, y_i; \mathbf{w_i}) = g(-y_i \langle \mathbf{w_i}, \mathbf{x} \rangle)$ *where* g *is non-decreasing, almost-everywhere differentiable and convex.*

Assumption FEAT-INDEP: *The features* $\mathbf{x}$ *are conditionally independent given the label* y_i *for the* i*-th one-vs-all classifier, i.e., for any two distinct induces* s, t, x_s *is independent of* x_t *given* y_i*, or more compactly,* $(x_s \perp x_t) \,|\, y_i$.

Assumption FEAT-EXP: *For each feature $x_j, j \in [d]$ and the i-th one-vs-all classifier $\mathbb{E}(x_j|y_i) = a_{ij}.y_i$ for some constant a_{ij}.*

Additionally, we introduce a new assumption on the distribution $\mathcal{D}$ as follows.

Assumption DIST-EXPC: *The input distribution $\mathcal{D}$ satisfies the following expectation for a function $h_i, i \in [k]$ (defined by Eqs. (E.42),(E.43), (E.44)) and constant g^* (defined by Eq. (E.41))*

$$\mathbb{E}\Big[h_i(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon)\Big] = 0 \tag{E.39}$$

$$Pr_{\mathcal{D}}[\epsilon < x_j < \epsilon + \rho] = 0, \quad \rho \text{ is a small constant} \tag{E.40}$$

$$x_j \geq \epsilon + \rho \implies (x_j - \epsilon)g_i^* \geq (x_j + \epsilon)g'(-y_i\langle \mathbf{w_i}, \mathbf{x} + \delta^* \rangle) \tag{E.41}$$

If $y_i\text{sgn}(w_{ij}) = -1$, then

$$h(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon) \geq (x_j + \epsilon)g_i^* - (x_j - \epsilon)g'(-y_i\langle \mathbf{w_i}, \mathbf{x} + \delta^* \rangle) \tag{E.42}$$

If $y_i\text{sgn}(w_{ij}) = 1 \wedge x_j > \epsilon$, then

$$-\Big((x_j - \epsilon)g_i^* - (x_j + \epsilon)g'(-y_i\langle \mathbf{w_i}, \mathbf{x} + \delta^* \rangle)\Big) \leq h(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon) \leq 0 \tag{E.43}$$

If $y_i\text{sgn}(w_{ij}) = 1 \wedge x_j \leq \epsilon$, then

$$h(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon) \geq (x_j + \epsilon)g'(-y_i\langle \mathbf{w_i}, \mathbf{x} + \delta^* \rangle) - (x_j - \epsilon)g_i^* \tag{E.44}$$

This assumption is not as restrictive as it may appear. Eq. E.40 can be satisfied naturally for discrete domains. For example, for images $x_j \in \{0, 1, 2, \cdots, 254, 255\}$; thus $\rho \in (0, 1)$. For continuous domains, ρ can be set to a small value and the values of x_j can be appropriately rounded in the input dataset.
For the rest of the discussion let us consider the case where $g'(z) = c$ (for example for hinge loss function $c = 1$ for $z > -1$) and $x_j \in [0, 1]$. Now consider,

$$g^* = (1 + \epsilon)c/\rho, \ \rho = 0.01$$
$$f_1(x) := ((x + \epsilon)g^* - (x - \epsilon)c)$$
$$f_2(x) := ((x + \epsilon)c - (x - \epsilon)g^*)$$
$$h_i(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon) = \begin{cases} f_1(x_j) & \text{if sgn}(w_{ij})y_i = +1 \\ f_2(x_j) & \text{otherwise} \end{cases}$$
$$\mathbb{E}[h_i(\text{sgn}(w_{ij})y_i, \mathbf{x}, \mathbf{w_i}, \epsilon)] = \int_0^1 Pr[x_j|y_i\text{sgn}(w_{ij}) = -1] \cdot f_1(x_j)dx +$$
$$\int_\epsilon^1 Pr[x_j|y_i\text{sgn}(w_{ij}) = +1] \cdot f_2(x)dx + \int_0^\epsilon Pr[x_j|y_i\text{sgn}(w_{ij}) = +1] \cdot f_2(x)dx$$

We observe that $f_1(x)$ is increasing in $x \in [0, 1]$, and $x \geq \epsilon + \delta \implies f_2(x) \leq 0$ and $f_2(x)$ is decreasing in $x \in [\epsilon + \delta, 1]$. Thus intuitively for $\mathbb{E}(h_i)$ to be zero, $Pr[x_j|\text{sgn}(w_{ij})y_i = -1]$ must have high values for lower magnitudes of x_j (say $x_j < 0.5$), and $Pr[x_j|\text{sgn}(w_{ij})y_i = -1]$ has low values for $x_j \leq \epsilon$ and high values for $x_j \geq \epsilon + \delta$. For example, let us assume $\epsilon = 0.1$ and that the distributions $Pr[x_i|\text{sgn}(w_{ij})y_i = +1]$ and $Pr[x_i|\text{sgn}(w_{ij})y_i = -1]$ can be approximated by truncated Gaussian distributions (with appropriate adjustments to ensure $Pr[\epsilon < x_j < \epsilon + \delta] = 0]$) with means m_1 and m_2 respectively. Then, it can be seen that there exists h_i for $m_1 < 0.3$ and $m_2 > 0.6$ such that $\mathbb{E}[h_i] = 0$.

The overall loss function, $\mathcal{L}_T$ for the multi-class classifier is the sum of the loss functions of each individual one-vs-all classifiers and is given by

$$\mathcal{L}_T(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \sum_{i=1}^{k} \mathcal{L}(\mathbf{x}, y_i; \mathbf{w_i})$$
$$= \sum_{i=1}^{k} g(-y_i\langle \mathbf{w_i}, \mathbf{x} \rangle) \qquad \text{[From Assumption } \textbf{LOSS-CVX}\text{]}$$

The **expected SGD update** $\overline{\Delta w_{ij}}$ is defined as follows:

$$\Delta w_{ij} = \begin{cases} \mathbb{E}\frac{\partial \mathcal{L}_T(\mathbf{x}+\delta^*,\mathbf{y};\mathbf{w})}{\partial w_{ij}} & \text{when } w_i = 0 \\ -\text{sgn}(w_{ij})\mathbb{E}\frac{\partial \mathcal{L}_T(\mathbf{x}+\delta^*,\mathbf{y};\mathbf{w})}{\partial w_{ij}} & \text{when } w_i \neq 0 \end{cases} \tag{E.45}$$

Also let

$$\overline{g_i'} := \mathbb{E}[g'(-y_i\langle \mathbf{w_i}, \mathbf{x}+\delta^*\rangle)], i \in [k] \tag{E.46}$$

Theorem E.1 (Expected SGD Update in Adversarial Training for Multi-Class Classification)**.** *For any loss function $\mathcal{L}$ satisfying assumptions **LOSS-CVX**, **FEAT-INDEP** and **FEAT-EXP**, if a data point $(\mathbf{x}, \mathbf{y})$ is randomly drawn from $\mathcal{D}$ that satisfies Assumption **DIST-EXPC**, and $\mathbf{x}$ is perturbed to $\mathbf{x}' = \mathbf{x} + \delta^*$, where δ^* is an $l_\infty(\epsilon)$-adversarial perturbation, then under the $l_\infty(\epsilon)$-adversarial loss $\mathcal{L}_T(\mathbf{x}, \mathbf{y}; \mathbf{w})$ the expected SGD-update of weight w_{ij}, namely $\overline{\Delta w_{ij}}$ satisfies the following properties*

1. *If $w_{ij} = 0$, then*

$$\overline{\Delta w_{ij}} = a_{ij}\overline{g_i'}$$

2. *If $w_{ij} \neq 0$, then*

$$\overline{\Delta w_{ij}} \leq \tilde{g}[a_{ij}\, sgn(w_{ij}) - \epsilon]$$
$$\tilde{g} \in \{\overline{g_i'}, g_i^*\}$$

Proof. For

$$\delta^* \in \mathbb{R}^d \text{ s.t} \tag{E.47}$$
$$\mathcal{L}_T(\mathbf{x}+\delta^*, \mathbf{y}; \mathbf{w}) = \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{D}}[\max_{||\delta||_\infty \leq \epsilon} \mathcal{L}_T(\mathbf{x}+\delta, \mathbf{y}; \mathbf{w})] \tag{E.48}$$

the shift by ϵ in x_j can be in either of the two directions, $y_i\text{sgn}(w_ij)$ or $-y_i\text{sgn}(w_ij)$. We have,

$$\frac{\partial \mathcal{L}_T(\mathbf{x}, \mathbf{y}; \mathbf{w}))}{\partial w_{ij}} = \sum_{s=1}^{k} \frac{\partial\big(g(-y_s\langle \mathbf{w}_s, \mathbf{x}+\delta^*\rangle)\big)}{\partial w_{ij}} = \frac{\partial g(-y_i\langle \mathbf{w}_i, \mathbf{x}+\delta^*\rangle)}{\partial w_{ij}} \tag{E.49}$$

Thus by Assumption **LOSS-CVX** either of the following two equations hold true

$$\frac{\partial \mathcal{L}_T(\mathbf{x}, \mathbf{y}; \mathbf{w}))}{\partial w_{ij}} = g'(-y_i\langle \mathbf{w}_i, \mathbf{x}+\delta^*\rangle)(-y_i x_j + \text{sgn}(w_{ij})\epsilon) \tag{E.50}$$
$$\frac{\partial \mathcal{L}_T(\mathbf{x}, \mathbf{y}; \mathbf{w}))}{\partial w_{ij}} = g'(-y_i\langle \mathbf{w}_i, \mathbf{x}+\delta^*\rangle)(-y_i x_j - \text{sgn}(w_{ij})\epsilon) \tag{E.51}$$

Thus when $w_{ij} = 0$,

$$\overline{\Delta w_{ij}} = \mathbb{E}\Big[y_i x_j g'(-y_i\langle \mathbf{w_i}, \mathbf{x}+\delta^*\rangle)\Big]$$
$$= a_{ij}\overline{g_i'}[\text{Follows from the proof in Theorem } \textbf{??} \text{ in the paper}]$$

Now let us consider the case when $w_{ij} \neq 0$.
Case I: Eq. E.50 is satisfied
This means that for δ^*, x_j is changed by ϵ in the direction of $-y_i\text{sgn}(w_{ij})$. Thus after multiplying throughout with $-\text{sgn}(w_{ij})$ and taking expectations, we have

$$\overline{\Delta w_{ij}} = \mathbb{E}\Big[[y_i x_j \text{sgn}(w_{ij}) - \epsilon]g'(\sum_{l=1, l\neq j}^{d} s_l\epsilon|w_{il}| + \epsilon|w_{ij}| - y_i\langle \mathbf{w_i}, \mathbf{x}\rangle)\Big] \tag{E.52}$$

where s_l represents the corresponding sign for the value $\epsilon|w_l|$ (based on which direction is x_l perturbed in) and the expectation is with respect to a random choice of data point $(\mathbf{x}, \mathbf{y})$. Let us define two random variables V and Z as follows

$$Z = y_i x_j \text{sgn}(w_{ij}) - \epsilon \tag{E.53}$$

$$V = -\sum_{l=1, l\neq j}^{d} |w_{il}| \Big(y_i x_l \text{sgn}(w_{il}) - s_l \epsilon \Big) \tag{E.54}$$

Thus,

$$\overline{\Delta w_{ij}} = \mathbb{E}[Z g'(V - |w_{ij}|Z)] \tag{E.55}$$

Let random variable Y correspond to the label y_i of the data point. Since Z is a function of feature x_j and Y , and V is a function of the remaining features and Y, Assumption **FEAT-INDEP** implies $(V \perp Z)|Y$. Additionally by Assumption **FEAT-EXP**

$$\mathbb{E}(Z) = \mathbb{E}(Z|Y) = a_j \text{sgn}(w_{ij}) - \epsilon \tag{E.56}$$

Since by Assumption **LOSS-CVX**, g' is a non-decreasing function, $g'(V - |w_{ij}|Z)$ is non-increasing in Z. Thus all three conditions of Lemma 4 are satisfied, with the random variables Z, V, Y and function f in the Lemma corresponding to random variables Z, V, Y and function g' respectively in the present theorem. Following an analysis similar to the one presented in the proof for Theorem **??**, we have

$$\overline{\Delta w_{ij}} \leq \mathbb{E}(Z)\overline{g'_i} = \overline{g'_i}[a_{ij}\text{sgn}(w_{ij}) - \epsilon] \tag{E.57}$$

Case II: Eq. E.51 is satisfied
In this case, for δ^* x_j is perturbed by ϵ in the direction $y_i \text{sgn}(w_{ij})$. Now multiplying both sides by $-\text{sgn}(w_{ij})$

$$\Delta w_{ij} = (y_i x_j \text{sgn}(w_{ij}) + \epsilon) g'(-y_i \langle \mathbf{w_i}, \mathbf{x} + \delta^* \rangle) \tag{E.58}$$

Now let us consider the case when $y_i \text{sgn}(w_{ij}) = -1$. From Assumption **DIST-EXPC** (Eqs. (E.58),(E.41) and (E.42)), we have

$$\Delta w_{ij} \leq (y_i x_j \text{sgn}(w_{ij}) - \epsilon) g_i^* + h_i(\text{sgn}(w_{ij}) y_i, \mathbf{x}, \mathbf{w_i}, \epsilon) \tag{E.59}$$

For the case when $y_i \text{sgn}(w_{ij}) = -1 \wedge x_j > \epsilon$, again from Eqs. (E.58),(E.41) and (E.43) Eq. (E.59) holds true. Similarly, for the case of $y_i \text{sgn}(w_{ij}) = -1 \wedge x_j \leq \epsilon$, the validity of Eq. (E.59) can be verified from Eqs. (E.58),(E.41) and (E.44). Now taking expectations over both sides of Eq. (E.59) results in

$$\overline{\Delta w_{ij}} \leq (a_{ij}\text{sgn}(w_{ij}) - \epsilon) g_i^* \qquad \text{[From Eq. (E.39)]} \tag{E.60}$$

This concludes our proof. □

F. Proof of Lemma 4.1

Lemma 4.1 (IG Attribution for 1-layer Networks)**.** *If $F(\boldsymbol{x})$ is computed by a 1-layer network (13) with weights vector $\boldsymbol{w}$, then the Integrated Gradients for all dimensions of $\boldsymbol{x}$ relative to a baseline $\boldsymbol{u}$ are given by:*

$$\text{IG}^F(\boldsymbol{x}, \boldsymbol{u}) = [F(\boldsymbol{x}) - F(\boldsymbol{u})] \frac{(\boldsymbol{x} - \boldsymbol{u}) \odot \boldsymbol{w}}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle}, \tag{14}$$

where the $\odot$ operator denotes the entry-wise product of vectors.

Proof. Since the function F, the baseline input $\boldsymbol{u}$ and weight vector $\boldsymbol{w}$ are fixed, we omit them from $\text{IG}^F(\boldsymbol{x}, \boldsymbol{u})$ and $\text{IG}_i^F(\boldsymbol{x}, \boldsymbol{u})$ for brevity. Consider the partial derivative $\partial_i F(\boldsymbol{u} + \alpha(\boldsymbol{x} - \boldsymbol{u}))$ in the definition (12) of $\text{IG}_i(\boldsymbol{x})$. For a given $\boldsymbol{x}$, $\boldsymbol{u}$ and α, let $\boldsymbol{v}$ denote the vector $\boldsymbol{u} + \alpha(\boldsymbol{x} - \boldsymbol{u})$. Then $\partial_i F(\boldsymbol{v}) = \partial F(\boldsymbol{v}) / \partial v_i$, and by applying the chain rule we get:

$$\partial_i F(\boldsymbol{v}) := \frac{\partial F(\boldsymbol{v})}{\partial v_i} = \frac{\partial A(\langle \boldsymbol{w}, \boldsymbol{v} \rangle)}{\partial v_i} = A'(z) \frac{\partial \langle \boldsymbol{w}, \boldsymbol{v} \rangle}{\partial v_i} = w_i A'(z),$$

where $A'(z)$ is the gradient of the activation A at $z = \langle \boldsymbol{w}, \boldsymbol{v} \rangle$. This implies that:

$$\begin{aligned}\frac{\partial F(\boldsymbol{v})}{\partial \alpha} &= \sum_{i=1}^{d} \left(\frac{\partial F(\boldsymbol{v})}{\partial v_i} \frac{\partial v_i}{\partial \alpha} \right) \\ &= \sum_{i=1}^{d} [w_i A'(z)(x_i - u_i)] \\ &= \langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle A'(z)\end{aligned}$$

We can therefore write

$$dF(\boldsymbol{v}) = \langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle A'(z) d\alpha,$$

and since $\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle$ is a scalar, this yields

$$A'(z) d\alpha = \frac{dF(\boldsymbol{v})}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle}$$

Using this equation the integral in the definition of $IG_i(x)$ can be written as

$$\begin{aligned}\int_{\alpha=0}^{1} \partial_i F(\boldsymbol{v}) d\alpha &= \int_{\alpha=0}^{1} w_i A'(z) d\alpha \\ &= \int_{\alpha=0}^{1} w_i \frac{dF(\boldsymbol{v})}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle} \\ &= \frac{w_i}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle} \int_{\alpha=0}^{1} dF(\boldsymbol{v}) \qquad \text{(F.61)} \\ &= \frac{w_i}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle} [F(\boldsymbol{x}) - F(\boldsymbol{u})],\end{aligned}$$

where (F.61) follows from the fact that $(\boldsymbol{x} - \boldsymbol{u})$ and $\boldsymbol{w}$ do not depend on α. Therefore from the definition (12) of $\mathrm{IG}_i(\boldsymbol{x})$:

$$\mathrm{IG}_i(\boldsymbol{x}) = [F(\boldsymbol{x}) - F(\boldsymbol{u})] \frac{(x_i - u_i) w_i}{\langle \boldsymbol{x} - \boldsymbol{u}, \boldsymbol{w} \rangle},$$

and this yields the expression (14) for $\mathrm{IG}(\boldsymbol{x})$. □

G. Proof of Theorem 5.1

Theorem 5.1 (Equivalence of Stable IG and Adversarial Robustness)**.** *For loss functions $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$ satisfying Assumption LOSS-CVX, the augmented loss inside the expectation (16) equals the $\ell_\infty(\varepsilon)$-adversarial loss inside the expectation (2), i.e.*

$$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) \; + \max_{||\boldsymbol{x}' - \boldsymbol{x}||_\infty \le \varepsilon} ||\, \mathrm{IG}^{\mathcal{L}_y}(\boldsymbol{x}, \boldsymbol{x}')||_1 \;\; = \;\; \max_{||\delta||_\infty \le \varepsilon} \mathcal{L}(\boldsymbol{x} + \delta, y; \boldsymbol{w}) \qquad (17)$$

Proof. Recall that Assumption LOSS-CVX implies $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = g(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle)$ for some non-decreasing, differentiable, convex function g. Due to this special form of $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w})$, the function $\mathcal{L}_y$ is a differential function of $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$, and by Lemma 4.1 the i'th component of the IG term in (17) is

$$\mathrm{IG}_i^{\mathcal{L}_y}(\boldsymbol{x}, \boldsymbol{x}';\, \boldsymbol{w}) = \frac{\boldsymbol{w}_i (\boldsymbol{x}' - \boldsymbol{x})_i}{\langle \boldsymbol{w}, \boldsymbol{x}' - \boldsymbol{x} \rangle} \cdot \big(g(-y\langle \boldsymbol{w}, \boldsymbol{x}' \rangle) - g(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle), \big)$$

and if we let $\Delta = \boldsymbol{x}' - \boldsymbol{x}$ (which satisfies that $\|\Delta\|_\infty \le \varepsilon$), its absolute value can be written as

$$\frac{\big| g(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle - y\langle \boldsymbol{w}, \Delta \rangle) \;-\; g(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle) \big|}{|\langle \boldsymbol{w}, \Delta \rangle|} \cdot |\, \boldsymbol{w}_i \, \Delta_i|$$

Let $z = -y\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ and $\delta = -y\langle \boldsymbol{w}, \Delta \rangle$, this is further simplified as $\frac{|g(z+\delta)-g(z)|}{|\delta|}|w_i \Delta_i|$. By Assumption LOSS-CVX, g is convex, and therefore the "chord slope" $[g(z+\delta) - g(z)]/\delta$ cannot decrease as δ is increased. In particular to maximize the ℓ_1-norm of the IG term in Eq (17), we can set δ to be largest possible value subject to the constraint $||\Delta||_\infty \leq \varepsilon$, and we achieve this by setting $\Delta_i = -y\,\mathrm{sgn}(\boldsymbol{w}_i)\varepsilon$, for each dimension i. This yields $\delta = \| \boldsymbol{w} \|_1 \varepsilon$, and the second term on the LHS of (17) becomes

$$\begin{aligned} |g(z+\delta) - g(z)| \cdot \frac{\sum_i |\boldsymbol{w}_i\, \Delta_i|}{|\delta|} &= |g(z + \varepsilon \| \boldsymbol{w} \|_1) - g(z)| \cdot \frac{\sum_i |\boldsymbol{w}_i|\varepsilon}{\| \boldsymbol{w} \|_1 \varepsilon} \\ &= |g(z + \varepsilon \| \boldsymbol{w} \|_1) - g(z)| \\ &= g(z + \varepsilon \| \boldsymbol{w} \|_1) - g(z) \end{aligned}$$

where the last equality follows because g is nondecreasing. Since $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}) = g(z)$ by Assumption LOSS-CVX, the LHS of (17) simplifies to

$$g(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle + \varepsilon \| \boldsymbol{w} \|_1),$$

and by Eq. (5), this is exactly the $\ell_\infty(\varepsilon)$-adversarial loss on the RHS of (17). □

H. Aggregate IG Attribution over a Dataset

Recall that in Section 4 we defined $\mathrm{IG}^F(\boldsymbol{x}, \boldsymbol{u})$ in Eq. (12) for a *single* input $\boldsymbol{x}$ (relative to a baseline input $\boldsymbol{u}$). This gives us a sense of the "importance" of each input feature in explaining a *specific* model prediction $F(\boldsymbol{x})$. Now we describe some ways to produce *aggregate* importance metrics over an entire dataset. For brevity let us simply write $\mathrm{IG}(\boldsymbol{x})$ and $\mathrm{IG}_i(\boldsymbol{x})$ and omit F and $\boldsymbol{u}$ since these are fixed for a given model and a given dataset.

Note that in Eq. 12, $\boldsymbol{x}$ is assumed to be an input vector in "exploded" space, i.e., all categorical features are (explicitly or implicitly) one-hot encoded, and i is the position-index corresponding to either a specific numerical feature, or a categorical feature-*value*. Thus if i corresponds to a categorical feature-value, then for any input $\boldsymbol{x}$ where $x_i = 0$ (i.e. the corresponding categorical feature-value is not "active" for that input), $\mathrm{IG}_i(\boldsymbol{x}) = 0$. A natural definition of the overall importance of a feature (or feature-value) i for a given model F and dataset $\mathcal{D}$, is the average of $|\,\mathrm{IG}_i(\boldsymbol{x})|$ over all inputs $\boldsymbol{x} \in \mathcal{D}$, which we refer to as the **Feature Value Impact** $FV_i[\mathcal{D}]$. For a categorical feature with m possible values, we can further define its **Feature-Impact** (FI) as the *sum* of $FV_i[\mathcal{D}]$ over all i corresponding to possible values of this categorical feature.

The FI metric is particularly useful in tabular datasets to gain an understanding of the aggregate importance of high-cardinality categorical features.

I. Definition of the Gini Index

The definition is adapted from (Hurley & Rickard, 2009): Suppose we are given a vector of non-negative values $\boldsymbol{v} = [v_1, v_2, v_3, \ldots, v_d]$. The vector is first *sorted* in non-decreasing order, so that the resulting indices after sorting are $(1), (2), (3), \ldots, (d)$, i.e., $v_{(k)}$ denotes the k'th value in this sequence. Then the Gini Index is given by:

$$G(\boldsymbol{v}) = 1 - 2\sum_{k=1}^{d} \frac{v_{(k)}}{||\,\boldsymbol{v}\,||_1}\Big(\frac{d-k+0.5}{d}\Big). \tag{I.62}$$

Another equivalent definition of the Gini Index is based on plotting the cumulative fractional contribution of the sorted values. In particular if the sorted non-negative values are $[v_{(1)}, v_{(2)}, \ldots, v_{(d)}]$, and for $k \in [d]$, we plot k/d (the fraction of dimensions up to k) vs $\frac{\sum_{i=1}^{k} v_{(i)}}{||\,\boldsymbol{v}\,||_1}$ (the fraction of values until the k'th dimension), then the Gini Index $G(\boldsymbol{v})$ is 0.5 minus the area under this curve

The Gini Index by definition lies in [0,1], and a higher value indicates more sparseness. For example if just one of the $v_i > 0$ and all the rest are 0, then $G(v) = 1.0$, indicating perfect sparseness. At the other extreme, if all v_i are equal to some positive constant, then $G(v) = 0$.

J. Experiments

J.1. Experiment Datasets and Methodology

We experiment with 4 public benchmark datasets. Below we briefly describe each dataset and model-training details.

MNIST. This is a classic image benchmark dataset consisting of grayscale images of handwritten digits 0 to 9 in the form of 28 x 28 pixels, along with the correct class label (0 to 9) (LeCun & Cortes, 2010). We train a Deep Neural Network consisting of two convolutional layers with 32 and 64 filters respectively, each followed by 2x2 max-pooling, and a fully connected layer of size 1024. Note that this is identical to the state-of-the-art adversarially trained model used by (Madry et al., 2017). We use 50,000 images for training, and 10,000 images for testing. When computing the IG vector for an input image, we use the predicted probability of the *true class* as the function F in the definition (12) of IG. For training each of the model types on MNIST, we use the Adam optimizer with a learning rate 10^{-4}, with a batch size of 50. For the naturally-trained model (with or without ℓ_1-regularization) we use 25,000 training steps. For adversarial training, we use 100,000 training steps overall, and to generate adversarial examples we use Projected Gradient Descent (PGD) with random start. The PGD hyperparameters depend on the specific ε bound on the ℓ_∞-norm of the adversarial perturbations: the number of PGD steps was set as $\varepsilon * 100 + 10$, and the PGD step size was set to 0.01.

Fashion-MNIST. This is another image benchmark dataset which is a drop-in replacement for MNIST (Xiao et al., 2017). Images in this dataset depict wearables such as shirts and boots instead of digits. The image format, the number of classes, as well as the number of train/test examples are all identical to MNIST. We use the same model and training details as for MNIST.

Mushroom. This is a standard tabular public dataset from the UCI Data Repository (Dheeru & Karra Taniskidou, 2017). The dataset consists of 8142 instances, each of which corresponds to a different mushroom species, and has 22 categorical features (and no numerical features), whose cardinalities are all under 10. The task is to classify an instance as edible (label=1) or not (label=0). We train a simple *logistic regression* model to predict the probability that the mushroom is edible, with a 70/30 train/test split, and use a 0.5 threshold to make the final classification. We train the models on 1-hot encoded feature vectors, and the IG computation is on these (sparse) 1-hot vectors, with the output function F being the final predicted probability. We train logistic regression models for this dataset, and for natural model training (with or without ℓ_1-regularization) we use the Adam optimizer with a learning rate of 0.01, batch size of 32, and 30 training epochs. Adversarial training is similar, except that each example batch is perturbed using the closed-form expression (5).

Spambase. This is another tabular dataset from the UCI Repository, consisting of 4601 instances with 57 numerical attributes (and no categorical ones). The instances are various numerical features of a specific email, and the task is classify the email as spam (label = 1) or not (label = 0). The model and training details are similar to those for the mushroom dataset.

The code for all experiments (included along with the supplement) was written using Tensorflow 2.0. The following subsections contain results that were left out of the main body of the paper due to space constraints.

J.2. Mushroom Dataset: Average IG-based Feature Impact

We contrast between the weights learned by natural training and adversarial training with $\varepsilon = 0.1$. Since all features in this dataset are categorical, many with cardinalities close to 10, there are too many features in the "exploded" space to allow a clean display, so we instead look at the average Feature Impact (FI, defined in Section H) over the (natural, unperturbed) test dataset, see Figure J.4. It is worth noting that several features that have a significant impact on the naturally-trained model have essentially no impact on the adversarially trained model.

J.3. Spambase: Average IG-based Feature Impact

We fix $\varepsilon = 0.1$ for adversarial training and show in Figure J.5 a bar-plot comparing the average Feature-Impacts (FI), between naturally-trained and adversarially-trained models. Note how the adversarially trained model has significantly fewer features with non-negligible impacts, compared to a naturally trained model.

J.4. MNIST and Fashion-MNIST: examples

Figs. J.6 and J.7 below show IG-based saliency maps of images correctly classified by three model types: Naturally trained un-regularized model, naturally trained model with ℓ_1-regularization, and an $\ell_\infty(\varepsilon)$-adversarially trained model. The values

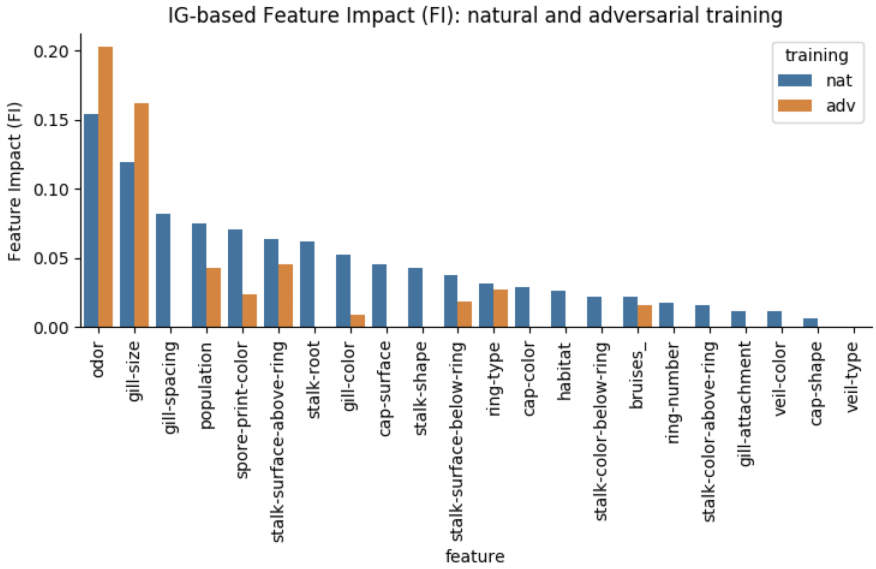


Figure J.4: Comparison of aggregate Feature Impact (FI) for a naturally-trained model, and an adversarially-trained model with $\varepsilon = 0.1$, on the mushroom dataset. The features are arranged left to right in decreasing order of the FI value in the naturally-trained model.

of λ and ε are those indicated in Table 1. In each example, all three models predict the correct class with high probability, and we compare the Gini Indices of the IG-vectors (with respect to the predicted probability of the true class). The sparseness of the saliency maps of the adversarially-trained models is visually striking compared to those of the other two models, and this is reflected in the Gini Indices as well. Figs. J.8 and J.9 show analogous results, but using the DeepSHAP (Lundberg & Lee, 2017) attribution method instead of IG. The effect of adversarial training on the sparseness of the saliency maps is even more visually striking when using DeepSHAP, compared to IG.

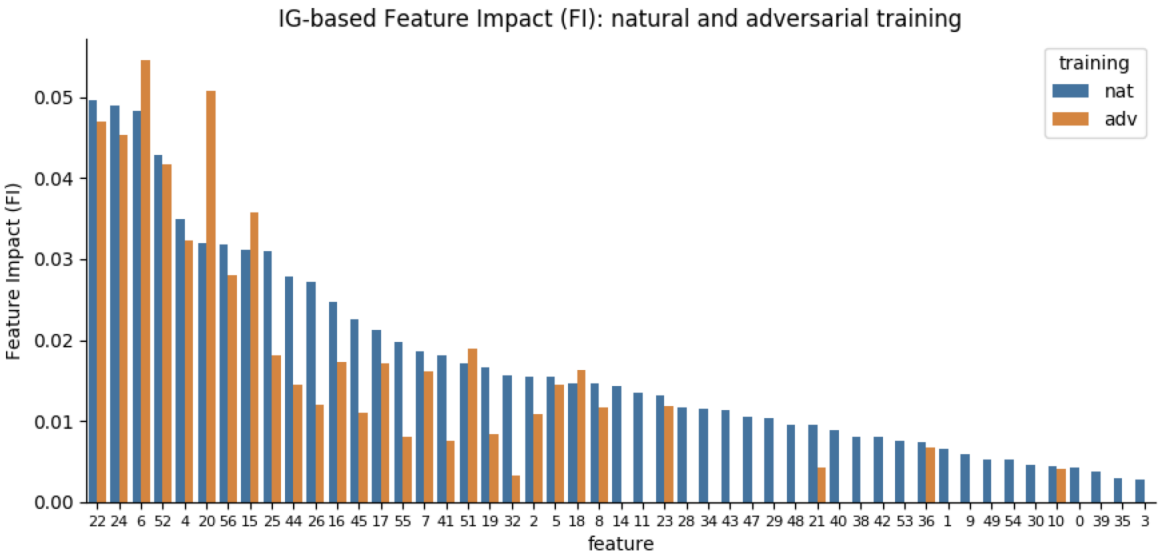
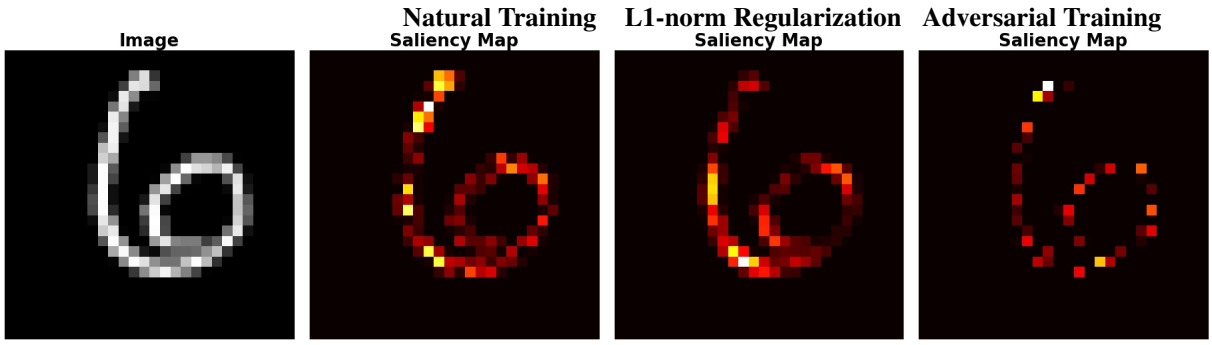
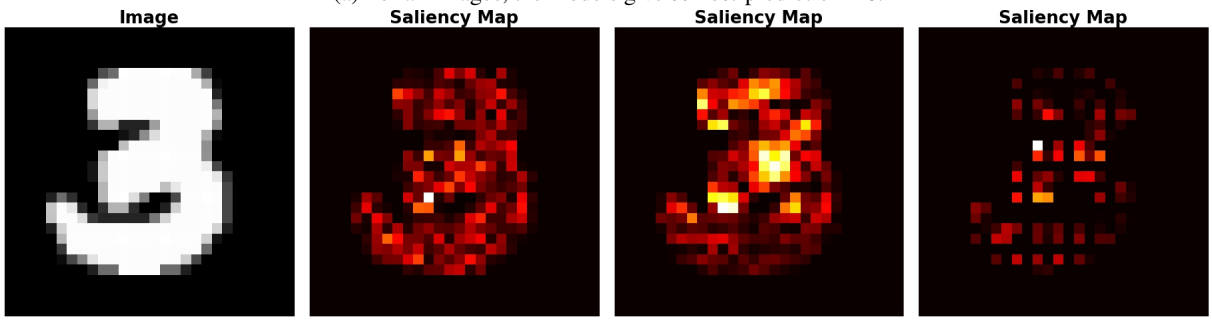


Figure J.5: Comparison of aggregate Feature Impact (FI) for a naturally-trained model, and an adversarially-trained model with $\varepsilon = 0.1$, on the spambase dataset. The features are arranged left to right in decreasing order of their FI in the naturally-trained model. To avoid clutter, we show only features that have an FI at least 5% of the highest FI (across both models).



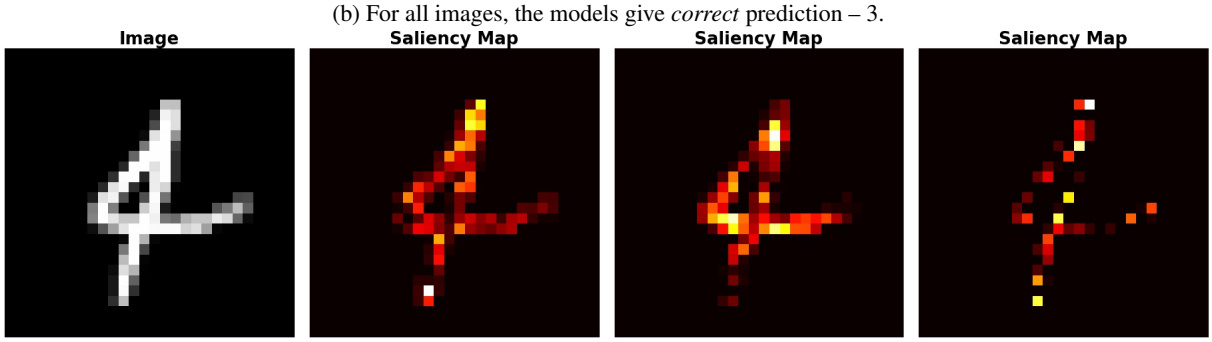
Gini: 0.9271 Gini: 0.9266 Gini: 0.9728

(a) For all images, the models give *correct* prediction – 6.



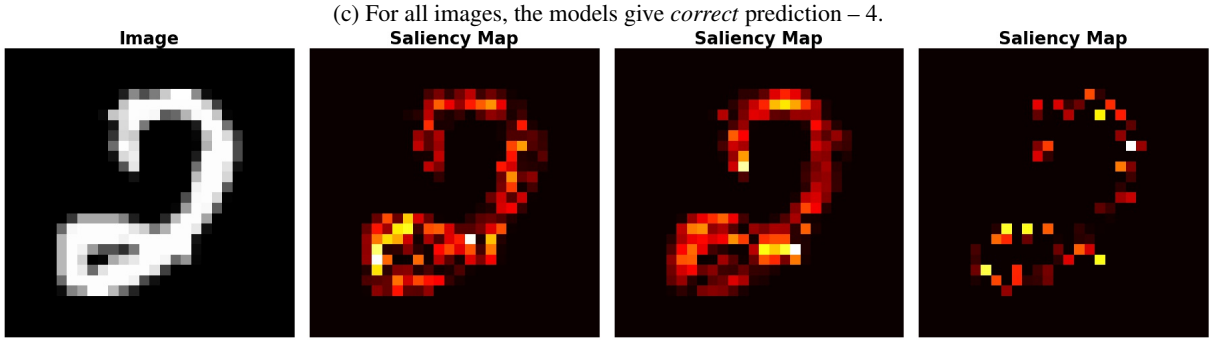
Gini: 0.8112 Gini: 0.8356 Gini: 0.9383

(b) For all images, the models give *correct* prediction – 3.



Gini: 0.9315 Gini: 0.9366 Gini: 0.9738

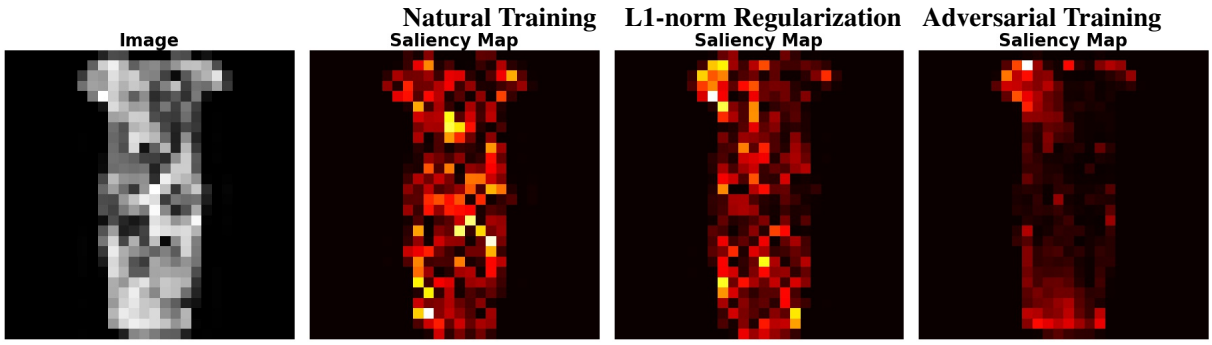
(c) For all images, the models give *correct* prediction – 4.



Gini: 0.8843 Gini: 0.8807 Gini: 0.9595

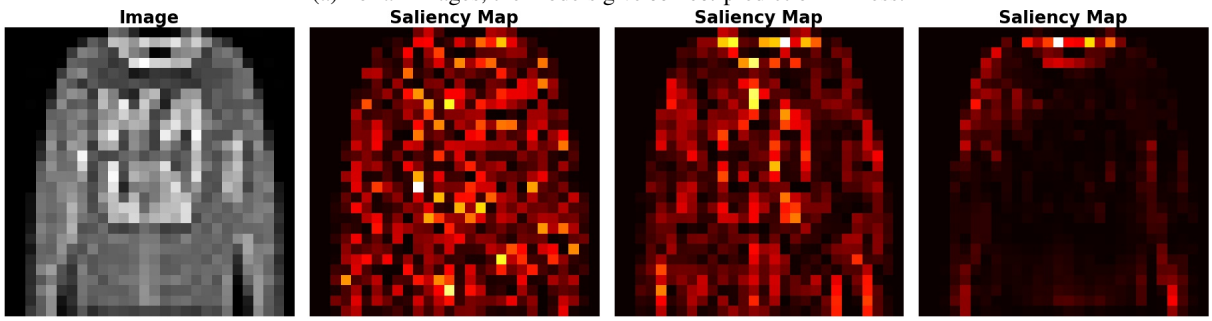
(d) For all images, the models give *correct* prediction – 2.

Figure J.6: Some examples on MNIST. We can see the saliency maps (also called feature importance maps), computed via IG, of adversarially trained model are much sparser compared to other models.



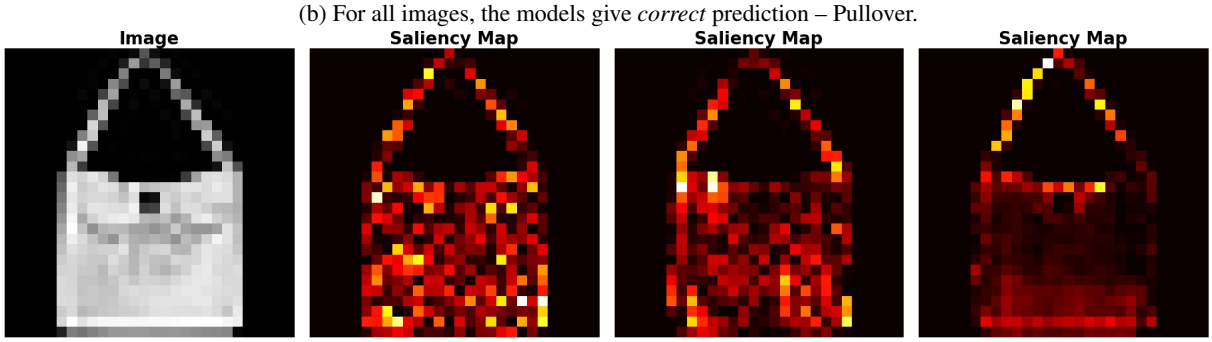
Gini: 0.8190 Gini: 0.8183 Gini: 0.8532

(a) For all images, the models give *correct* prediction – Dress.



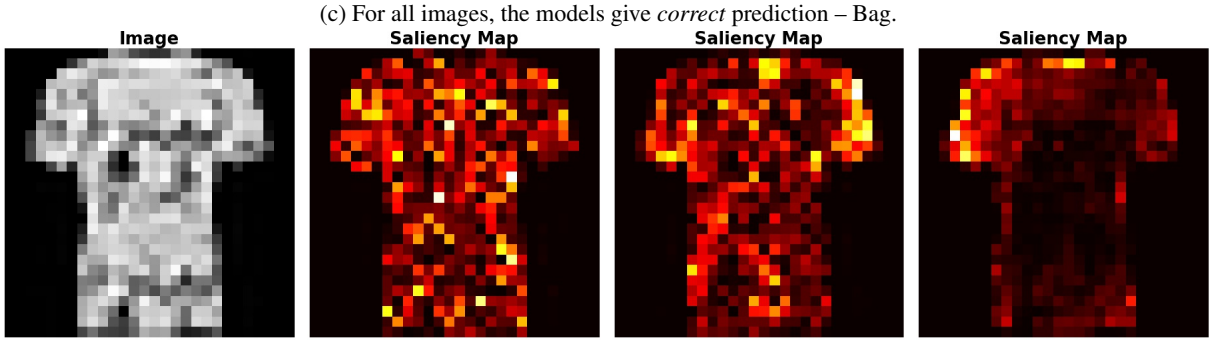
Gini: 0.5777 Gini: 0.5925 Gini: 0.7024

(b) For all images, the models give *correct* prediction – Pullover.



Gini: 0.7698 Gini: 0.7784 Gini: 0.7981

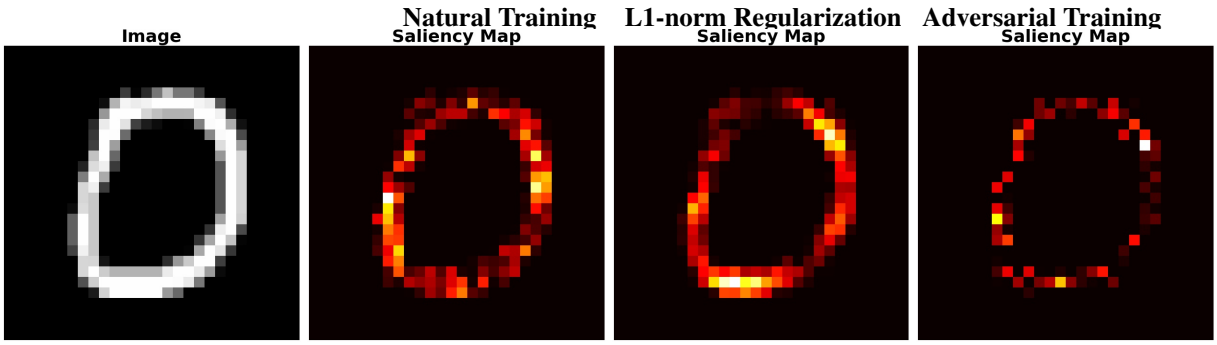
(c) For all images, the models give *correct* prediction – Bag.



Gini: 0.6840 Gini: 0.6899 Gini: 0.7503

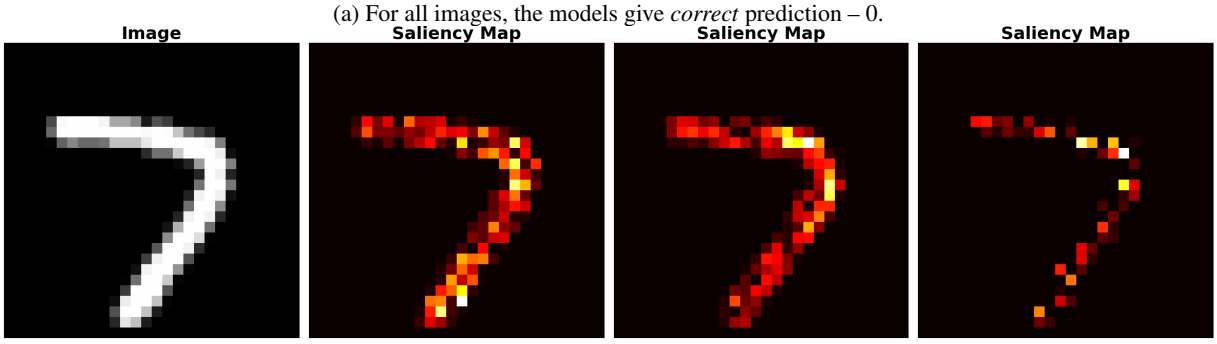
(d) For all images, the models give *correct* prediction – T-shirt.

Figure J.7: Some examples on Fashion-MNIST. We can see the saliency maps (also called feature importance maps), computed via IG, of adversarially trained model are much sparser compared to other models.



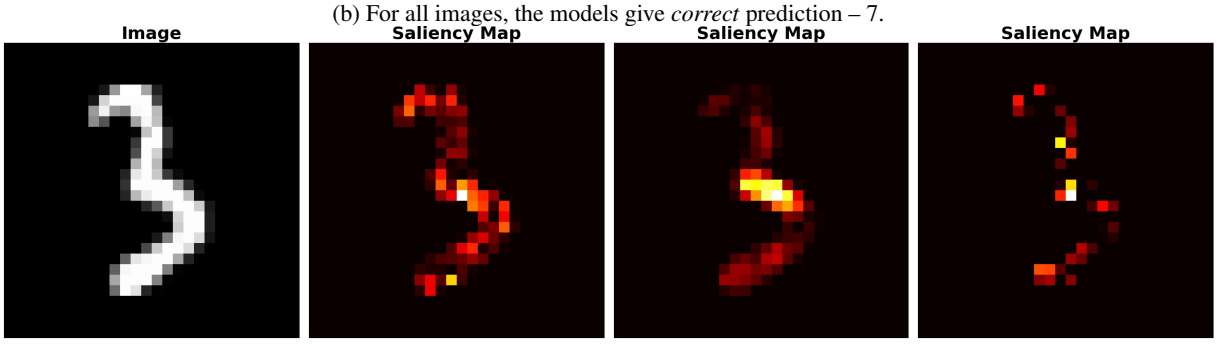
Gini: 0.8982 Gini: 0.9000 Gini: 0.9528

(a) For all images, the models give *correct* prediction – 0.



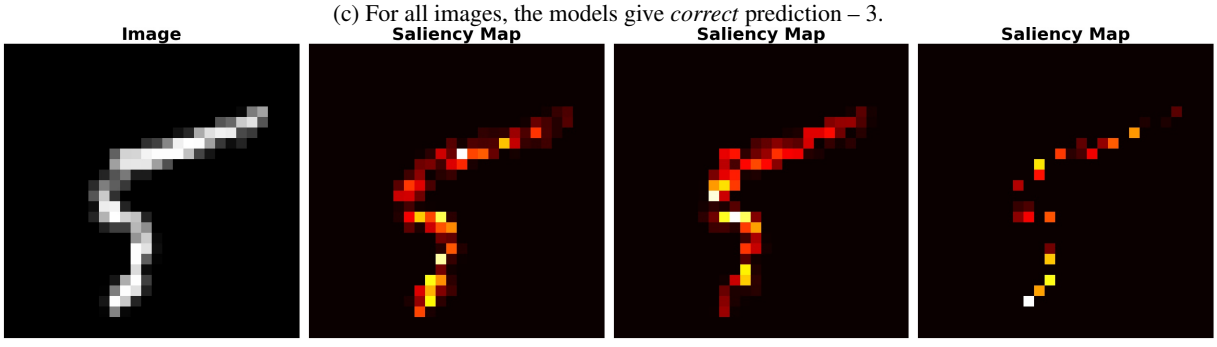
Gini: 0.9156 Gini: 0.9156 Gini: 0.9685

(b) For all images, the models give *correct* prediction – 7.



Gini: 0.9373 Gini: 0.9452 Gini: 0.9773

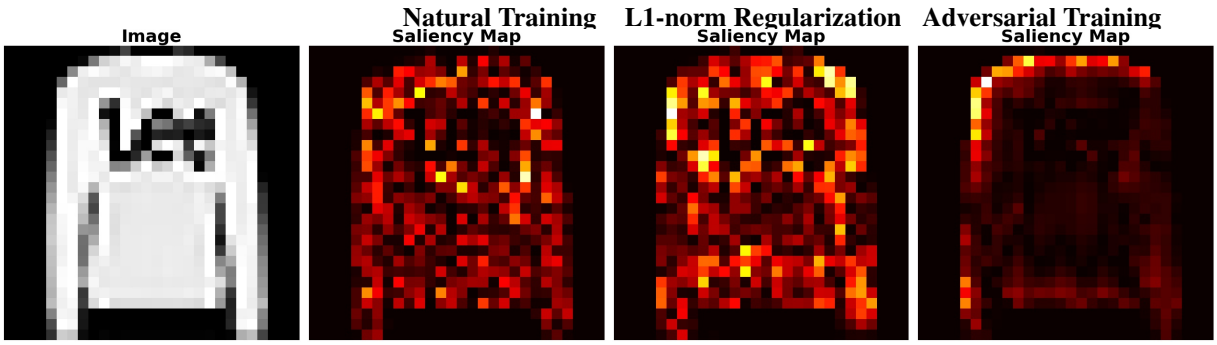
(c) For all images, the models give *correct* prediction – 3.



Gini: 0.9476 Gini: 0.9473 Gini: 0.9825

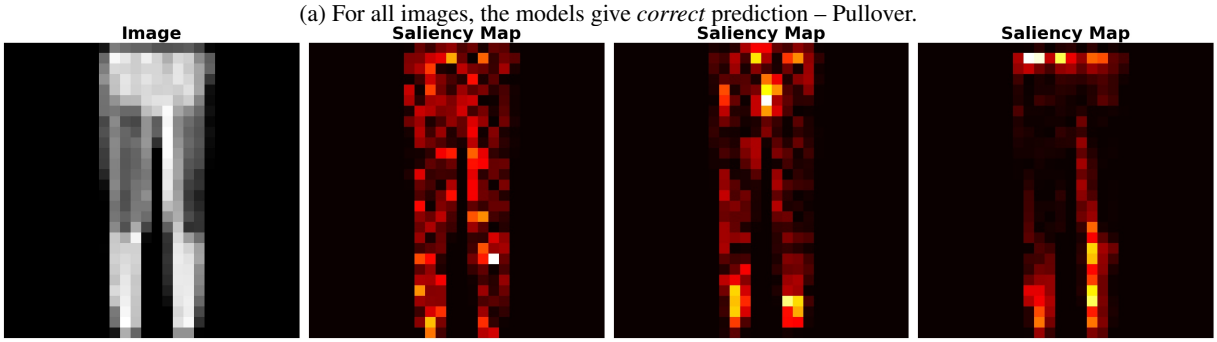
(d) For all images, the models give *correct* prediction – 5.

Figure J.8: Some examples on MNIST. We can see the saliency maps (also called feature importance maps), computed via DeepSHAP, of adversarially trained model are much sparser compared to other models.



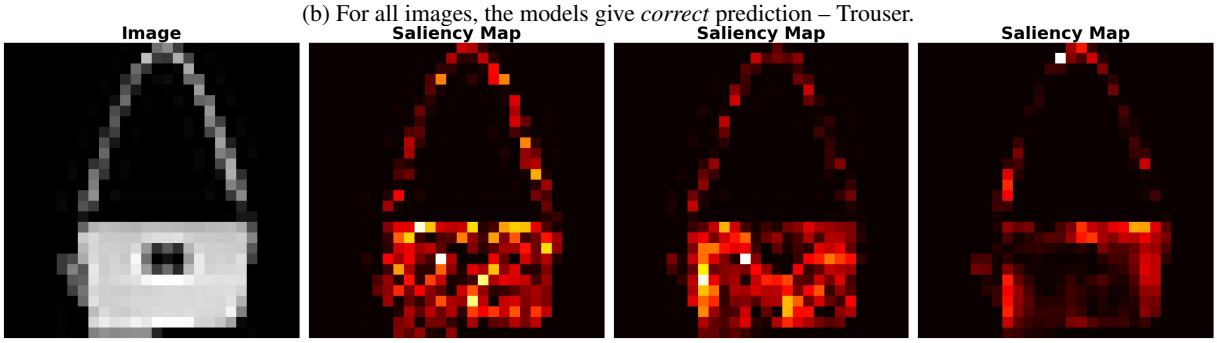
Gini: 0.6749 Gini: 0.6676 Gini: 0.7435

(a) For all images, the models give *correct* prediction – Pullover.



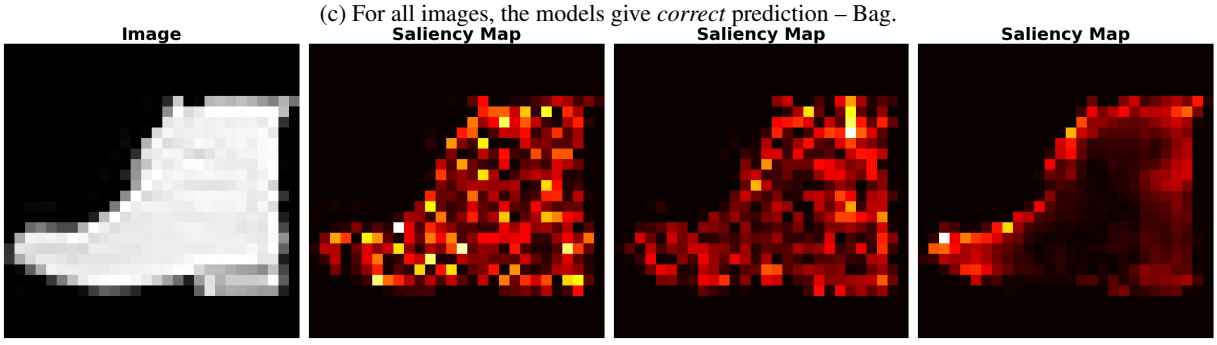
Gini: 0.8322 Gini: 0.8628 Gini: 0.8953

(b) For all images, the models give *correct* prediction – Trouser.



Gini: 0.8343 Gini: 0.8374 Gini: 0.8683

(c) For all images, the models give *correct* prediction – Bag.



Gini: 0.7701 Gini: 0.7575 Gini: 0.7920

(d) For all images, the models give *correct* prediction – Ankle boot.

Figure J.9: Some examples on Fashion-MNIST. We can see the saliency maps (also called feature importance maps), computed via DeepSHAP, of adversarially trained model are much sparser compared to other models.