

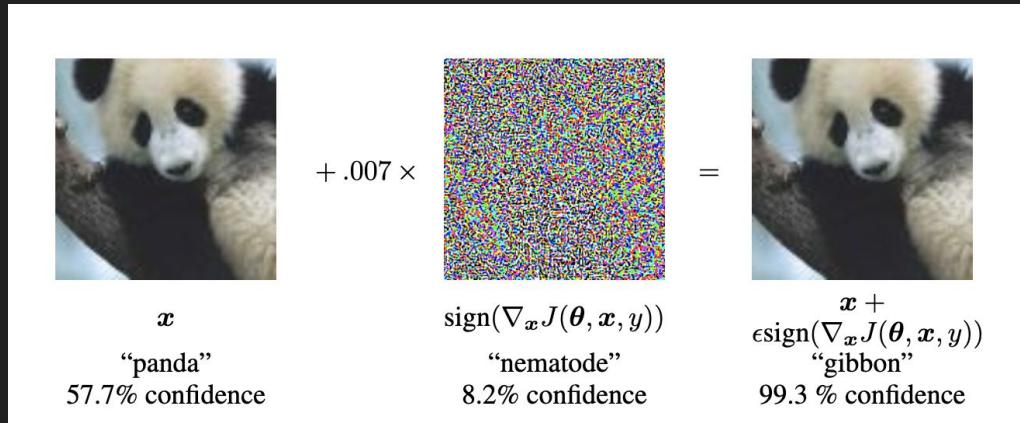
midterm progress update: counterintuitive properties of advx



a traditional introduction

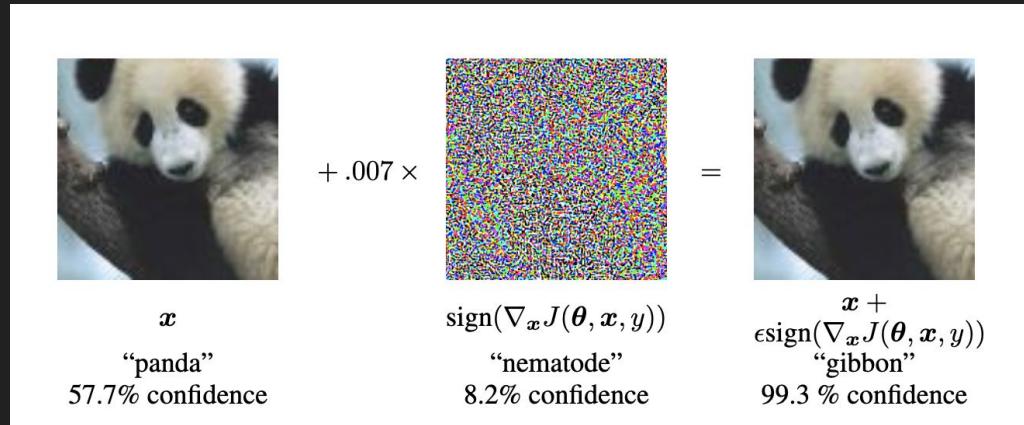
adversarial examples (in computer vision):

X	input image	
P	perturbation	
$X + P + \epsilon$	perturbed input	no semantic loss, recognizable to humans
$f: X \rightarrow y$	classifier	trying to fit human perception
$h: X \rightarrow y$	human judgement	implicit
$f(X + P) = t$	misclassification	target based on perturbation



adversarial examples (in computer vision):

X	input image	
P	perturbation	
$X + P + \epsilon$	perturbed input	no semantic loss, recognizable to humans
$f: X \rightarrow y$	classifier	trying to fit human perception
$h: X \rightarrow y$	human judgement	implicit
$f(X + P) = t$	misclassification	target based on perturbation



can you tell the difference?

if not, then that's a bad sign.

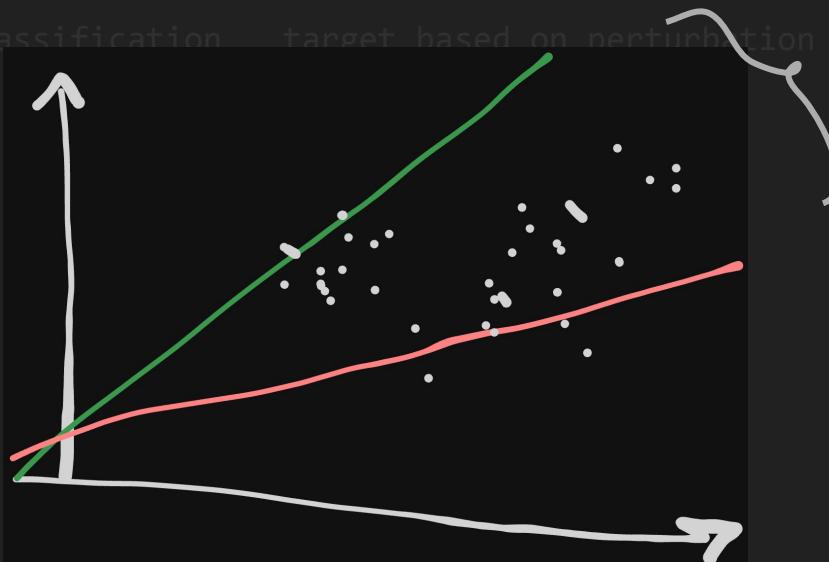
adversarial examples (in computer vision):

X	input image	
P	perturbation	
$X + P + \epsilon$	perturbed input	no semantic loss, recognizable to humans

$f: X \rightarrow y$ classifier trying to fit human perception
 $h: X \rightarrow y$ human judgement implicit

$f(X + P) = t$ misclassification target based on perturbation

human-machine vision gap



adversarial examples (in computer vision):

$$\begin{array}{l} X \\ P \\ X + P + \varepsilon \end{array}$$

input image
perturbation
perturbed input

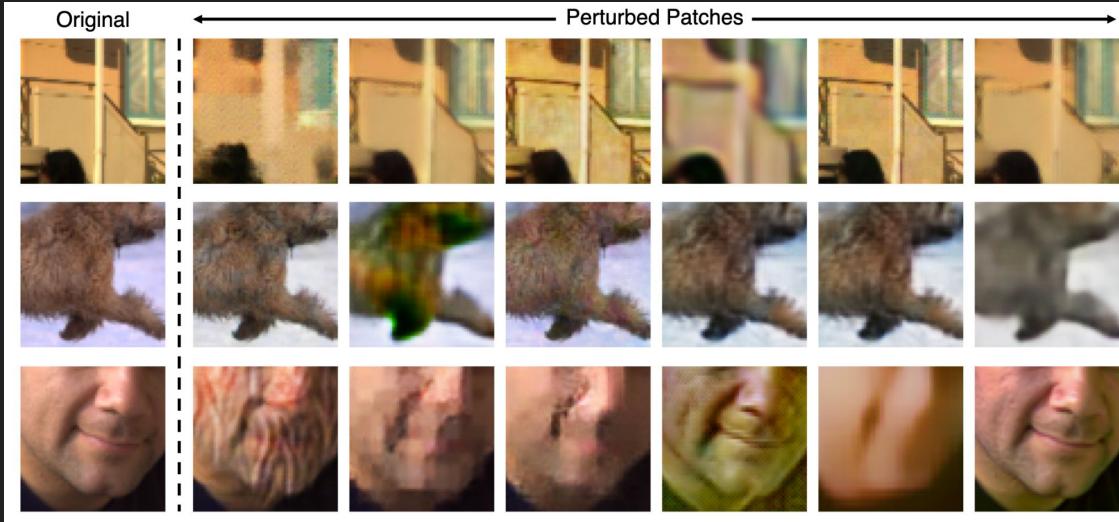
no semantic loss, recognizable to humans

$$\begin{array}{l} f: X \rightarrow y \\ h: X \rightarrow y \end{array}$$

classifier
human judgement

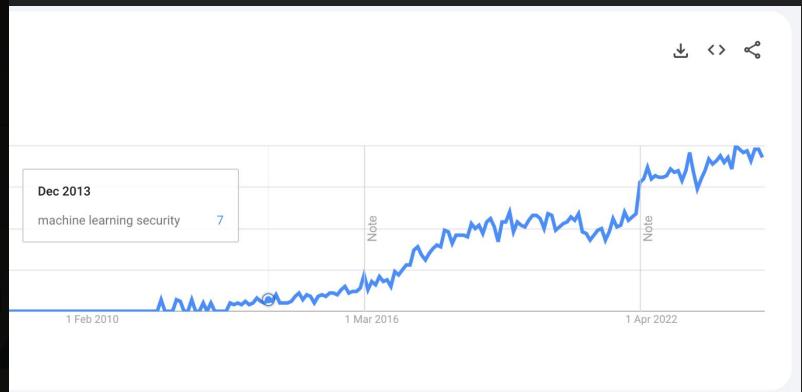
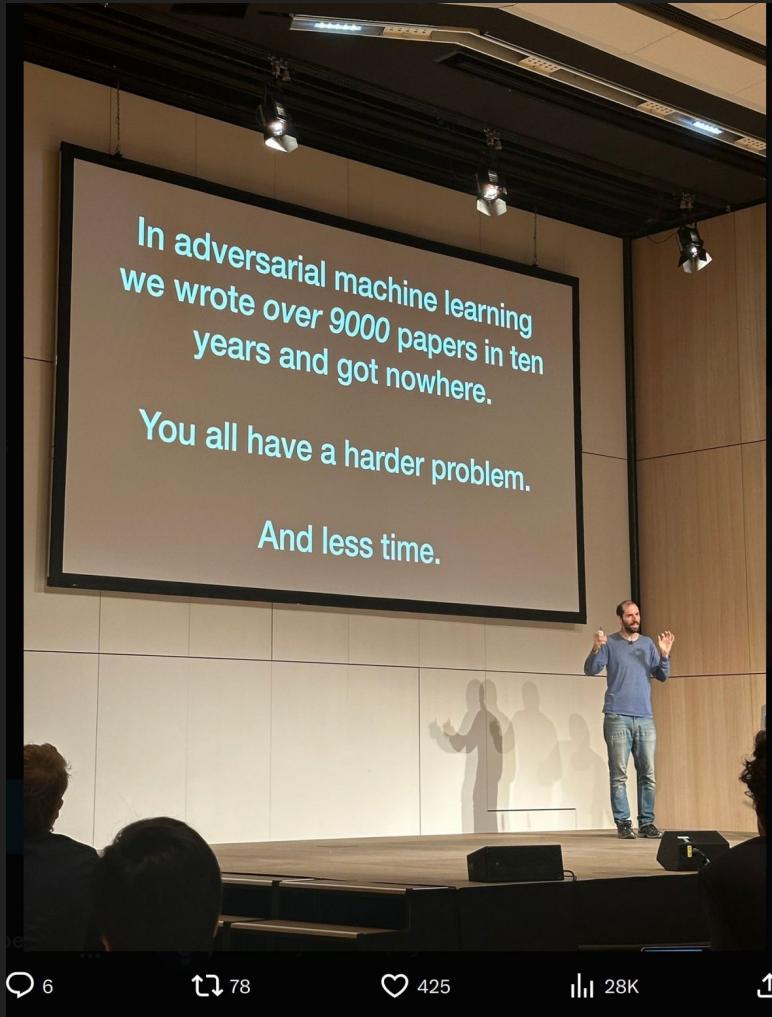
trying to fit human perception
implicit

$$f(X + P) =$$



semantic similarity metrics like (LPIPS)

unsolved mysteries



arXiv > cs > arXiv:1312.6199

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 21 Dec 2013 (v1), last revised 19 Feb 2014 (this version, v4)]

Intriguing properties of neural networks

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Goodfellow, Rob Fergus

adversarial examples: unsolved mysteries

- what are they?
- why are they so close to the image?
- why don't they resemble the target class?
- why do adversarial robustness and accuracy trade-off?
- ...

adversarial examples: unsolved mysteries

- what are they?
- why are they so close to the image?
- why don't they resemble the target class?
- why do **adversarial robustness** and accuracy trade-off?
- ...



informal definition:

not being fooled by the perturbations

formal definition:

Theoretically Principled Trade-off between Robustness and Accuracy

Hongyang Zhang*
CMU & TTIC
hongyangz@cs.cmu.edu

Eric P. Xing
CMU & Petuum Inc.
epxing@cs.cmu.edu

Yaodong Yu†
University of Virginia
yy8ms@virginia.edu

Laurent El Ghaoui
UC Berkeley
elghaoui@berkeley.edu

Jiantao Jiao
UC Berkeley
jiantao@eecs.berkeley.edu

Michael I. Jordan
UC Berkeley
jordan@cs.berkeley.edu

Abstract

We identify a trade-off between robustness and accuracy that serves as a guiding principle in the design of defenses against adversarial examples. Although this problem has been widely studied empirically, much remains unknown concerning the theory underlying this trade-off. In this work, we decompose the prediction error for adversarial examples (robust error) as the sum of the natural (classification) error and boundary error, and provide a differentiable upper bound using the theory of classification-calibrated loss, which is shown to be the tightest possible upper bound uniform over all probability distributions and measurable predictors. Inspired by our theoretical analysis, we also design a new defense method, TRADES, to trade adversarial robustness off against accuracy. Our proposed algorithm performs well on real-world datasets. The methodology is the foundation for a new challenge, the Adversarial Robustness Trade-off Challenge in which researchers can compete to find the best defense.

adversarial examples: unsolved mysteries

- what are they?
- why are they so close to the image?
- why don't they resemble the target class?
- why do adversarial robustness and accuracy trade-off?

adi shamir: “Any attempt to robustify a network by limiting all its directional derivatives will make it harder to train and thus less accurate.”



adversarial examples: unsolved mysteries

- what are they?
- why are they so close to the image?
- why don't they resemble the target class?
- why do adversarial robustness and accuracy trade-off?

adi shamir: “Any attempt to robustify a network by limiting all its directional derivatives will make it harder to train and thus less accurate.”

madry's lab:

- | | |
|------------------------|--|
| a) useless features | not picked up by feature extractor |
| b) robust features | comprehensible to humans, stable under perturbations |
| c) non-robust features | incomprehensible to humans
very predictive but also very brittle
derived from data distribution patterns |



adversarial examples: unsolved mysteries

- what are they?
- why are they so close to the image?
- why don't they resemble the target class?
- why do adversarial robustness and accuracy trade-off?

adi shamir: “Any attempt to robustify a network by limiting all its directional derivatives will make it harder to train and thus less accurate.”

madry's lab:

a) useless features	not picked up by feature extractor
b) robust features	comprehensible to humans, stable under perturbations
c) non-robust features	incomprehensible to humans very predictive but also very brittle derived from data distribution patterns

predictive accuracy without generalizability is useless for downstream tasks.

there is a solution!

Robustness and Accuracy Could Be Reconcilable by (Proper) Definition

Tianyu Pang^{1,2} Min Lin² Xiao Yang¹ Jun Zhu¹ Shuicheng Yan²

Abstract

The trade-off between robustness and accuracy has been widely studied in the adversarial literature. Although still controversial, the prevailing view is that this trade-off is inherent, either empirically or theoretically. Thus, we dig for the origin of this trade-off in adversarial training and find that it may stem from the improperly defined robust error, which imposes an inductive bias of local invariance — an overcorrection towards smoothness. Given this, we advocate employing local equivariance to describe the ideal behavior of a robust model, leading to a self-consistent robust error named SCORE. By definition, SCORE facilitates the reconciliation between robustness and accuracy, while still handling the worst-case uncertainty via robust optimization. By simply substituting KL divergence by variants of distance metrics, SCORE can be efficiently minimized. Empirically, our models achieve top-rank performance on RobustBench under AutoAttack. Besides, SCORE provides instructive insights for explaining the overfitting phenomenon and semantic input gradients observed on robust models.

1. Introduction

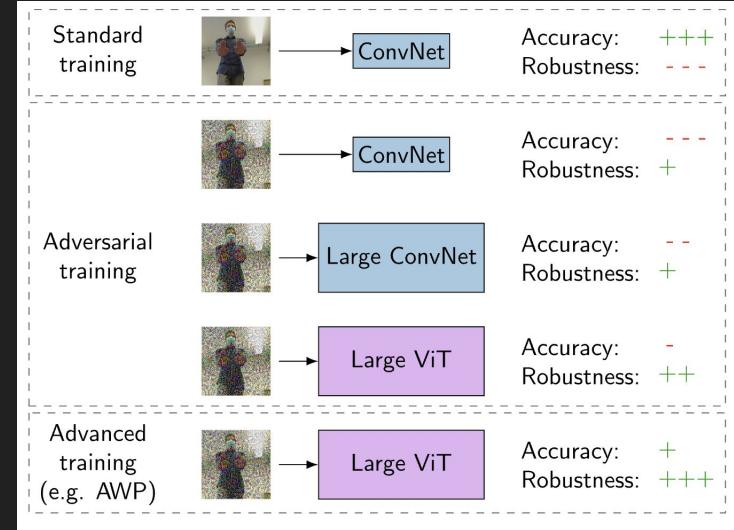
The trade-off between *adversarial* robustness and *clean* accuracy has been widely observed (Schmidt et al., 2018; Su et al., 2018; Zhang et al., 2019; Wang et al., 2020a,b). On some simple cases, this trade-off is even shown to provably exist (Tsipras et al., 2019; Nakkiran, 2019; Raghunathan et al., 2020; Javanmard et al., 2020; Yu et al., 2021). With that, many strategies have been proposed to alleviate this trade-off via, e.g., early-stopping (Rice et al., 2020; Zhang et al., 2020), instance reweighting (Balaji et al., 2019; Zhang et al., 2021a), and exploiting extra data (Alayrac et al., 2019;

Nevertheless, other findings show the opposite. Stutz et al. (2019) and Yang et al. (2020) argue that robustness and accuracy can both be achievable through manifold analysis locally Lipschitz functions. Rozsa et al. (2016) and Gilmer et al. (2018) also reveal that better generalization helps robustness on both toy and large-scale datasets.

Given arguments on both sides, it is much debated whether the robustness-accuracy trade-off is intrinsically there. But before considering its existence, let us first reach a consensus on how a robust model is supposed to behave — the definition of robustness. A most popular one is made by Madry et al. (2018), who define robustness in terms of robust error formulated as a locally maximized loss function. This definition is widely adopted in adversarial training (AT) (Shafahi et al., 2019; Wong et al., 2020). Besides, there are also some other definitions. For example, Szegedy et al. (2014) define robustness as the minimal perturbation required to flip the predicted labels, which is applied in several adversarial attack studies (Moosavi-Dezfooli et al., 2016; Carlini and Wagner, 2017).

We then try to describe how the robustness-accuracy trade-off stems from the previously defined robust error. Recall that in supervised learning, we have a joint data distribution $p_d(x, y)$, and a discriminative model $p_\theta(y|x)$ for the label y , which is conditional on the input x . In standard settings, we obtain an accurate model via minimizing the expected KL divergence between $p_d(y|x)$ and $p_\theta(y|x)$, i.e., the **standard error** $R_{\text{standard}}(\theta)$ (see Eq. (1)) w.r.t. the parameters θ . The optimal solution θ^* satisfies $p_\theta(y|x) = p_d(y|x)$.

In adversarial settings, Madry et al. (2018) propose to minimize the **robust error** $R_{\text{Madry}}(\theta)$ (see Eq. (3)) for training reliable models. Compared to the standard error, the robust error contains an inner maximization problem, finding the point $x' \in B(x)$ that maximizes the KL divergence between $p_d(y|x)$ and $p_\theta(y|x')$, where $B(x)$ is a set of allowed points around x . Minimizing the robust error w.r.t. θ increases or



Lechner, Mathias, et al. "Revisiting the adversarial robustness-accuracy tradeoff in robot learning." IEEE Robotics and Automation Letters 8.3 (2023): 1595-1602.

in conclusion:

adversarial examples are about generalizability, not security.

they help us:

- to augment data such that models learn the right features
- understand the learning process & human-machine vision gap

my progress

steps:

0. literature review

The screenshot shows a code editor with a dark theme. The left sidebar lists files and folders, including 'main', 'research', 'autoencoders.md', 'breaking captcha.md', 'crossmax self ensembling.md', 'deepfake.md', 'dimples.md', 'intriguing.md', 'not bugs features.md', 'perceptual ball.md', 'platon.md', 'transfer to humans.md', and 'unsegment anything.md'. The main pane displays a Python script titled 'unsegment anything.md'.

UAD stage 1: deformation with flow fields

- $\hat{I}^* = \operatorname{argmin}_I L_D + \lambda_C L_C + \lambda_F L_F$
- learn the optimal deformed target-image
- image-deformation-function can be learned through backprop because it's differentiable
- lambda are coefficients of each loss term

where:

- $\hat{I} = D_w(I)$
 - \hat{I} = preturbed / deformed image
 - w = flow field weights that define a delta for all coordinates in image
 - $\hat{I}^{(i+j\Delta t, j+\Delta w)}$ – but in theory any kind of deformation (ie. rotation, translation, scaling, warping) would have been fine
- $L_D = SSIM(\hat{I}, I)$
 - structural similarity index
 - zero loss is achieved quickly, even when there's still room for improvement
- $L_C = \operatorname{lambda}_1 |\operatorname{mathcal}(L)(TV)| + |\operatorname{lambda}_2 |\operatorname{mathcal}(L)_x|_{\operatorname{var}}|$
 - control loss
 - total variation loss to regularize and promote: locally smooth spatial transformation, globally uniform deformation like shifting
 - creates effect of assembling wrapped and shifted images
- $L_F(\hat{I}, I + r^*)$
 - fidelity loss

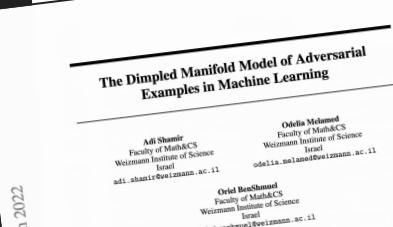
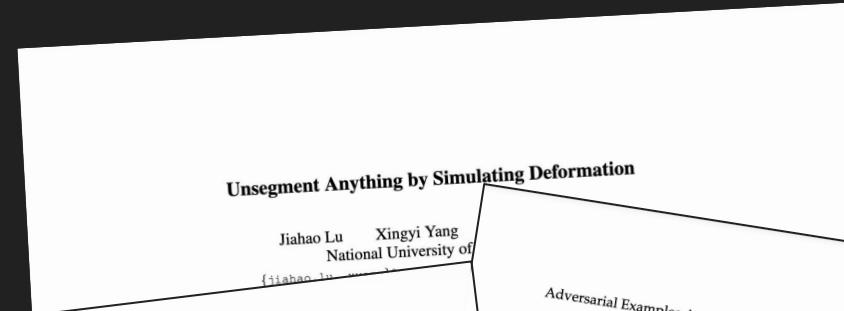
UAD stage 2: feature simulation

- $r^* = \operatorname{arg} \min_r L_P(\hat{I}, I + r) - L_P(I, I + r)$
- reduce the distance of the adversarial perturbation to learned target image
- increase the distance of the original image to the learned target image

evaluation

- measuring efficiency of attacks
- mean Intersection over Union (mIoU) = average attack performance (lower is better)
- attack success rate at IoU < 50% (ASR@50) = number of destroyed output masks (higher is better)
- improvement over previous models, ie. ASR@50 at 69.95 (previously at 65.86)

arXiv:2106.10151v2 [cs.LG] 1 Jun 2022

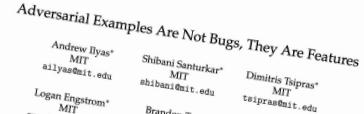


Abstract

The extreme fragility of deep neural networks, when presented with tiny perturbations in their inputs, was independently discovered by several research groups in 2013. However, despite enormous efforts to understand adversarial examples since 2013, no quantitative phenomenon has been established. In this paper, we introduce a new conceptual framework for the understanding boundary between training and testing, which we call the 'Dimpled Manifold Model'. In particular, we demonstrate that training is divided into two randomly oriented phases in a hyperplane: fast-clinging process in which the dimensional manifold oriented decisionary gets very close to the input, and slow-clinging process in which the clinging plane contains all the training samples. Next, there is a typical slow-clinging phase which occurs in the valence of the decision boundary, which moves it to the correct side of the training examples. This model provides a simple explanation for why adversarial noise is more effective than random noise rather than the target class. This noise is also used to show that adversarial noise is adversarially trained, and to show that labeled images can still correctly classify images that undergo the generated dimpling. The use of adversarial training is to design the decision boundaries that are robust to the decision boundary. We discuss the success and demonstrate the different properties of adversarial and off-manifold adversarial examples. We describe the results of numerous experiments which support this new model. The results are both low-dimensional synthetic datasets and high-dimensional natural datasets.

1 Introduction

In 2013 Szegedy et al.(2013) and Biggio et al.(2013) independently demonstrated the surprising fact that well-trained deep neural networks were extremely fragile when presented with tiny perturbations. They found that neural networks trained a lot of images and many attempts to



Adversarial examples have attracted significant attention in machine learning, but the reasons for their robustness and pervasiveness remain unclear. We demonstrate that adversarial examples can be directly attributed to the presence of non-redundant features (detachable pattern in the class distribution). After capturing these features in a theoretical framework, we establish that the empirical evidence in prior datasets, which is often cited as supporting a simple setting where we can rigorously tie the perturbation to the geometry of the data, is misleading. Specifically, we find that adversarial examples are robust and the learned geometry of the data.

Introduction

Pervasive brittleness of deep neural networks, *E.g.,* [\[16\]](#), *Eqn 10b*, *ID3*, *Ath-18*] has attracted significant interest in recent years. Particularly worrisome is the phenomenon of adversarial examples. *I.e.* imperceptibly perturbed natural inputs that induce erroneous predictions. The state-of-the-art classification work has proposed a variety of explanations for this phenomenon, ranging from theoretical arguments based on concentration of measure in high-dimensions (*Eqn 16*) to visualizations that further support this finding (*Eqn 25*). These findings, however, are often unable to fully capture behaviors we observe in real-world work in the field. In this paper, we view adversarial examples as abstractions arising after the network has learned independent features maximizing accuracy (*Eqn 18*–*Eqn 21*) or post-processing of network inputs (*Eqn 22*–*Eqn 24*). From this point of view, it is natural to think of adversarial robustness as a goal that can be pursued independently of maximizing accuracy (*Eqn 18*–*Eqn 21*). Specifically, we propose a new perspective on the phenomenon of adversarial examples. In contrast to prior work, we view adversarial vulnerability as a fundamental consequence of the dominant training paradigm: a direct result of our model's sensitivity to well-generalizing features in the data. By train our model to only maximize (distributional) accuracy. Consequently, classifiers learn signals that are so generic, so, even though they look incomprehensible to humans. After all, the world and Mt. datasets do admit highly predictive yet imperceptible features. We posit that

steps:

0. literature review

1. getting inspired by hCaptcha perturbations

- building an offline CAPTCHA clone
- rendering hCAPTCHA masks

blackbox, not model specific
just manipulates contours and colors

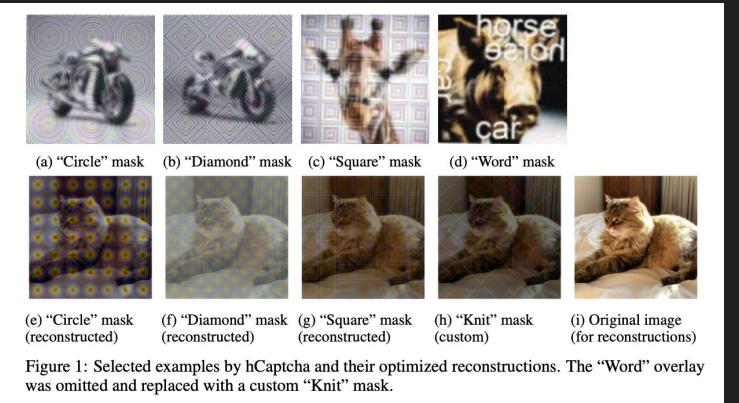


Figure 1: Selected examples by hCaptcha and their optimized reconstructions. The "Word" overlay was omitted and replaced with a custom "Knit" mask.

solve me if you can.
accuracy: 0 / 0

Select all images with hydrants

Please click each image containing a chicken on a tree

If there are None, click Skip

EN

Scanning 2/8

steps:

0. literature review

1. getting inspired by hCaptcha perturbations
2. selecting the best performing models

- comparing models by family
- comparing zero shot benchmark leaderboards
 - ie. *robustbench*
- fine tuning the best performing model

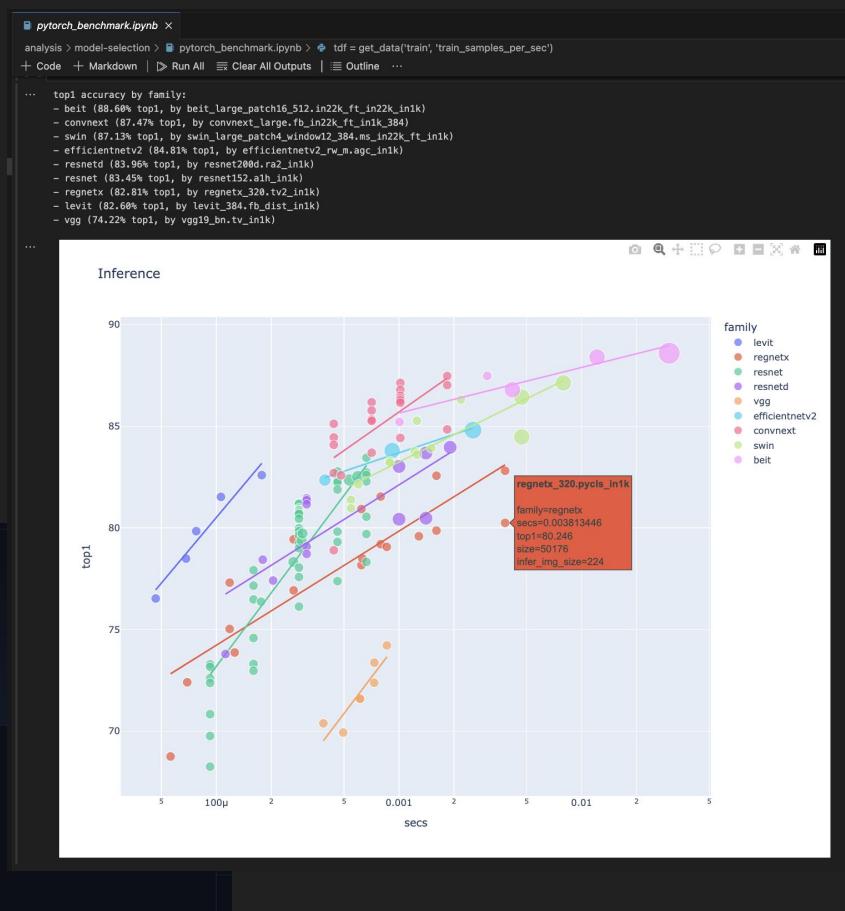
sueszli/robustified_clip_vit

Image Classification License: apache-2.0

Model card Files and versions Community Settings

robustified CLIP ViT-L/14@336px using hcaptcha masks (geometric image overlays) on visual-layer/imagenet-1k-vl-enriched.

for source code and example usage see: <https://github.com/ETH-DISCO/advx-bench>



steps:

0. literature review
1. getting inspired by hCaptcha perturbations
2. selecting the best performing models
3. designing experiments

- imagenet + hcaptcha perturbations
- hyperparameter optimization (opacity, density, fgsm epsilon)
- perceptual quality metrics to avoid transfer to humans

$$\text{final score} = \text{quality} - (\text{adv acc rank} - \text{acc rank})$$

Where:

$$\text{quality} = 0.15 \cdot \text{cs} + 0.25 \cdot \text{psnr}_n + 0.35 \cdot \text{ssim} + 0.25 \cdot (1 - \text{lips}_n)$$

And:

$$\text{psnr}_n = \text{clip}\left(\frac{\text{psnr} - 30}{20}, 0, 1\right)$$

$$\text{lips}_n = \text{clip}(\text{lips}, 0, 1)$$



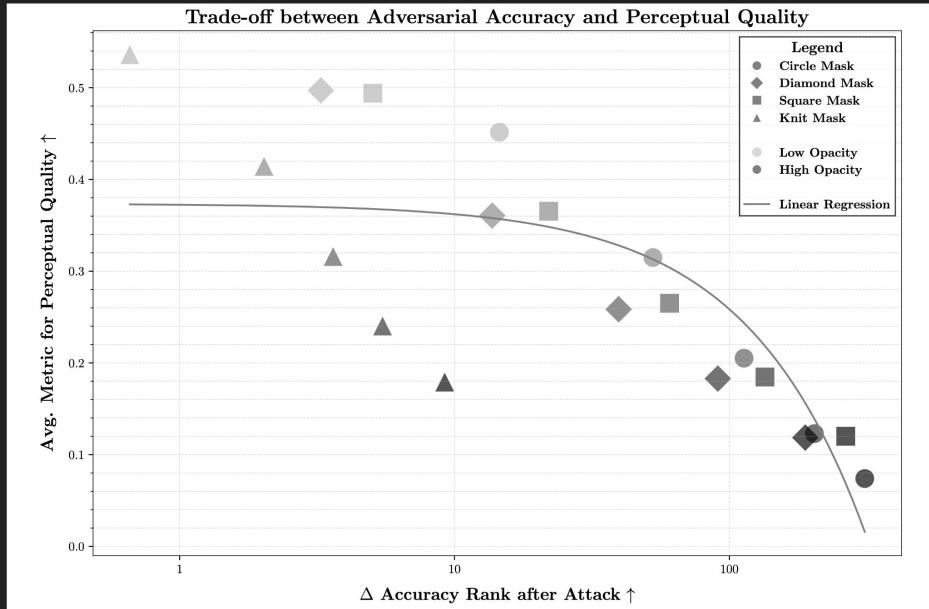
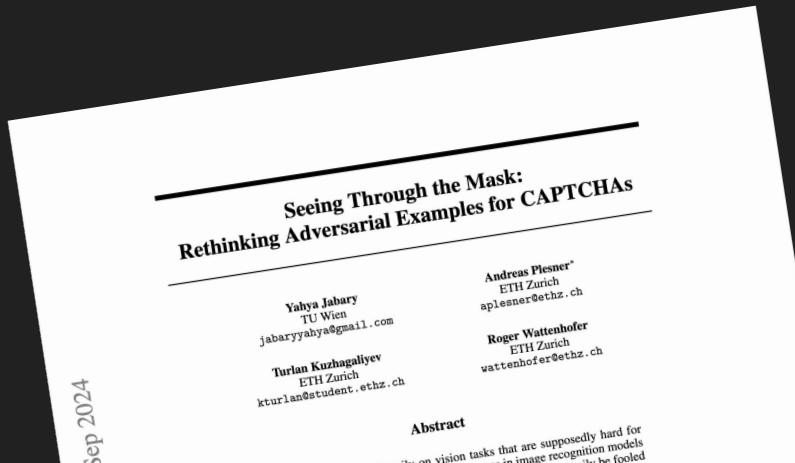
Bartoldson, Brian R., et al. "Adversarial Robustness Limits via Scaling-Law and Human-Aligned Studies." arXiv preprint arXiv:2404.09349 (2024).

steps:

0. literature review
 1. getting inspired by hCaptcha perturbations
 2. selecting the best performing models
 3. designing experiments
 4. **benchmarking**
-
- 2.0 weeks of cluster debugging & writing docs (*see: @ETH-DISCO/cLuster-tutorial*)
 - 1.0 weeks of hyperparam tuning
 - 0.5 weeks of experimentation

steps:

0. literature review
1. getting inspired by hCaptcha perturbations
2. selecting the best performing models
3. designing experiments
4. benchmarking
5. submitting results as workshop paper

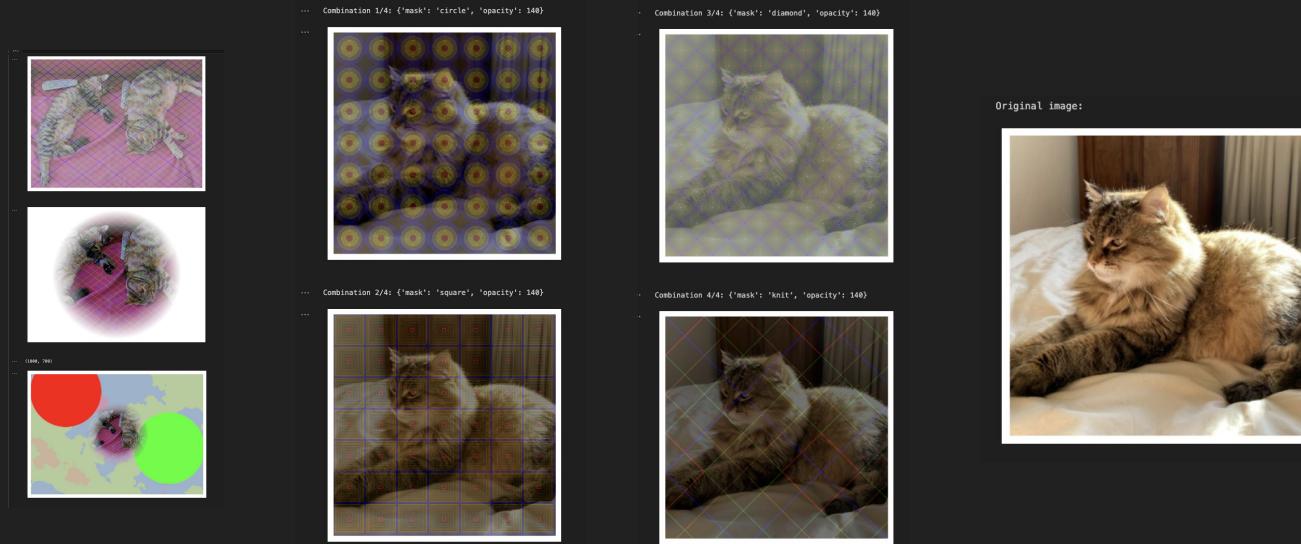


what's next?

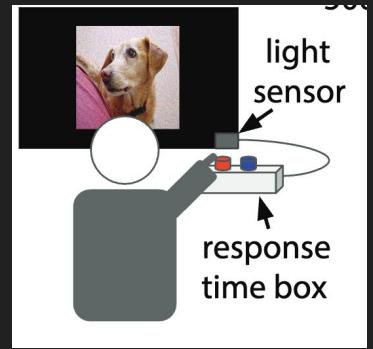
i'd appreciate your advice!

studying geometric distortions of hCaptcha masks

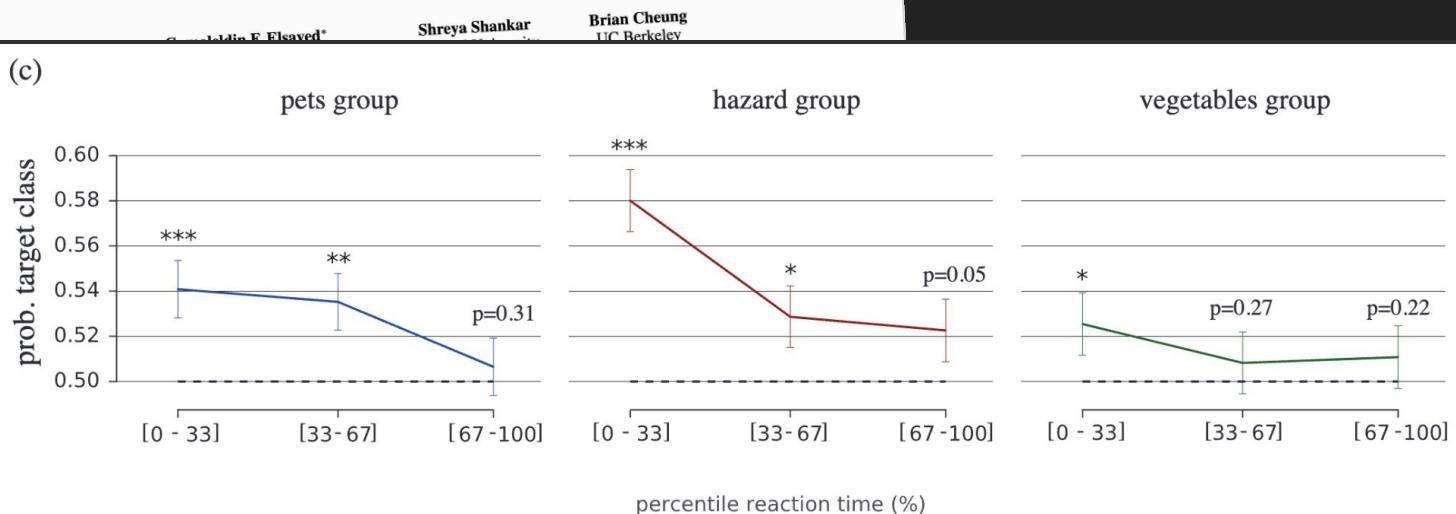
- why were they introduced by this company?
- what makes them so effective?
can we look into intermediary layers to understand?



studying survival-optimized features of human vision



Adversarial Examples that Fool both Computer Vision and Time-Limited Humans



usually an image formed by making small perturbations to the input image. Constructing adversarial examples [13, 24, 27, 33, 39] rely on access to both the architecture and the parameters of the model to perform gradient-based optimization on the input. Without similar access to the model, our approach does not seem applicable to constructing adversarial examples for humans.

self-ensembling

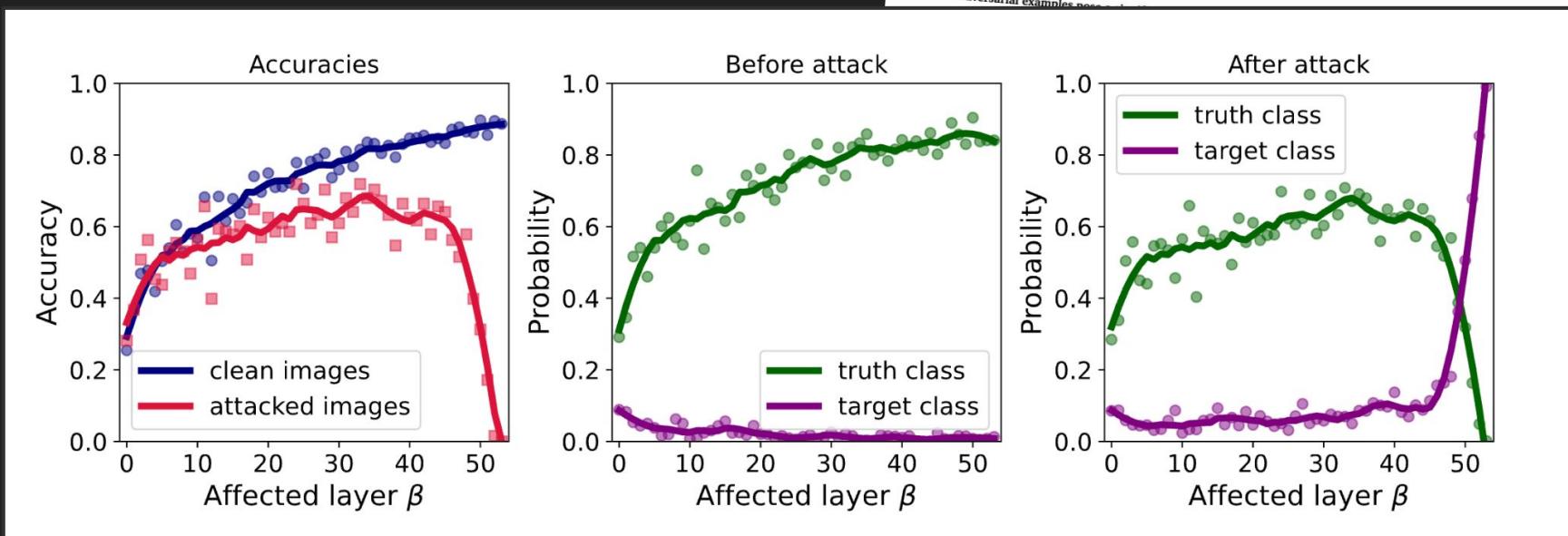
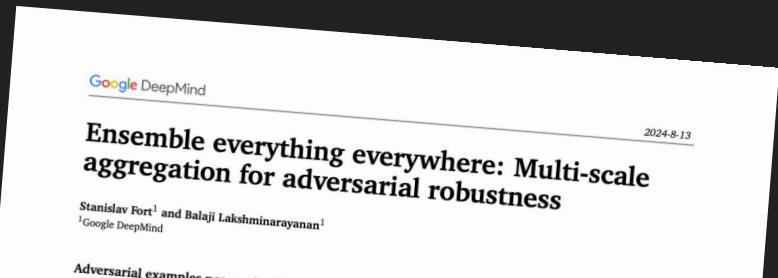


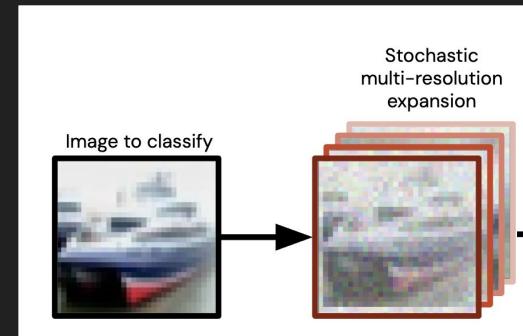
Figure 1 | We use a multi-resolution decomposition (a) of an input image and a partial decorrelation of predictions of intermediate layers (b) to build a classifier (c) that has, by default, adversarial robustness comparable or exceeding state-of-the-art (f), even without any adversarial training. Optimizing inputs against it leads to interpretable changes (d) and images generated from scratch (e).

down sampling

classifier makes a decision about all N versions at once

mimicking nature:

- 1) random noise
- 2) a random jitter in both axes (*microsaccades*)
- 3) a small random change in contrast (*cone cells*)
- 4) a small random color-grayscale shift (*cone cells*)



down sampling

classifier makes a decision about

mimicking nature:

- 1) random noise
- 2) a random jitter in both axes
- 3) a small random change in contrast (*cone cells*)
- 4) a small random color-grayscale shift (*cone cells*)

my idea: combining with NeRFs (Neural Radiance Fields)

<https://www.matthewtancik.com/nerf>

capturing all angles, distances, under various blurs, rotations, illuminations, contrasts while semantic content is preserved.



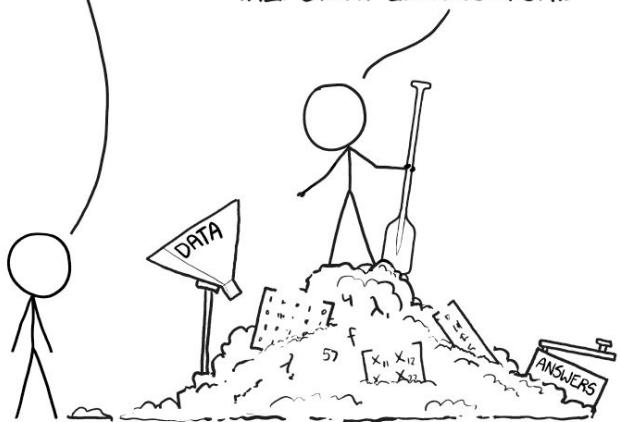
thank you for listening!

THIS IS YOUR MACHINE LEARNING SYSTEM?

YUP! YOU POUR THE DATA INTO THIS BIG
PILE OF LINEAR ALGEBRA, THEN COLLECT
THE ANSWERS ON THE OTHER SIDE.

WHAT IF THE ANSWERS ARE WRONG?

JUST STIR THE PILE UNTIL
THEY START LOOKING RIGHT.



thank you for listening!

Q&A / discussion