Adversarial examples in deep neural networks

Ginevra Carbone

5 giugno 2019

What we'll talk about

- Tricking Neural Networks
  - What do we mean by adversarial examples?
  - Why are these adversaries relevant?
- Manifold Hypothesis
- Enhanced problems
- Attacks
- Defences

Tricking Neural Networks

What do we mean by adversarial examples?

In the context of deep neural networks, an adversarial example is a **small perturbation of the input** leading even a very robust model to an incorrect classification.

Defense	Target	No Attack	FGSM	PGD	C&W	**SPT**
No Defense	C_p	99.02%	9.28%	**0.00%**	**0.00%**	9.74%
	C_{a0}	98.83%	5.55%	**0.00%**	**0.00%**	9.75%
	C_{a1}	98.73%	7.18%	0.03%	**0.00%**	10.52%
	C_{a2}	98.33%	8.25%	0.09%	**0.00%**	8.93%
	C_{a3}	98.58%	11.44%	**0.00%**	**0.00%**	9.91%
PGD Adv. Tr. (Tr. Acc) $\epsilon = 0.3$ $\alpha = 0.01$	C_p	98.08%	93.24%	88.14%	32.00%	**13.47%**
	C_{a0}	97.65%	84.51%	68.07%	**5.00%**	5.10%
	C_{a1}	98.13%	89.01%	73.92%	6.00%	**4.39%**
	C_{a2}	98.20%	87.78%	73.16%	**4.00%**	8.98%
	C_{a3}	96.90%	92.54%	87.15%	45.00%	**7.32%**

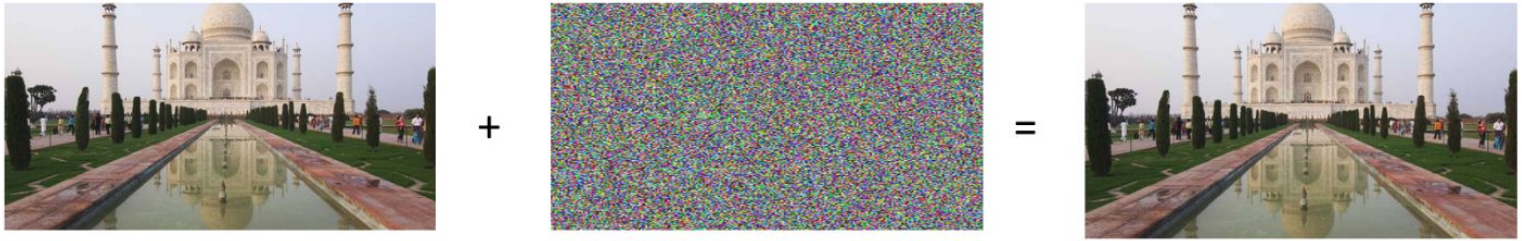
Peng et al., 2018

Tricking Neural Networks

What do we mean by adversarial examples?

Talking about **image classification** in particular, we can generally distinguish between two kinds of adversarial examples:

- **Perturbation images**: adding some noise to a given image can produce a perturbed image which is visually indistinguishable from the original one, but misclassified;



Classified as “Taj Mahal”

Small Perturbation

Classified as “Not Taj Mahal”

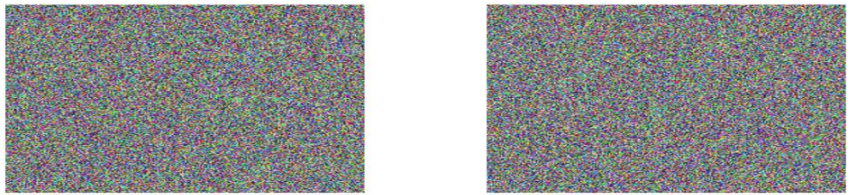
Dube et al., 2018

Tricking Neural Networks

What do we mean by adversarial examples?

Talking about **image classification** in particular, we can generally distinguish between two kinds of adversarial examples:

- **Perturbation images**: adding some noise to a given image can produce a perturbed image which is visually indistinguishable from the original one, but misclassified;
- **Fake unrecognizable images**: deep networks can classify with high confidence images which are visually unrecognizable.



Fake Unrecognizable Images
Classified as “Taj Mahal”

Dube et al., 2018

Tricking Neural Networks

Why are these adversaries relevant?

The vulnerability of deep learning models to adversarial attacks has received growing attention in recent years, since it shows how the accuracy of a network alone is not a good measure of **robustness**.

These perturbations are often not perceivable by human beings, so the defense against them is crucial for **security** reasons, especially in computer vision algorithms. Just think about wrong classification of road signs by self driving cars, wrong recognition of medical images, faces or airport scanning images.

Eykholt et al., 2018

Sharif et al., 2016

Tricking Neural Networks

Why are these adversaries relevant?

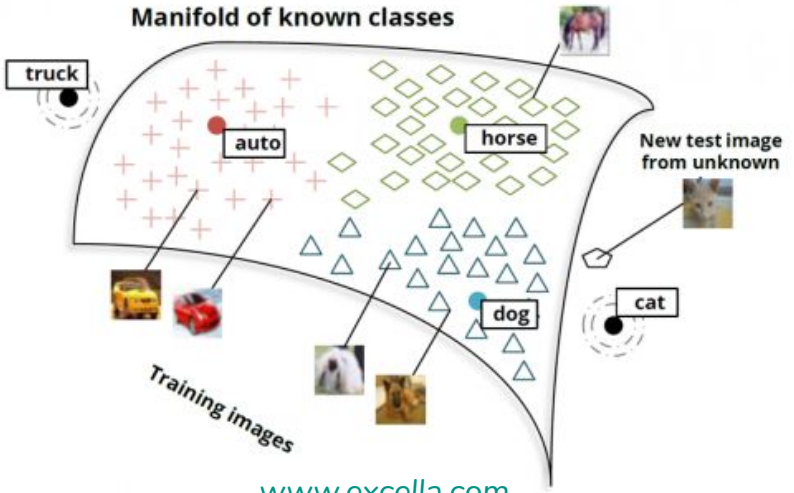
Another problem is the one of **transferability** of the attacks: an attack generated for a particular classification model is often misclassified by a different model, with a completely different architecture or a different training set (Goodfellow et al., 2015).

- Tricking Neural Networks
- Manifold Hypothesis
  - Decision boundary in a high dimensional space
  - Defining adversarial examples and robust classifiers
- Enhanced problems
- Attacks
- Defences

Manifold hypothesis

Decision boundary in a high dimensional space

The robustness of classifiers is strongly related to the **geometry of the learned decision boundaries**. In particular, adversarially perturbed points are always extremely close to these surfaces (Dube et al., 2018).

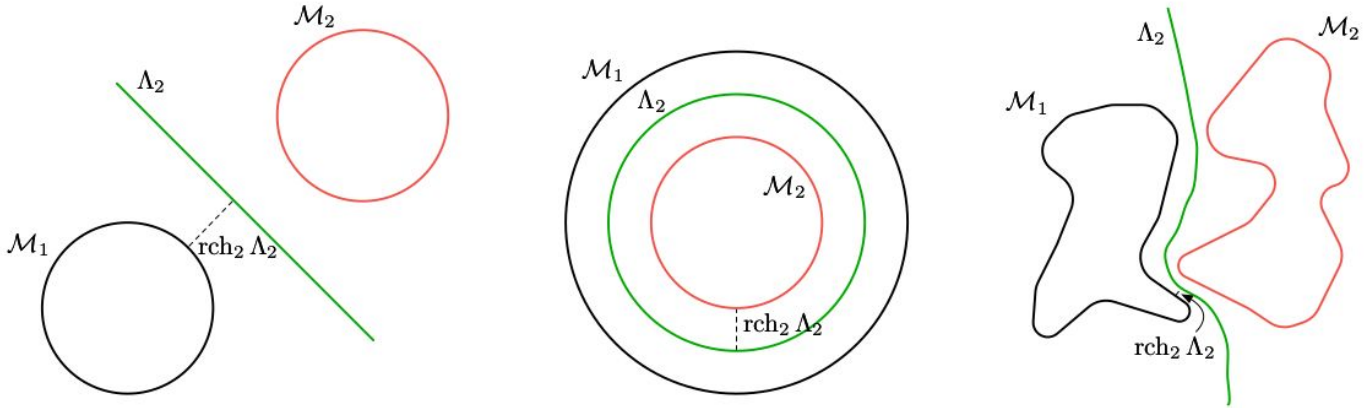


The idea behind the Manifold Hypothesis is to model data as being sampled from **low-dimensional manifolds**, corresponding to the classification regions, embedded in a **high-dimensional space** $\mathbb{R}^d$ and to represent the decision boundaries as **hypersurfaces** of the space (Khoury et al., 2018). This approach allows to reduce the high-dimensionality of the input space.

Manifold hypothesis

Decision boundary in a high dimensional space

Let K be the total number of classes and d the dimensionality of points.
We can model **classification regions** as manifolds $\mathcal{M}_1, ..., \mathcal{M}_K \subseteq \mathbb{R}^d$ and the space of the inputs as the **data manifold** $\mathcal{M} := \cup_{1 \leq j \leq K} \mathcal{M}_j \subseteq \mathbb{R}^d$, whose dimension is a certain $k \leq d$.
In order to learn **robust decision boundaries**, our multiclass classifier $f : \mathbb{R}^d \to \mathbb{R}^K$ has to be robust in a ϵ-neighbour around the data manifold.

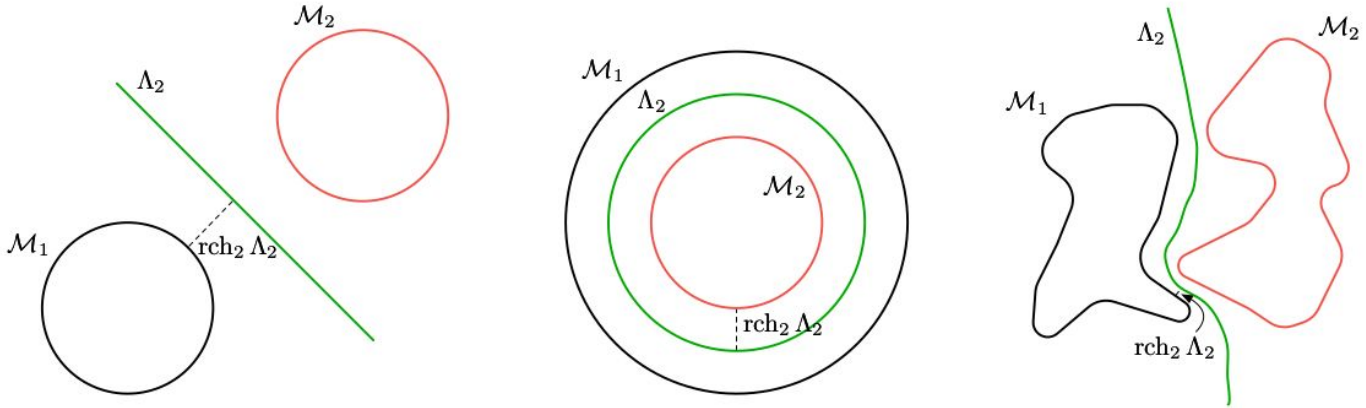


Khuory et al., 2018

Manifold hypothesis

Decision boundary in a high dimensional space

The **decision boundary** Λ_p of $\mathcal{M}$, in the norm $||\cdot||_p$, is the set of points having two or more closest points on distinct manifolds. Intuitively, the decision boundary generalizes the notion of *maximum margin* and separates the class manifolds when they do not intersect (Khuory et al, 2018).



Khuory et al., 2018

Manifold hypothesis

Defining adversarial examples and robust classifiers

The ϵ-neighbour of $\mathcal{M}$ is the set of points whose distance from $\mathcal{M}$ is the p-norm is lower than ϵ

$$\mathcal{M}^{\epsilon,p} = \{x \in \mathbb{R}^d : inf_{x' \in \mathcal{M}} ||x - x'||_p \leq \epsilon\}.$$

An ϵ-**adversarial example** is a point $x \in \mathcal{M}_i^{\epsilon,p}$ such that $f(x) \neq i$, so f is said to be **robust** to ϵ-adversarial examples if it correctly classifies all of $\mathcal{M}^{\epsilon,p}$.

What we'll talk about

- Tricking Neural Networks
- Manifold Hypothesis
- Enhanced problems
  - High codimension
  - Robustness under different p-norms
  - Complexity of the data manifold surface
- Attacks
- Defences

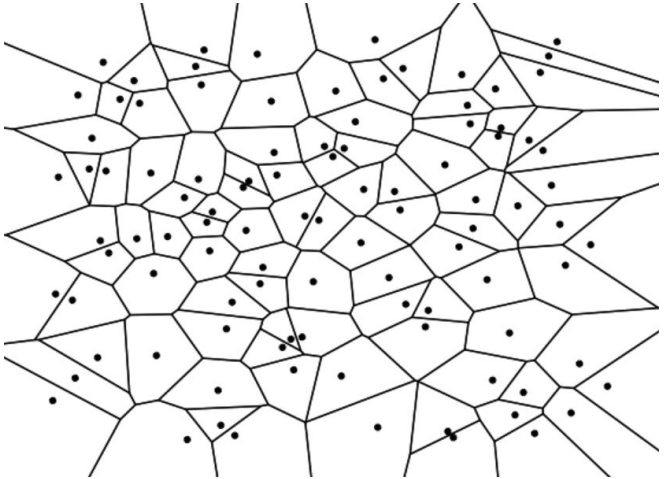
Enhanced problems

High codimension

Khuory et al. (2018) first highlighted the role of codimension in adversarial examples. They showed that adversarial perturbations arise in the **directions normal to the data manifold**, so as the codimension increases there is an increasing number of directions in which to build adversarial perturbations.

aaroncheng.me

This is also the reason why **k-Nearest Neighbor classifiers** are more robust in high codimensions, due to geometric properties of their decision boundary away from data: the *Voronoi cells* of the samples are elongated in the directions normal to the data manifold when the sample is dense (Dey et al., 2007).



Enhanced problems

Robustness under different p-norms

Robustness conditions change under different p-norms: **no single decision boundary can be optimally robust in all norms** (Khuory et al., 2018).
This means that if a classifier is trained to be robust under $||\cdot||_\infty$ norm, poor robustness under the $||\cdot||_2$ norm should be expected. This kind of phenomenon was previously attributed to *overfitting* instead.

Enhanced problems

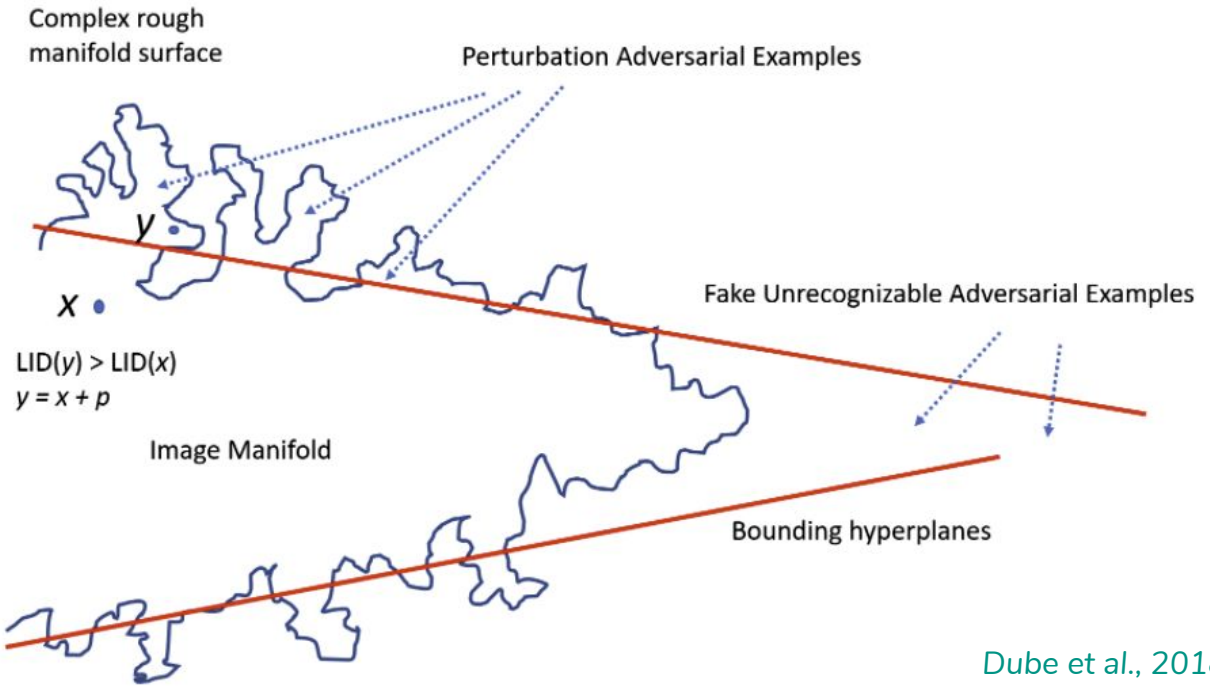
Complexity of the data manifold surface

A characterization of adversarial perturbations in terms of the complexity of the *adversarial regions* where they locate is given by the concept of **local intrinsic dimensionality** (LID).

The *LID of each point can be estimated* using the distances to its k-nearest neighbors within the sample (Amsaleg et al, 2016). Through these estimates, Ma et al. (2018) and Dube et al. (2018) empirically showed that the **transition from normal example to adversarial example** typically follows directions in which the **estimated LID value increases.**

Enhanced problems

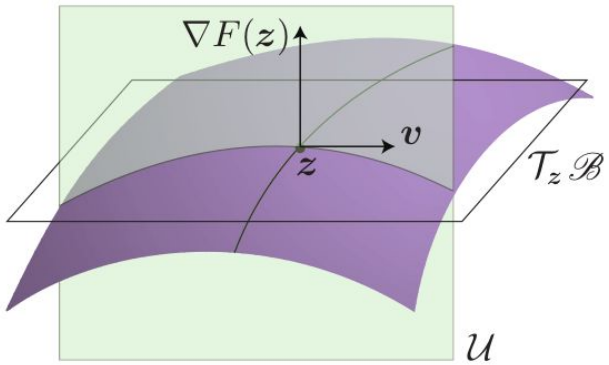
Complexity of the data manifold surface



Dube et al., 2018

Enhanced problems

Complexity of the data manifold surface



Fawzi et al., 2018

Similarly, Fawzi et al. (2018) examined sensitivity to adversarial examples in relation to the **curvature of decision boundaries**. Their results show that when a decision boundary has a small curvature, the classifier is more robust to adversarial examples.

Enhanced problems

Complexity of the data manifold surface

The better a model is able to approximate the ground truth data manifold, the less it suffers from adversarial examples. Unfortunately the surfaces are **too complex** to be exactly learned by **finitely many neurons**, so the vulnerability to adversarial attack is inevitable as the data scales in both size and intrinsic dimensionality, regardless of the nature of the data (Amsaleg et al, 2018).

Exact characterization of the surfaces of image manifolds remains **open problem.**

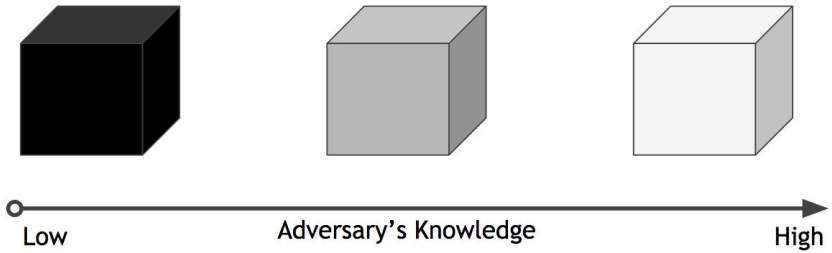
- Tricking Neural Networks
- Manifold Hypothesis
- Enhanced problems
- Attacks
  - White-box vs black-box
  - Targeted vs non-targeted
  - Gradient based attacks
    - FGSM
    - BIM
    - Deepfool
    - Carlini & Wagner
    - Comparison
- Defences

Attacks

White-box vs black-box

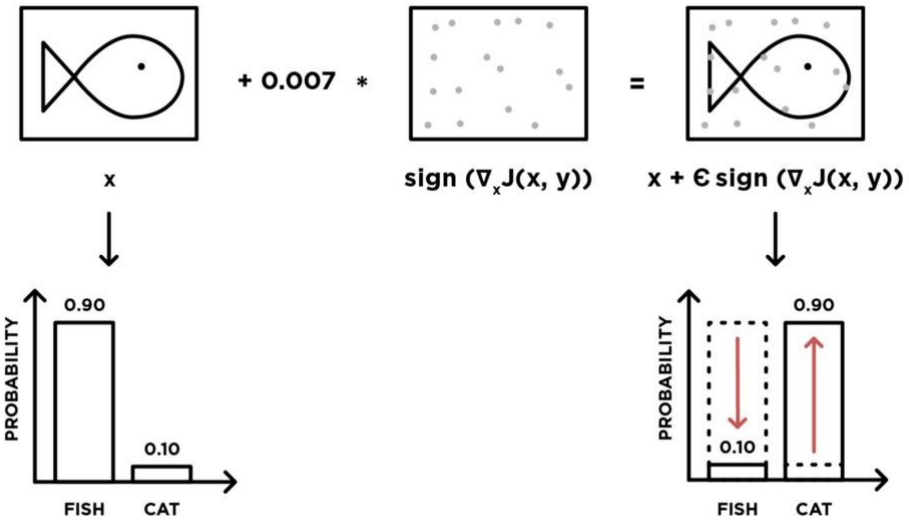
We can distinguish between two types of attacks depending on the *knowledge of the attacker* about the structure of the model:

- **white-box attacks** have full access to the weights, the number of layers, the architecture and the hyperparameters of the classifier;
- **black-box** attacks can be based on one of these two properties:
  - **transferability across policies**, when they know the training architecture and hyperparameters, but not its random initialization;
  - **transferability across algorithms,** when only know the output of the model (label or confidence score).



Attacks

Targeted vs non-targeted



- **targeted attacks** misguide deep neural networks to a specific class, usually in a multi-class classification problem
- **non-targeted attacks** do not assign a specific class to the output and can be obtained by:
  - generating several targeted attacks and choosing the one with the smallest perturbation
  - minimizing the probability of the correct class

Attacks

Gradient based attacks

Many adversarial attacks are related to the principle of the gradient descent: instead of minimizing the loss on a given training set, the idea is that of **increasing the loss** with respect to the available data for a given model with **fixed weights.**

The goal of **gradient descent** algorithm is that of finding the *optimal model parameters* θ for the classifier $f(x, \theta)$, by updating them at each step of the training algorithm. In order to do that, we choose a **loss function** ℓ**,** measuring the inconsistency between the predicted label $\hat{y} = f(x, \theta)$ and the true label y.

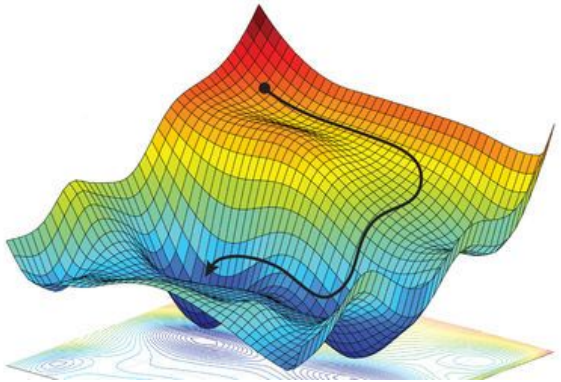
$$\ell : \mathbb{R}^K \times \mathcal{Y} \longrightarrow \mathbb{R}$$
$$(f(x, \theta), y) \mapsto \ell(f(x, \theta), y)$$

Attacks

Gradient based attacks

Gradient descent **iteratively minimizes the objective function** ℓ by starting from a random point θ and taking small steps in the direction of its gradient:

$$\min_\theta \ \ell(f(x,\theta),y)$$
$$\theta \leftarrow \theta - \alpha \nabla_\theta \ell(f(x,\theta),y)$$



www.sciencemag.org

The **learning rate** α is the hyperparameter responsible for the amount of change (speed of learning) of the model at each step. For a sufficiently small step size and a non-pathological function this will always converge, at least to a local minimum.

Attacks

Gradient based attacks

Given the network parameters θ and the strength $\epsilon > 0$ of the adversary, an adversarial example can be expressed, *in terms of the loss function*, as

$$\tilde{x} := argmax_{\tilde{x}:||\tilde{x}-x||_p \leq \epsilon} \ell(f(\tilde{x}, \theta), y).$$

Basically, this means going from a **gradient descent on the weights** of the network to a **gradient ascent on the data.**

Attacks

Fast gradient sign method (FGSM)

This method aims at controlling the l_p norm of the adversarial perturbation by **adding noise in the direction of the gradient** of the cost function with respect to the data (Goodfellow et al., 2015).
For the l_∞ norm, a perturbation

$$\eta = \epsilon \cdot sign(\nabla_x \ell(f(x, \theta), y))$$

with **attack strength** ϵ generates the adversarial example $\tilde{x} = x + \eta$.

In the case of l_1, l_2 norms the perturbation is given by $\eta = \epsilon \cdot \frac{\nabla_x \ell(f(x,\theta),y)}{||\nabla_x \ell(f(x,\theta),y)||_p}$.

> The choice of the parameter ϵ influences the **strength of the attack** since if it is too small, the perturbed point may not have a different label from the true one.

Attacks

Basic iterative method (BIM)

The **Basic Iterative Method** by Kurakin et al. (2017) is an iterative extension of FGSM and it is targeted towards the least likely class of the prediction.
At each step all perturbed *pixel values are clipped,* in order to avoid large changes in the adversarial image

$$x_0 = x$$
$$x_{n+1} = Clip_{x,\xi}\{x_n + \epsilon \cdot sign(\nabla_x \ell(f(x,\theta),y))\}$$

The **clipping function** attacks with a fixed perturbation ξ and then projects the result back onto the ϵ-ball centered in the original input:

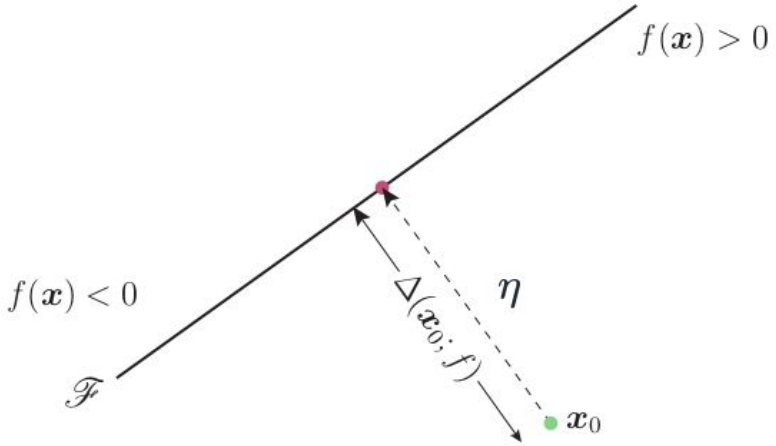
$$Clip_{x,\xi}\{\tilde{x}\} = min\{255, x+\xi, max\{0, x-\epsilon, \tilde{x}\}\}$$

The attack was originally designed for l_∞-norm perturbations, but can easily be extended to other norms.

Attacks
Deepfool

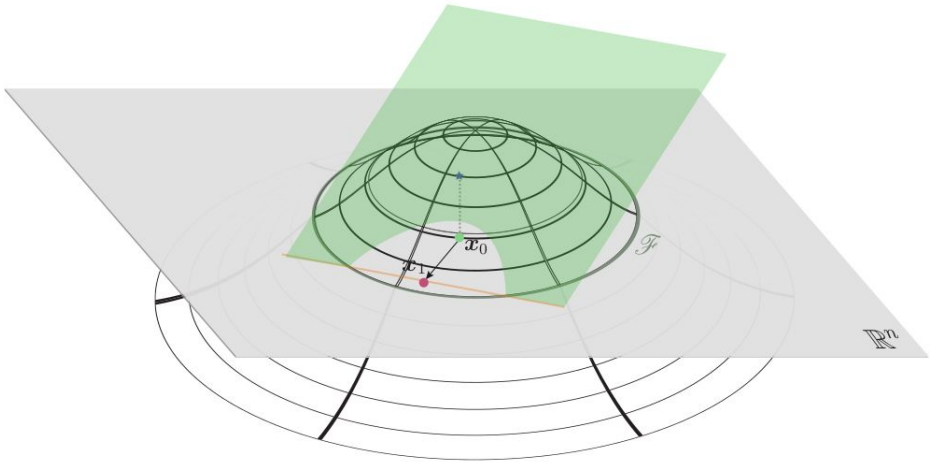
Deepfool untargeted attack, proposed by Moosavi-Dezfooli et al. (2016), finds the *nearest decision boundary* for each given input x, in the l_2 norm, and pushes it over the boundary.
This is very simple in the case of a **linear binary classifier**, since the minimal perturbation for a separating hyperplane $w^T x + b = 0$ is simply the orthogonal projection onto it

$$\eta = -\frac{f(x)}{||w||^2} w.$$



Moosavi-Dezfooli et al., 2016

Attacks
Deepfool



Moosavi-Dezfooli et al., 2016

For a general **binary non-linear** (differentiable) **classifier** this method iteratively linearizes the classifier f around the input point until it produces a misclassification

$$argmin_{\eta_i} ||\eta_i||_2$$

$$f(x_i) + \nabla f(x_i)^T \eta_i = 0$$

Attacks

Carlini & Wagner

Carlini-Wagner attack (Carlini & Wagner, 2016) directly optimizes in favour of the minimal distance from the original example, under the constraint of the example being misclassified by the original model.

The **optimization problem** is

$$min_{\delta}||\delta||_p + c \cdot \ell(x+\delta)$$
$$x + \delta \in [0,1]^d$$

The box constraint $[0,1]^d$ is encoded through a change of variable by expressing δ in terms of the hyperbolic tangent.

Attacks

Carlini & Wagner

The **loss function** for this attack represents the distance between the target class t and the most-likely class:

$$\ell(\tilde{x}) = max\{max[Z(\tilde{x})_i : i \neq t] - Z(\tilde{x})_t, -k\}.$$

Here $Z(\tilde{x})$ denotes the *logits layer output* on the adversarial input, t represents the *target misclassification*, while the constant k controls the **confidence score:** when k is small, the generated $\tilde{x}$ is a low confidence adversarial example.

The attacks is pretty costly, since it involves solving a nontrivial optimization problem, but it is *one of the most effective* among those currently known.

Attacks Methods	Adversarial Falsification	Adversary's Knowledge	Adversarial Specificity	Perturbation Scope	Perturbation Limitation	Attack Frequency	Perturbation Measurement	Datasets	Architectures
L-BFGS Attack [19]	False Negative	White-Box	Targeted	Individual	Optimized	Iterative	ℓ_2	MNIST, ImageNet, YoutubeDataset	AlexNet, QuocNet
Fast Gradient Sign Method (FGSM) [55]	False Negative	White-Box	Non-Targeted	Individual	N/A	One-time	element-wise	MNIST, ImageNet	GoogLeNet
Basic Iterative Method (BIM) and Iterative Least-Likely Class (ILLC) [20]	False Negative	White-Box	Non-Targeted	Individual	N/A	Iterative	element-wise	ImageNet	GoogLeNet
Jacobian-based Saliency Map Attack (JSMA) [82]	False Negative	White-Box	Targeted	Individual	Optimized	Iterative	ℓ_2	MNIST	LeNet
DeepFool [83]	False Negative	White-Box	Non-Targeted	Individual	Optimized	Iterative	$\ell_p(p \in 1, \infty)$	MNIST, CIFAR10, ImageNet	LeNet, CaffeNet, GoogLeNet
CPPN EA Fool [84]	False Positive	White-Box	Non-Targeted	Individual	N/A	Iterative	N/A	MNIST, ImageNet	LeNet, AlexNet
C&W's Attack [85]	False Negative	White-Box	Targeted	Individual	Optimized	Iterative	$\ell_1, \ell_2, \ell_\infty$	MNIST, CIFAR10, ImageNet	GoogLeNet
Zeroth Order Optimization [78]	False Negative	Black-Box	Targeted & Non-Targeted	Individual	Optimized	Iterative	ℓ_2	CIFAR10, ImageNet	GoogLeNet
Universal Perturbation [86]	False Negative	White-Box	Non-Targeted	Universal	Optimized	Iterative	$\ell_p(p \in 1, \infty)$	ImageNet	CaffeNet, VGG, GoogLeNet, ResNet
One Pixel Attack [87]	False Negative	Black-Box	Targeted & Non-Targeted	Individual	Constraint	Iterative	ℓ_0	CIFAR10	VGG, AllConv, NiN
Feature Adversary [88]	False Negative	White-Box	Targeted	Individual	Constraint	Iterative	ℓ_2	ImageNet	CaffeNet, VGG, AlexNet, GoogLeNet
Hot/Cold [81]	False Negative	White-Box	Targeted	Individual	Optimized & Constraint	One-time	PASS	MNIST, ImageNet	LeNet, GoogLeNet, ResNet
Natural GAN [79]	False Negative	Black-Box	Non-targeted	Individual	Optimized	Iterative	ℓ_2	MNIST, LSUN, SNLI	LeNet, LSTM, TreeLSTM
Model-based Ensembling Attack [89]	False Negative	White-Box	Targeted & Non-Targeted	Individual	Constraint	Iterative	ℓ_2	ImageNet	VGG, GoogLeNet, ResNet
Ground-Truth Attack [90]	False Negative	White-Box	Targeted	Individual	Optimized	Iterative	ℓ_1, ℓ_∞	MNIST	3-layer FC

Yuan et al., 2018

- Tricking Neural Networks
- Manifold Hypothesis
- Enhanced problems
- Attacks
- Defences
  - Adversarial training
  - AdvGAN
  - Lipschitz regularization

Defences

“No method of defending against adversarial examples is yet completely satisfactory. This remains a rapidly evolving research area.”

A. Kurakin, I. Goodfellow, S. Bengio et al., 2018

Defences

Adversarial training

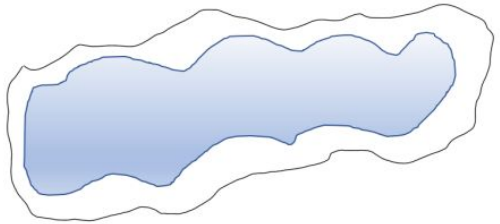
The process of training a classifier by including adversarial examples in the training set is called **adversarial training** (Goodfellow et al., 2015) and it's probably the most popular approach to obtain robust models, because it basically allows to *convert any attack into a defense.*

The biggest limitation of adversarial training is that it tends to **overfit the specific attack** used at training time.

Defences

Adversarial training

The interesting thing is noticing how this approach differs from traditional **data augmentation***:* instead of augmenting data with transformations which are *expected to occur* in the test set (translations, rotations, etc.), only the *most unlikely* examples are added.



Dube et al., 2018

This method corresponds to a **dilation of the manifold:** adversarial examples are learned in a halo around the surface, which makes the manifold smoother (Dube et al., 2018)

Defences

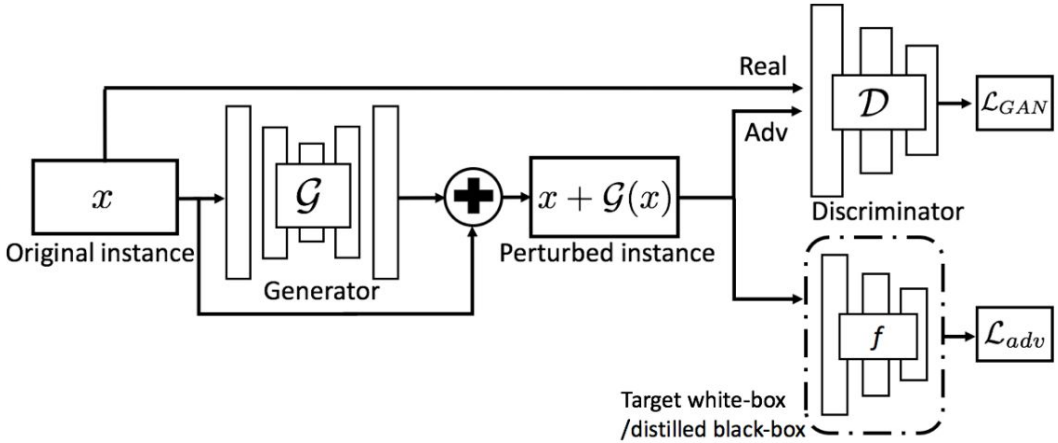
AdvGAN

Xiao et al. in 2018 proposed crafting adversarial examples using GANs: the **AdvGAN** model

As for any GAN algorithm, this model contains:

- the **generator** $\mathcal{G}$, that takes the original instance x and generates a perturbation $\mathcal{G}(x)$;
- the **discriminator** $\mathcal{D}$, that is used to distinguish the input point x from its generated perturbation.

Additionally the **target model** f is in play.



Xiao et al., 2018

Defences

AdvGAN

The loss function for AdvGAN is expressed in terms of both:

- the **adversarial loss**

  $$\mathcal{L}_{adv} = \mathbb{E}_x \ell(f_\theta(x + \mathcal{G}_\phi(x)), y)$$

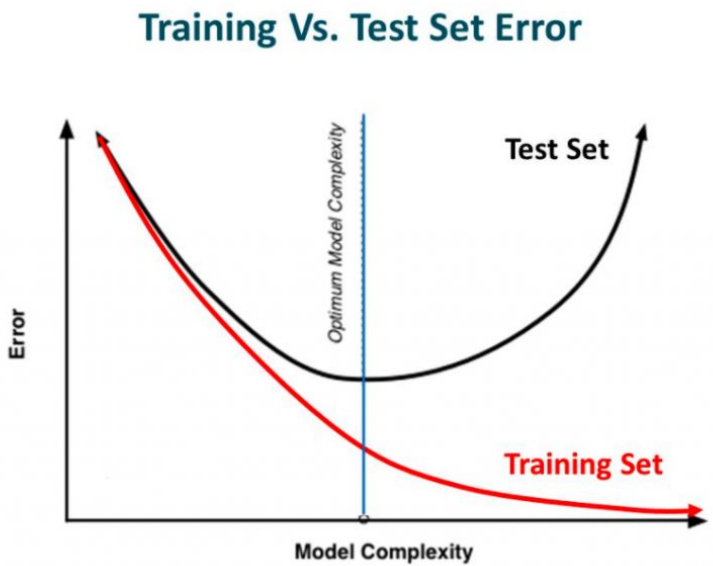
  which maximizes the attack success rate (promotes the *misclassification of perturbations*);
- the **GAN loss**

  $$\mathcal{L}_{GAN} = \mathbb{E}_x log\mathcal{D}_\rho(x) + \mathbb{E}_x log(1 - \mathcal{D}_\rho(x + \mathcal{G}_\phi(x)))$$

  subject to the min-max problem:
  - gradient descent over $\mathcal{G}$ minimizes the likelihood of the discriminant being correct over perturbations, thus making sure that the *perturbed data resembles the original data;*
  - gradient ascent over $\mathcal{D}$ optimizes the parameters ρ towards the *correct discrimination of points.*

Defences

Lipschitz regularization



It is well known that as the model learns to correctly classify the training data, its *complexity increases* in such a way that it performs poorly on unseen test data.

This phenomenon is known as **overfitting** and is avoided by adding to the average training loss a **regularization term** $R(f)$, which imposes a penalty on the complexity of f.
In the context of deep learning, this factor penalizes the weight matrices θ of the nodes

$$min_\theta \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i, \theta), y_i) + \lambda R(f)$$

Defences

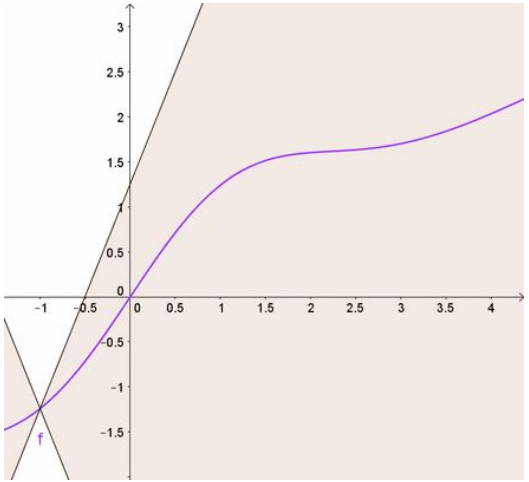
Lipschitz regularization

The **Lipschitz constant** of the model can be used to establish *adversarial robustness metrics*, compute upper and lower *bounds for an adversarial perturbation* on and to define *regularization terms.*

A function $f : \mathcal{X} \to \mathcal{Y}$ is said to be **Lipschitz continuous** it if satisfies

$$d_{\mathcal{Y}}(f(x_1), f(x_2)) \leq L_f \cdot d_{\mathcal{X}}(x_1, x_2), \forall x_1, x_2 \in \mathcal{X}$$

for some real-valued $L_f \geq 0$.



wikipedia.org

Defences

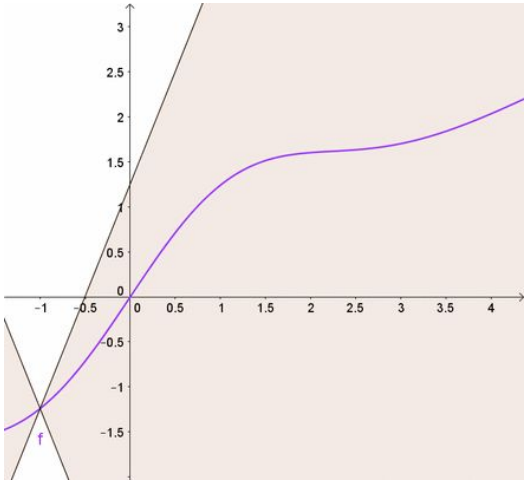
Lipschitz regularization

Neural networks can be expressed as a **composition on non-linear functions** representing the *hidden layers* in the network

$$f(x) = (\phi_l \circ ... \circ \phi_1)(x).$$

Supposing that each ϕ_j is L_j-Lipschitz continuous, f turns out to be at least $L_1 \cdot ... \cdot L_l$- Lipschitz continuous:

$$L \leq \prod_{j=1}^{l} L_j.$$



wikipedia.org

Defences

Lipschitz regularization

Cissé et al. (2017) provide a layerwise regularization method which penalizes the *global lipschitz constant* of the network.
In their **Parseval Network** all the non-linearities of the network are at most 1-Lipschitz, so that the resulting global Lipschitz constant is smaller than 1.

Hein et al. (2017) propose an instance-specific *lower bound on the norm of an adversarial perturbation,* based on the local Lipschitz constant, and define the **Cross-Lipschitz regularization** functional.

Weng et al. (2018) introduce **Clever,** the first *attack-independent* robustness metric for neural networks. They derive a *lower bound on the minimal perturbation* required to craft adversaries by computing an estimate of the local Lipschitz constant in a ℓ_p ball around each point.

Defences

Lipschitz regularization

Some recent work has been done on the **Lipschitz constant L_ℓ of the loss** function ℓ

This constant controls the norm of a perturbation η on a sample x

$$||\ell(f(x+\eta,\theta),y) - \ell(f(x,\theta),y)||_{\mathbb{R}^K} \leq L_\ell \cdot ||\eta||_{\mathbb{R}^d},$$

so *maps with a smaller Lipschitz constant may lead to more adversarially robust models* (Oberman et al., 2018).

Defences

Lipschitz regularization

Finlay et al. (2019) augment the training loss with a sample-efficient Lipschitz regularization method called **worst case adversarial training** (WCAT).
They notice that the regularization of the Lipschitz constant of the loss is equivalent to the *regularization of f in one direction*, but at much lower cost, because the loss is a scalar:

$$\nabla_x \ell(f(x,\theta),y) = \nabla_\theta \ell(f(x,\theta),y)\nabla_x f(x,\theta)$$

So they compute an *underestimate the Lipschitz constant of ℓ,* by sampling the norm of the gradient on a minibatch $\mathcal{X}$ of training points

$$max_{x\in\mathcal{X}}||\nabla \ell(f(x,\theta),y)|| \leq L_\ell,$$

and significantly improve adversarial robustness on the trained networks.

Recap

- Tricking Neural Networks
- Manifold Hypothesis
- Enhanced problems
- Attacks
- Defences

Conclusions

Adversarial example show that many of the modern machine learning algorithms can be fooled in unexpected ways.

Both in terms of attacks and defences, a lot of theoretical problems still remain open.
From an applicative point of view, no one has yet designed a powerful defense algorithm that can resist to a variety of attacks.

> *"We encourage machine learning researchers to get involved and design methods for preventing adversarial examples, in order to close this gap between what designers intend and how algorithms behave. "*
>
> OpenAI, 2017

References

- **Aghdam et al.** (2017) Explaining Adversarial Examples by Local Properties of Convolutional Neural Networks
- **Akhtar et al.** (2017) Defense against Universal Adversarial Perturbations
- **Amsaleg et al.** (2016) The Vulnerability of Learning to Adversarial Perturbation Increases with Intrinsic Dimensionality
- **Bengio** (2016) Deep Learning
- **Bishop** (2006) Pattern Recognition and Machine Learning
- **Bortolussi et al.** (2019) Intrinsic Geometric Vulnerability of High-Dimensional Artificial Intelligence
- **Carlini et al.** (2017) Towards Evaluating the Robustness of Neural Networks
- **Cissé et al.** (2017) Parseval Networks: Improving Robustness to Adversarial Examples
- **Dezfooli et al.** (2016) DeepFool: a simple and accurate method to fool deep neural networks
- **Dube et al.** (2018) High Dimensional Spaces, Deep Learning and Adversarial Examples
- **Eykholt et al.** (2018) Robust Physical-World Attacks on Deep Learning Models
- **Fawzi et al.** (2018) Empirical study of the topology and geometry of deep networks
- **Finlay et al.** (2019) Improved robustness to adversarial examples using Lipschitz regularization f the loss
- **Franceschi et al.** (2018) Robustness of classifiers to uniform ell_p and Gaussian noise

References

- **Gilmer et al.** (2018) The Relationship Between High-Dimensional Geometry and Adversarial Examples
- **Goodfellow et al.** (2014) Generative Adversarial Networks
- **Goodfellow et al.** (2015) Explaining and Harnessing Adversarial Examples
- **Gouk et al.** (2018) Regularisation of Neural Networks by Enforcing Lipschitz Continuity
- **He et al.** (2018) Decision Boundary Analysis of Adversarial Examples
- **Hein et al.** (2017) Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation
- **Khoury et al.** (2018) On the Geometry of Adversarial Examples
- **Kurakin et al.** (2017) Adversarial Machine Learning at Scale
- **Kurakin et al.** (2018) Adversarial Attacks and Defences Competition
- **Madry et al.** (2017) Towards Deep Learning Models Resistant to Adversarial Attacks
- **Ma et al.** (2018) Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality
- **Oberman et al.** (2018) Lipschitz regularized Deep Neural Networks converge and generalize
- **Papernot et al.** (2017) Practical Black-Box Attacks against Machine Learning
- **Poursaeed et al.** (2018) Generative Adversarial Perturbations
- **Rauber et al.** (2018) Foolbox: A Python toolbox to benchmark the robustness of machine learning models
- **Serban et al.** (2018) Adversarial Examples - A Complete Characterisation of the Phenomenon

References

- **Tramèr et al.** (2018) Ensemble Adversarial Training: Attacks and Defenses
- **Tsipras et al.** (2018) Robustness May Be at Odds with Accuracy
- **Wang et al.** (2017) A Theoretical Framework for Robustness of (Deep) Classifiers against Adversarial Examples
- **Weng et al.** (2018) Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach
- **Xiao et al.** (2018) Generating Adversarial Examples with Adversarial Networks
- **Yuan et al.** (2018) Adversarial Examples: Attacks and Defenses for Deep Learning