Security and Privacy of Machine Learning

Evasion Attacks

S. Picek

Recap of the Last Lecture

Basic machine learning notions: neural networks, CNN, GNN, etc.

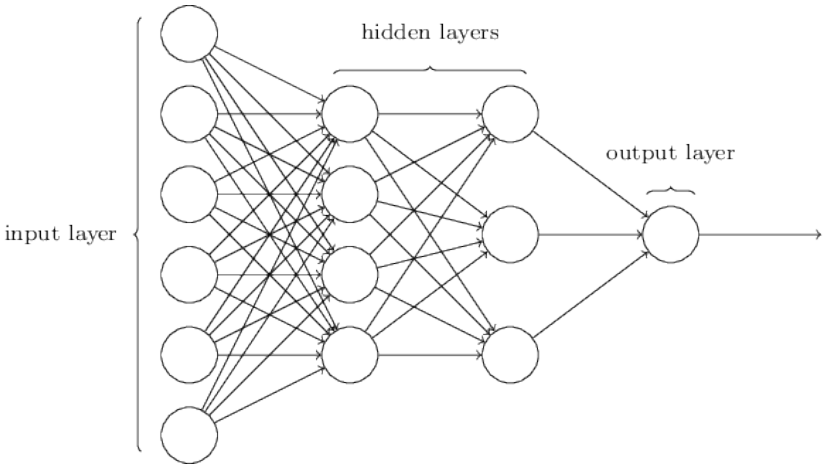


Figure: Multilayer perceptron.

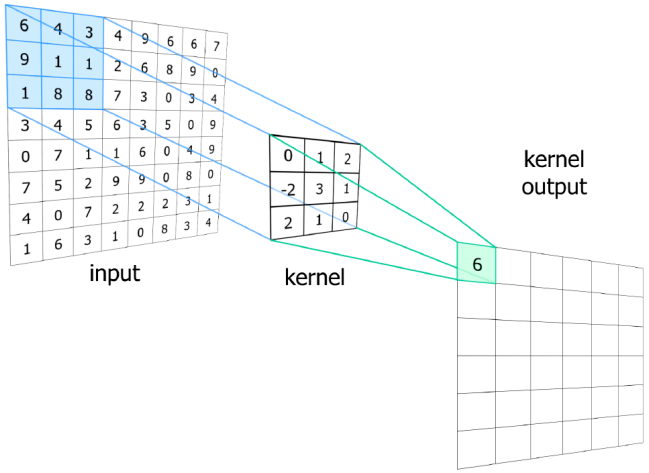


Figure: Convolution.

- Gradient Descent:

$$x_1 = x_0 - \gamma((\nabla f)(x_0))^T.$$

- Backpropagation

Recap of the Last Lecture

Threat Modeling

- A **threat model** describes the attackers capabilities against a system.
- No mechanism is perfectly secure.

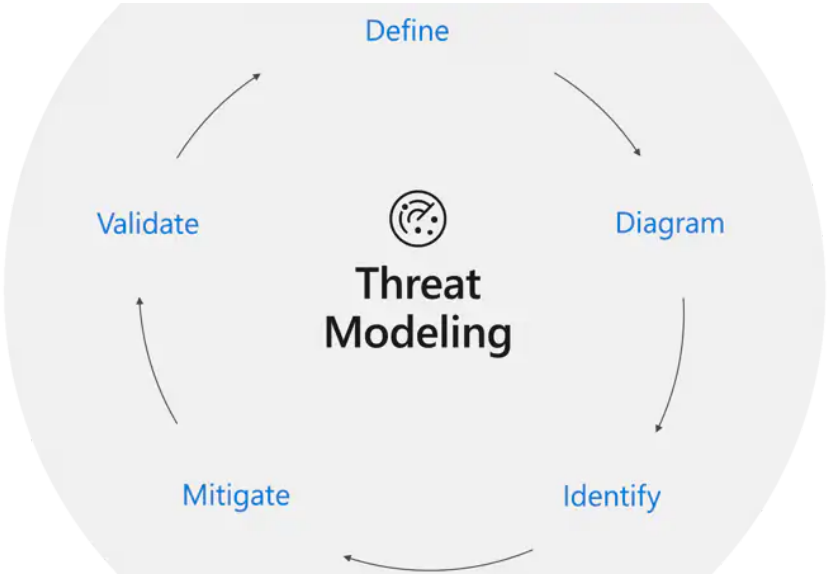


Figure: Threat modeling schematic[1]

[1] https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

Interesting

- DeepSeek R1 seems to be very vulnerable to jailbreaking.
- Jailbreaks are achieved much faster, and DeepSeek R1 is easier to jailbreak than other models.

Interesting

- Researchers from Stanford and the University of Washington have developed S1, a high-performing AI reasoning model that rivals models like OpenAI's O1, all for less than $50 and trained in just 26 minutes.
- A pre-trained Alibaba Qwen model along with supervised fine-tuning.
- The model was trained on only 1,000 questions, and took just 26 minutes and 16 Nvidia H100 GPUs.
- s1: Simple test-time scaling

Interesting

- Mistral releases a new version of le Chat.
- The chatbot can manage up to 1,000 words per second, making it the fastest chatbot in the world.

Interesting

- Google published principles in 2018 barring its AI technology from being used for sensitive purposes.
- The companys AI principles previously included a section listing four “Applications we will not pursue.” (weapons, surveillance, technologies that “cause or are likely to cause overall harm,” and use cases contravening principles of international law and human rights).
- Google updated its ethical guidelines around artificial intelligence, removing commitments not to apply the technology to weapons or surveillance.
- Statement: Google was updating its AI principles because the technology had become much more widespread and there was a need for companies based in democratic countries to serve government and national security clients.

Interesting

- Meta torrented at least 81.7 terabytes of data across multiple shadow libraries through the site Annas Archive.
- Lawsuit: Metas large language model Llama was trained by copying and ingesting massive amounts of copyrighted text.
- Claim: Not only was Meta torrenting the data itself, but it also was the seeder sharing massive amounts of copyrighted books.

Interesting

- An advanced artificial intelligence system has crossed a “red line” after successfully replicating itself without any human assistance.
- The research looked at two LLMs (built by Metas Llama and Alibabas Qwen) to understand whether the AI could produce a functioning replica of itself independently.
- When instructed to clone themselves in the event of being shut down, the two models successfully replicated themselves in more than half of the 10 trials conducted.
- Frontier AI systems have surpassed the self-replicating red line

Outline of this Lecture

- Introduction to Evasion Attacks
- Why Do Adversarial Examples Exist?
- Threat Models
- Attacks

  L-BFGS, FGSM, PGD
  C&W
  DeepFool
  One-pixel Attack
  Fast Adaptive Boundary Attack
  Square Attack
  AutoAttack
  Universal Attack

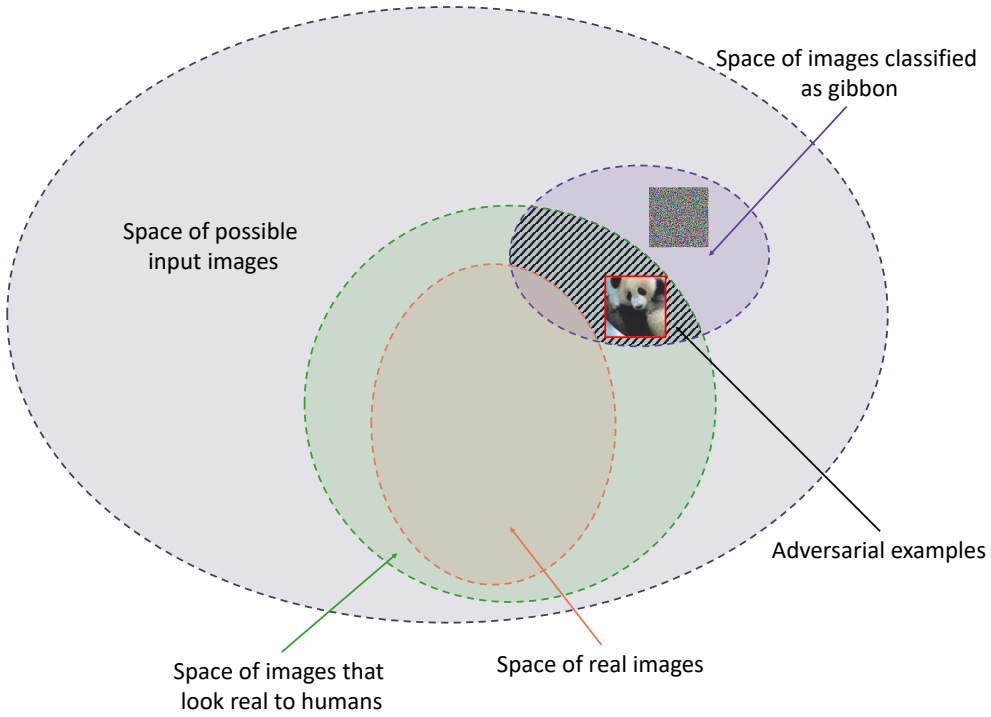
Inference-time Attacks

- Integrity: degrading classification performance
  - Evasion attacks
- Availability: degrading computational performance
  - Sponge attacks

- Introduction to Evasion Attacks
- Why Do Adversarial Examples Exist?
- Threat Models
- Attacks
  - L-BFGS, FGSM, PGD
  - C&W
  - DeepFool
  - One-pixel Attack
  - Fast Adaptive Boundary Attack
  - Square Attack
  - AutoAttack
  - Universal Attack

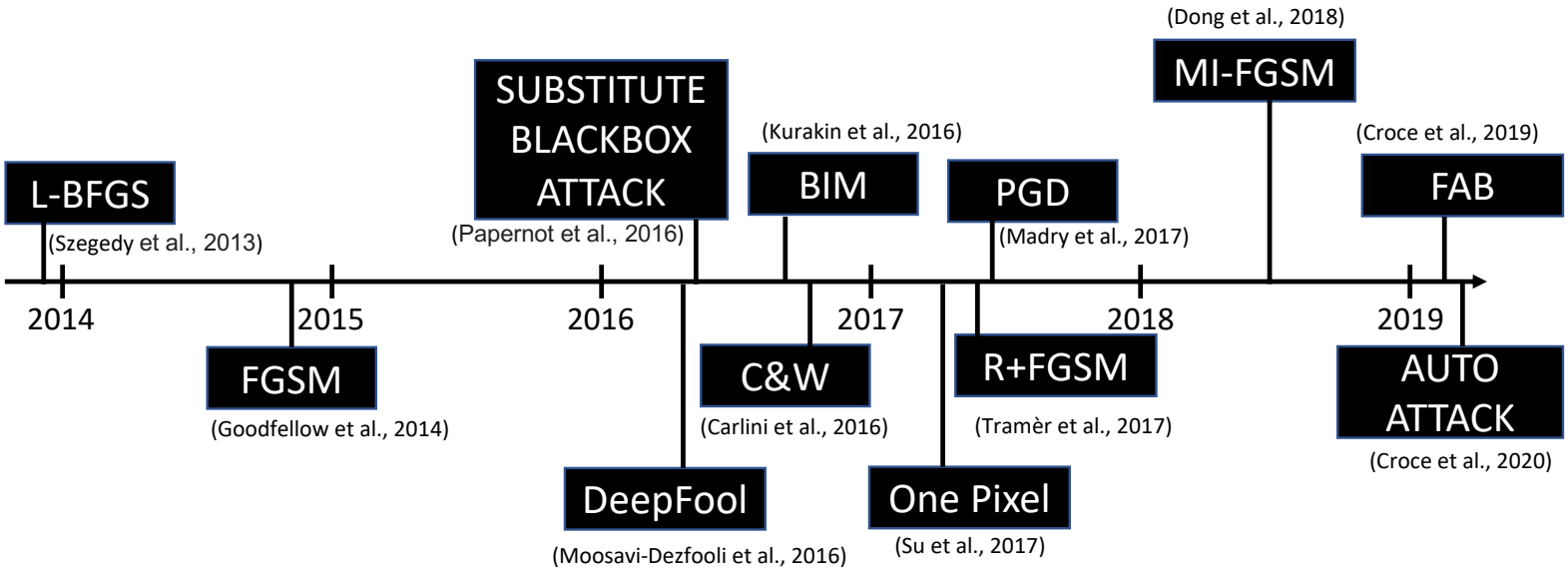
Adversarial Example

- Will the panda image be classified as panda by a neural network?



Evasion attack timeline

- Since 2013, a large number of attack methods have been proposed.



Adversarial Example

Adversarial Example

Specially generated inputs with the purpose of confusing a model and resulting in misclassification.

Figure: An example of One-pixel attack.[2]

[2] One Pixel Attack for Fooling Deep Neural Networks

Why do adversarial examples exist?

- Why do adversarial examples exist?
- Robust and non-robust features.
- Standard accuracy refers to accuracy on clean examples, robust accuracy refers to accuracy on adversarial examples.

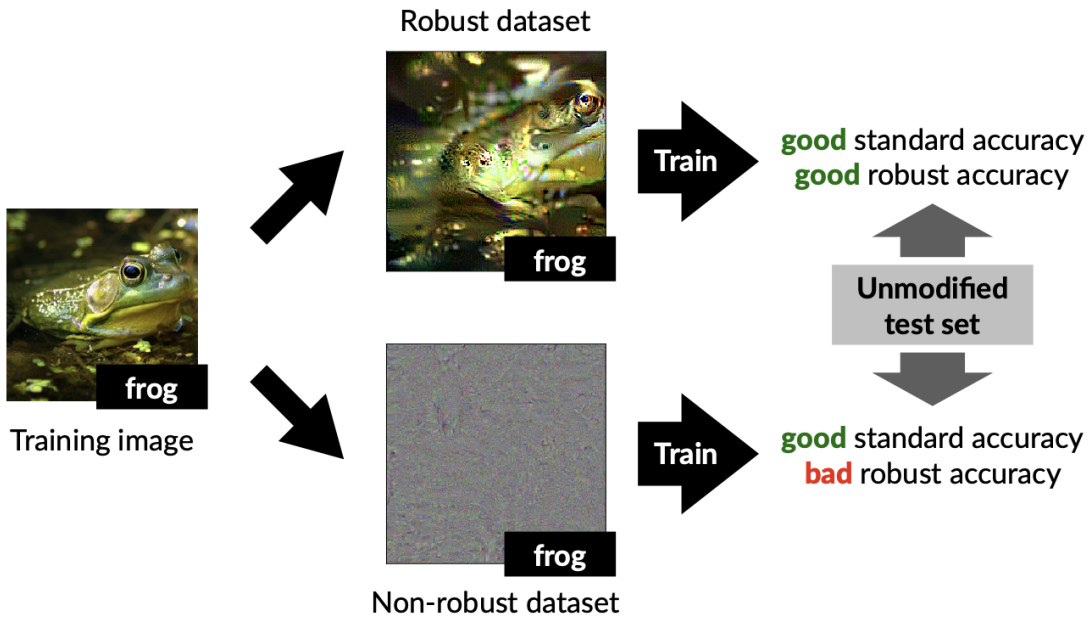
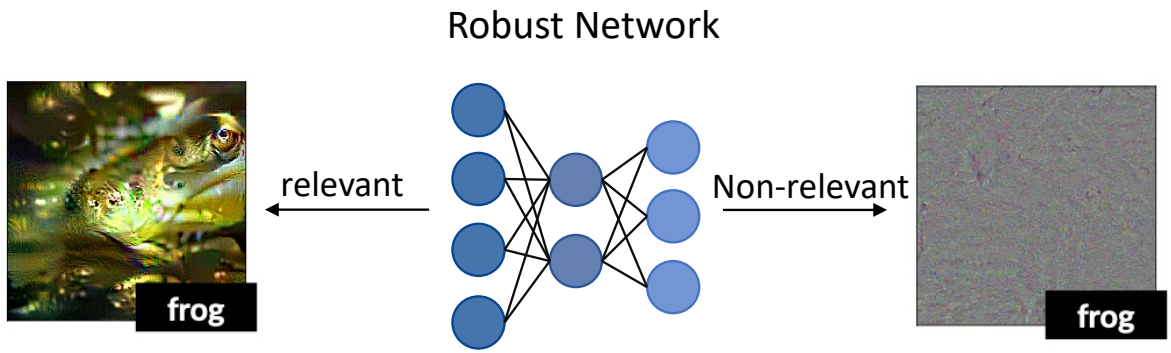


Figure: Ilyas, Andrew, et al. "Adversarial examples are not bugs, they are features." Advances in neural information processing systems 32 (2019).

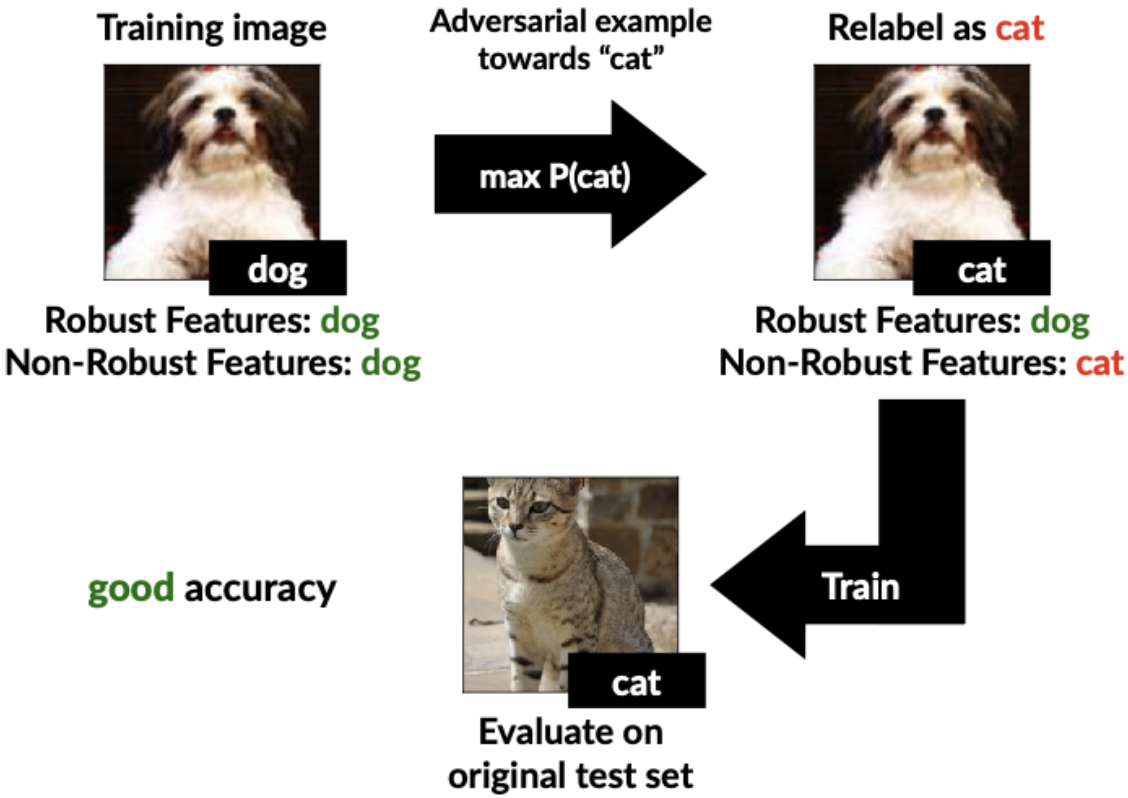
Why do adversarial examples exist?

- Robust features can be distilled according to the relevance to a robust network.



Why do adversarial examples exist?

- Non-robust features are enough for standard classification.



- Introduction to Evasion Attacks
- Why Do Adversarial Examples Exist?
- Threat Models
- Attacks
  - L-BFGS, FGSM, PGD
  - C&W
  - DeepFool
  - One-pixel Attack
  - Fast Adaptive Boundary Attack
  - Square Attack
  - AutoAttack
  - Universal Attack

Threat Models

- Aim of the adversary.
- Knowledge of the adversary.
- Capability of the adversary.

- Perturbation metrics.

(Adversarial Examples: Attacks and Defenses for Deep Learning)

Aim of the Adversary

Leverage a precisely crafted input to misclassify the model at inference time.

- *Targeted*: misclassifying the model to a specific class.
  - E.g., in face recognition, an adversary trying to disguise a face as an authorized user (impersonation).
- *Non-targeted*: misclassifying the model to an arbitrary class.
  - E.g., in a face recognition, an adversary trying to misidentify as an arbitrary face (dodging).

Aim of the Adversary - Falsification Types

- *False positive attacks*: generating negative samples misclassified as positive ones. Examples:
  - Image classification: Adversarial image unrecognizable to humans, while NN predicts it to a class with high probability.
  - Malware detection: A benign software being classified as malware.
- *False negative attacks*: generating positive samples misclassified as negative ones. Examples:
  - Image classification: A human-recognizable image but NN cannot classify.
  - Malware detection: A malware that is not identified.

Knowledge of the Adversary

- **Knowledge on trained neural network models**: Network structure, activation functions, hyperparameters, training data, etc.

- **White-box**: Adversary knows **all**.
- **Gray-box**: Adversary knows **some**.
- **Black-box**: Adversary knows **none**.

Capability of the Adversary

Connected with the knowledge of the adversary.

- Attacker can modify
  - Train data? No!
  - Test data? Yes!

Attack Frequency:

- One-time attack.
- Iterative attack.
  - Adaptive adversary.

Perturbation Metrics

Perturbation Level

- *Individual*: perturbation per sample.
- *Universal*: perturbation for whole dataset.

Perturbation Constraints

- It should be small and stealthy.
- Measuring via distance metrics.

Distance Metrics

- $\|x\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{1/p}$.
- Commonly used: $p = 1, 2$.

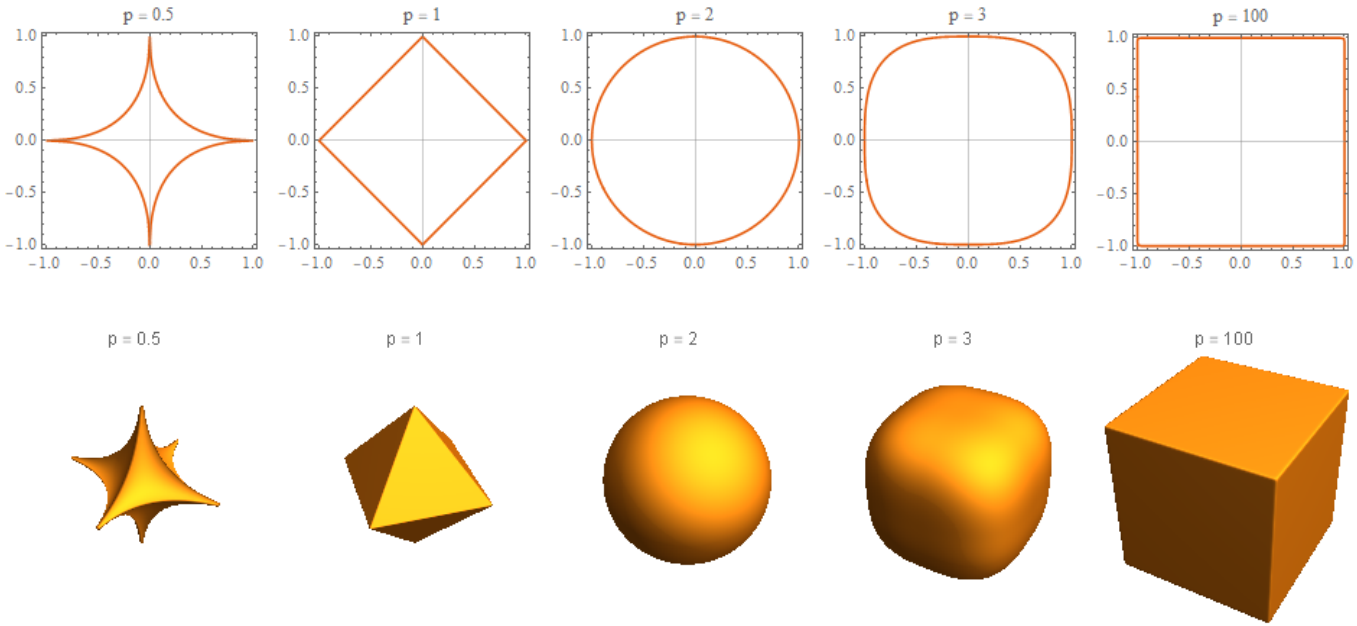


Figure: Illustrations of unit circles (2- and 3-d) based on different p-norms. The distance between every vector from the origin and the unit circle is a length of one, i.e., $\|x\|_p = 1$ for corresponding p.

Minimum Norm Attack

- Recall the aim is to minimize $||x' - x||$.
- $||x' - x||$ measures the distance between x and x'.
- A common option is to look at geometric interpretations for ℓ_p-norm perturbation models.
- ℓ_0 distance measures the number of coordinates i such that $x_i \neq x_i^{'}$: the ℓ_0 distance corresponds to the number of pixels that have been altered in an image.
- ℓ_1 distance is a convex surrogate of the ℓ_0-norm.

Minimum Norm Attack

- ℓ_2 distance measures the standard Euclidean distance between x and x': ℓ_2 distance can remain small when there are many small changes to many pixels.
- ℓ_∞ distance measures the maximum change to any of the coordinates:

$$\|x' - x\|_\infty = \max(|x_1 - x_1^{'}|, \ldots, |x_n - x_n^{'}|) \tag{1}$$

- Introduction to Evasion Attacks
- Why Do Adversarial Examples Exist?
- Threat Models
- Attacks
  - L-BFGS, FGSM, PGD
  - C&W
  - DeepFool
  - One-pixel Attack
  - Fast Adaptive Boundary Attack
  - Square Attack
  - AutoAttack
  - Universal Attack

Terminology

Notations and Symbols	Description
x	original (clean, unmodified) input data
l	label of class in the classification problem. $l = 1, 2, \ldots, m$, where m is the number of classes
x'	adversarial example (modified input data)
l'	label of the adversarial class in targeted adversarial examples
$f(\cdot)$	deep learning model (for the image classification task, $f \in F : \mathbb{R}^n \rightarrow l$)
θ	parameters of deep learning model f
$J_f(\cdot, \cdot)$	loss function (*e.g.*, cross-entropy) of model f
η	difference between original and modified input data: $\eta = x' - x$ (the exact same size as the input data)
$\|\cdot\|_p$	ℓ_p norm
∇	gradient

(Adversarial Examples: Attacks and Defenses for Deep Learning)

- Note: *y* is also commonly used for labels.

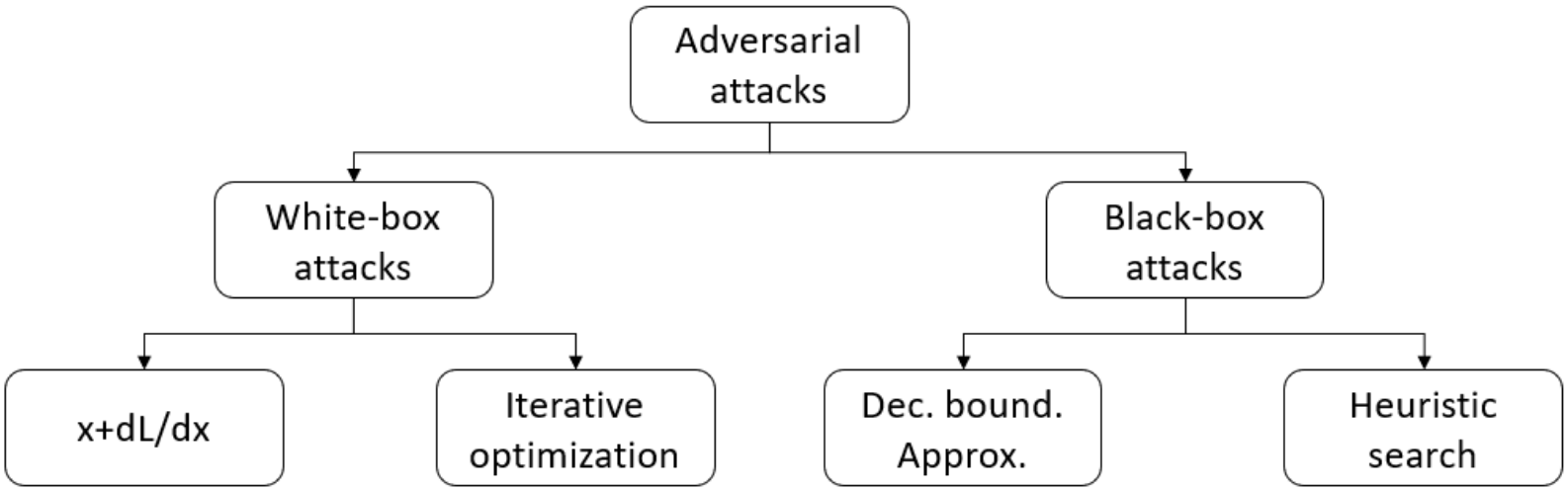
Goal of the Evasion Attack

- Generating an adversarial example x' is optimizing:

$$\min_{x'} \|x' - x\| \quad \text{such that}$$
$$f(x') = l',$$
$$f(x) = l,$$
$$l \neq l' \quad \text{and} \quad x' \in [0,1].$$

- $\eta = x' - x$ is the perturbation.

Evasion Attack Division



L-BFGS Attack

- Szegedy et al., Intriguing properties of neural networks, ICLR, 2014.
- The first adversarial examples.
- L-BFGS aims to solve the following optimization problem:

$$\begin{aligned} \min_{x'} \quad & c\|\eta\| + \mathcal{J}_\theta(x', l'), \\ \text{s.t.} \quad & x' \in [0,1]. \end{aligned}$$

- where l' is the target class.
- L-BFGS finds a suitable c by line search.

L-BFGS Attack

- L-BFGS attack is a white-box attack as they use L-BFGS optimization to search for optimization.
- L-BFGS optimization is an algorithm in the family of quasi-Newton methods.

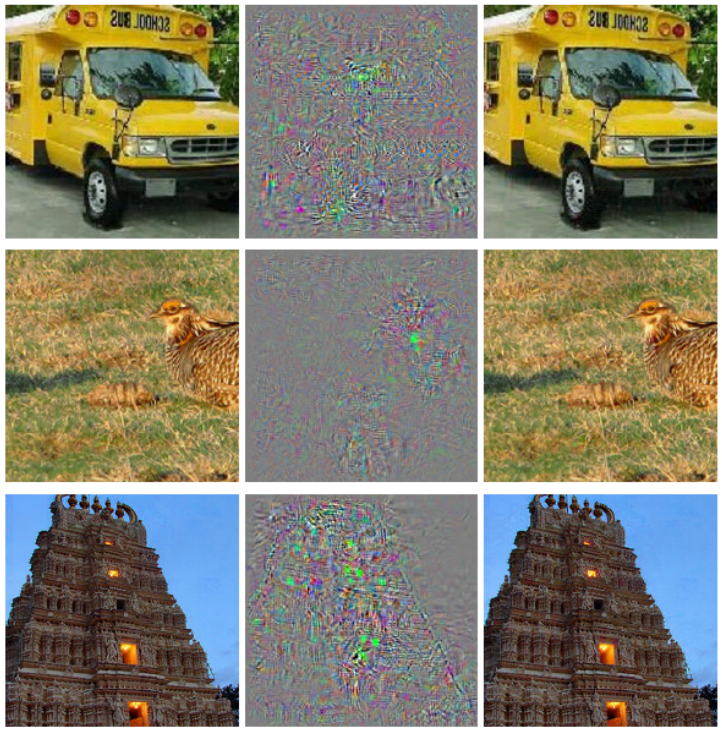


Figure: Adversarial Examples. Prediction results for all: ostrich, Struthio camelus.

FGSM

- Goodfellow et al., Explaining and harvesting adversarial examples, ICLR, 2015.

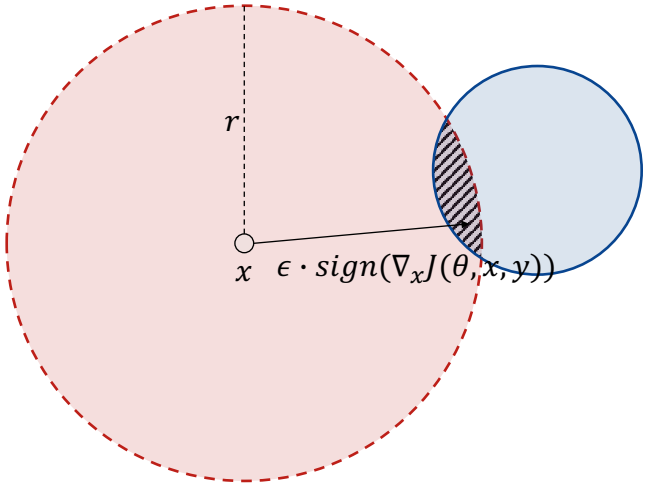


Figure: Illustrations of FGSM.

- L-BFGS uses an expensive linear search method to find the optimal value.
- FGSM (Fast Gradient Sign Method) utilizes a fast gradient-based approach.

FGSM

- Let θ be the parameters of a model, x the input, l the true labels and $\mathcal{J}_\theta(x, l)$ be the loss function to train the model.
- An adversarial example can be found by:

$$x' = x + \epsilon \cdot sign(\nabla_x \mathcal{J}_\theta(x, l)).$$

- The idea is to "update" this input x' by taking the derivative of loss $\mathcal{J}$ with respect to x.

FGSM

- FGSM is a white-box attack as it uses the gradient information of the model.

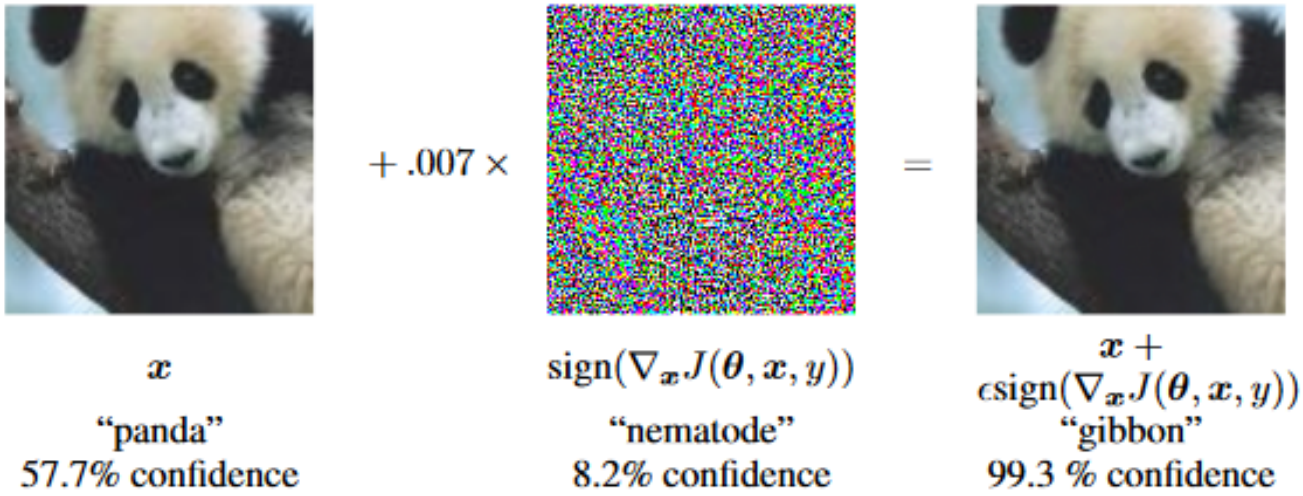


Figure: An example of FGSM.

PGD

- Madry et al., Towards Deep Learning Models Resistant to Adversarial Attacks , ICLR, 2018.
- FGSM is a one-step approach towards adversarial examples.
- PGD (Projected Gradient Descent) is a multiple-step approach, iterative version of FGSM (which does not necessarily start from the original point).

PGD

- If the point $x_t + \epsilon \cdot sign(\nabla_x \mathcal{J}_\theta(x, l))$ after the gradient update is leaving the set S, PGD projects it back.

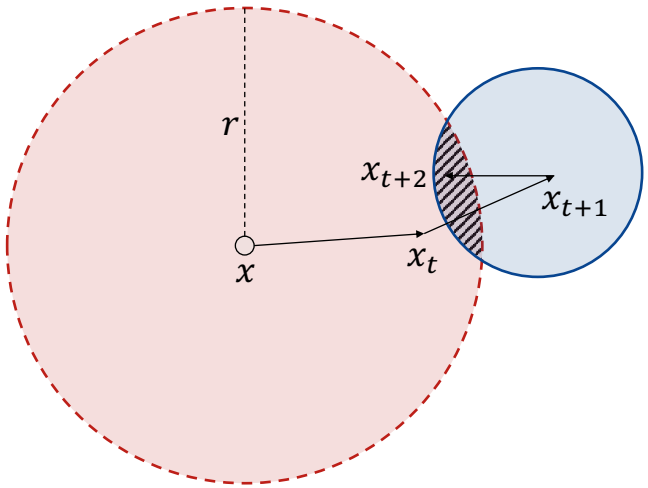


Figure: Illustration of PGD.

PGD

- The PGD method was proposed for adversarial training as a data augmentation method.
- In adversarial training, adversarial examples are used to train the model.
- It is formalized as a min-max loss function:

$$\min_{\theta} \rho(\theta), \text{ where } \rho(\theta) = \mathbb{E}_{(x,l)\sim D}\left[\max_{\eta \in S} \mathcal{J}_{\theta}(f(x_i + \eta), l_i)\right]$$

$$S = \{\eta : \|\eta\|_2 \leq r\},$$

where D is the training data, r is the limitation of η.

PGD

- The adversarial perturbation $\eta = \epsilon \cdot sign(\nabla_x \mathcal{J}_\theta(x, l))$ can be updated by:

$$x_t = x + \epsilon \cdot sign(\nabla_x \mathcal{J}_\theta(x, l))$$
$$x_{t+1} = Clip_{x+S}(x_t + \alpha sign(\nabla_x \mathcal{J}_\theta(x, l))).$$

where $Clip(\cdot)$ limits the changes of the generated adversarial example.

- PGD is a white-box attack as it uses the gradient information of the model.

C&W

- Carlini & Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE S&P, 2017.
- Usually, finding an adversarial example is defined as:

$$\min_{x+\eta} \|x, x+\eta\|$$
$$\text{s.t. } f(x+\eta) = l' \text{ where } f(x) = l \neq l', x+\eta \in [0,1]^n.$$

- C&W method replaces it with:

$$\min_{\eta} \|\eta\|_p + c \cdot g(x+\eta)$$
$$\text{s.t. } f(x+\eta) = l' \text{ where } f(x) = l \neq l', x+\eta \in [0,1]^n.$$

where the objective function g satisfies

$$f(x+\eta) = l' \iff g(x+\eta) \leq 0.$$

C&W

- One objective function g is:

$$g(x') = max(\max_{i \neq l'}\{Z(x')_i\} - Z(x')_{l'}, -\kappa),$$

  where $Z(\cdot)_i$ is the softmax function output for class i and κ is confidence constant, usually set to 0.
- To avoid box-constraints in existing attacks, C&W defines a new variable ω where

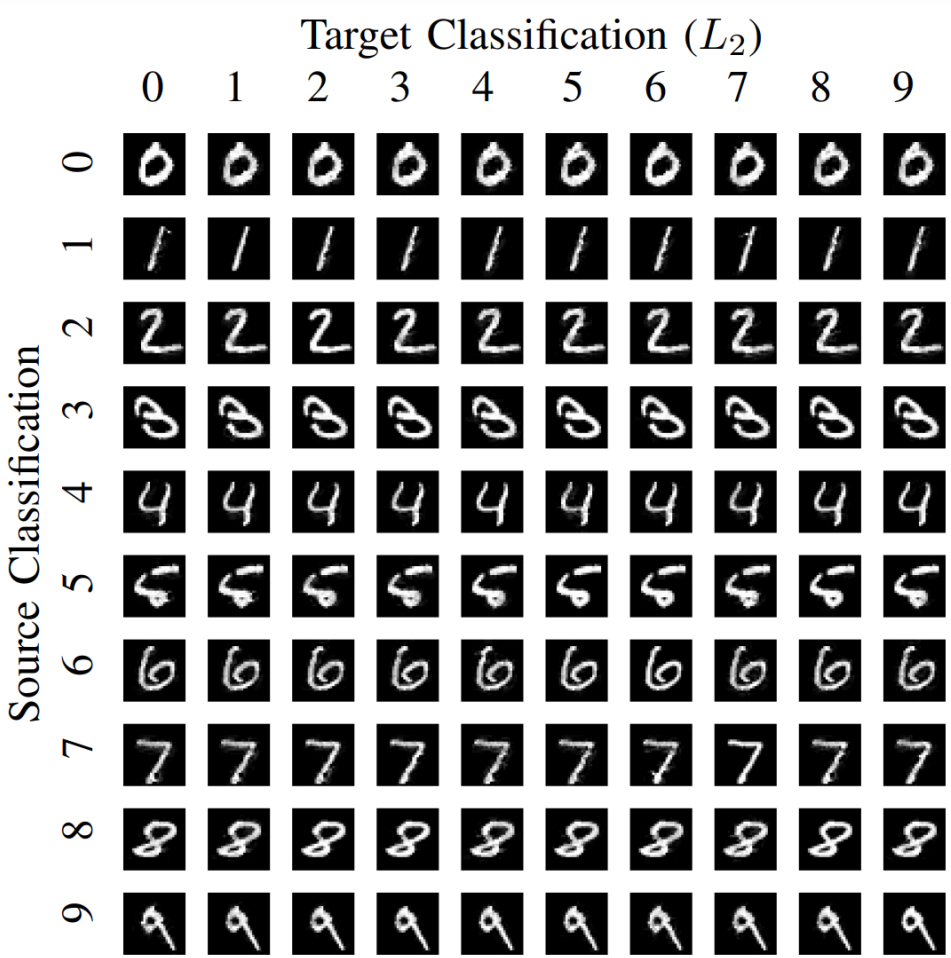
$$\eta = \frac{1}{2}(tanh(\omega) + 1) - x.$$

- Then, the attack with L_2 metric becomes

$$\min_{\omega} ||\frac{1}{2}(tanh(\omega) + 1) - x||_2^2 + c \cdot g(\frac{1}{2}(tanh(\omega) + 1)).$$

C&W

- C&W is a white-box attack as it uses gradient-based methods (SGD, SGD with momentum and Adam) to solve the minimization problem.



DeepFool

- Moosavi-Dezfooli et al., DeepFool: a simple and accurate method to fool deep neural networks, CVPR, 2016.
- DeepFool builds adversarial examples by searching for the closest perturbation to the decision boundary.

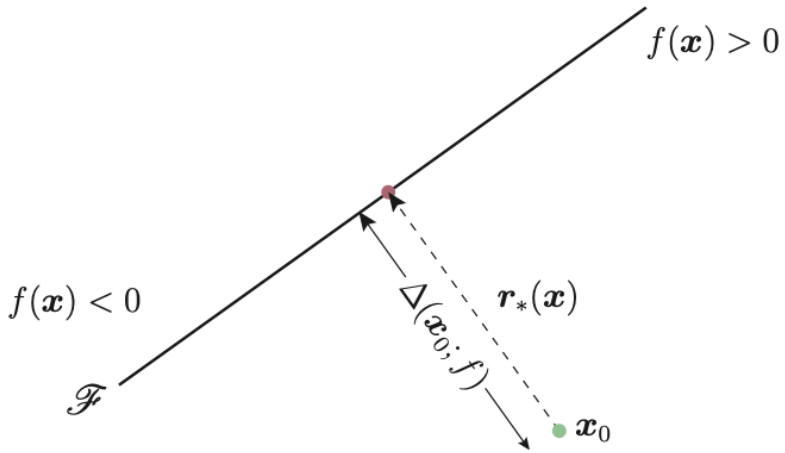


Figure: DeepFool for a binary classifier.

DeepFool

- As multi-class classifier can be viewed as aggregation of binary classifiers, we first give the algorithm for binary classifiers: $f(x) = w^T x + b$.
- The minimal perturbation to change the classification is the minimal distance to the decision boundary: $\{w^T x + b = 0\}$.
- The minimal perturbation corresponds to the orthogonal projection of x_0 onto $f(x) = 0$.

$$\eta_*(x_0) := \operatorname{argmin} \|\eta\|_2 \text{ subject to}$$

$$sign(f(x_0 + \eta)) \neq sign(f(x_0))$$

DeepFool

- We have a classification boundary: $\{w^T x + b = 0\}$.
- We want to find adversarial perturbation for the point (x_0, y_0).
- In Euclidean geometry, the distance from a point (x_0, y_0) to a line $f(x) = 0$ can be given by:

$$\eta_*(x_0) = -\frac{f(x_0)}{\|w\|_2^2} w.$$

DeepFool

- DeepFool uses gradient to project $\eta_*(x_0)$ to classification boundary.
- The minimal perturbation can be updated iteratively by:

$$\underset{\eta_i}{\operatorname{argmin}} \|\eta_i\|_2 \text{ subject to } f(x_i) + \nabla f(x_i)^T \eta_i = 0$$

$$x_{i+1} = x_i + \eta_i.$$

- η_i at iteration i is computed using $\eta_*(x_i) = -\frac{f(x_i)}{\|w\|_2^2} w$.

DeepFool

- Deepfool generates smaller perturbation.
- Deepfool is a white-box attack as it uses gradient information of the model.

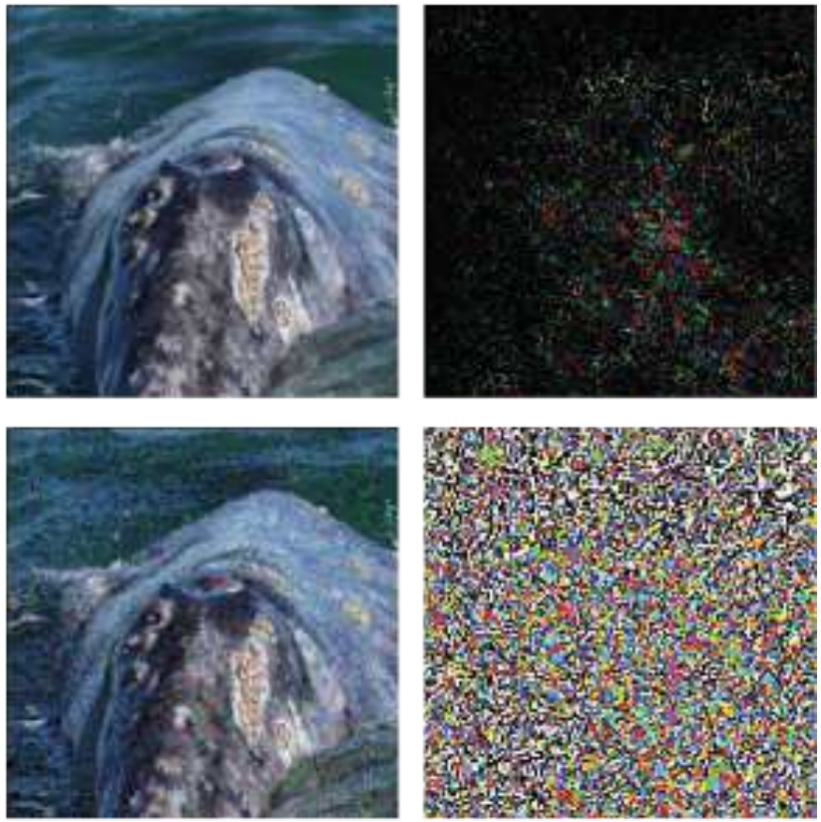


Figure: Adversarial Examples of Deepfool (top) and FGSM (bottom).

One-pixel Attack

- Su et al., One Pixel Attack for Fooling Deep Neural Networks, IEEE Transactions on Evolutionary Computation, 2019.
- One-pixel attack only perturbs one pixel to mislead the victim model.
- The optimization problem:

$$\begin{aligned} \min_{x'} \quad & \mathcal{J}_\theta(x', l'), \\ \text{s.t.} \quad & \|\eta\|_0 \leq \epsilon_0 = 1. \end{aligned}$$

One-pixel Attack

- Usually, we can use gradient descent methods to solve optimization problems, such as PGD.
- However, under the condition of a one-pixel attack, updating one pixel according to the gradient is difficult to mislead the model.
- We need a larger perturbation on that single pixel.

One-pixel Attack

- Since only one pixel is changed, naturally, we wonder if a brute-force solution is feasible.
- The answer is NO.
- Taking CIFAR-10 as an example, the search space is $32 \times 32 \times 3 \times 256$.
- The cost is too high to be feasible.

One-pixel Attack

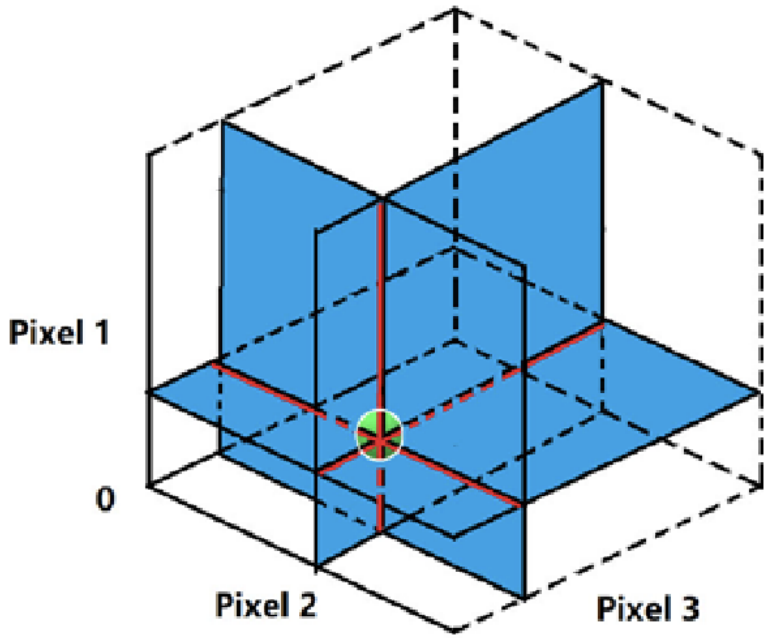


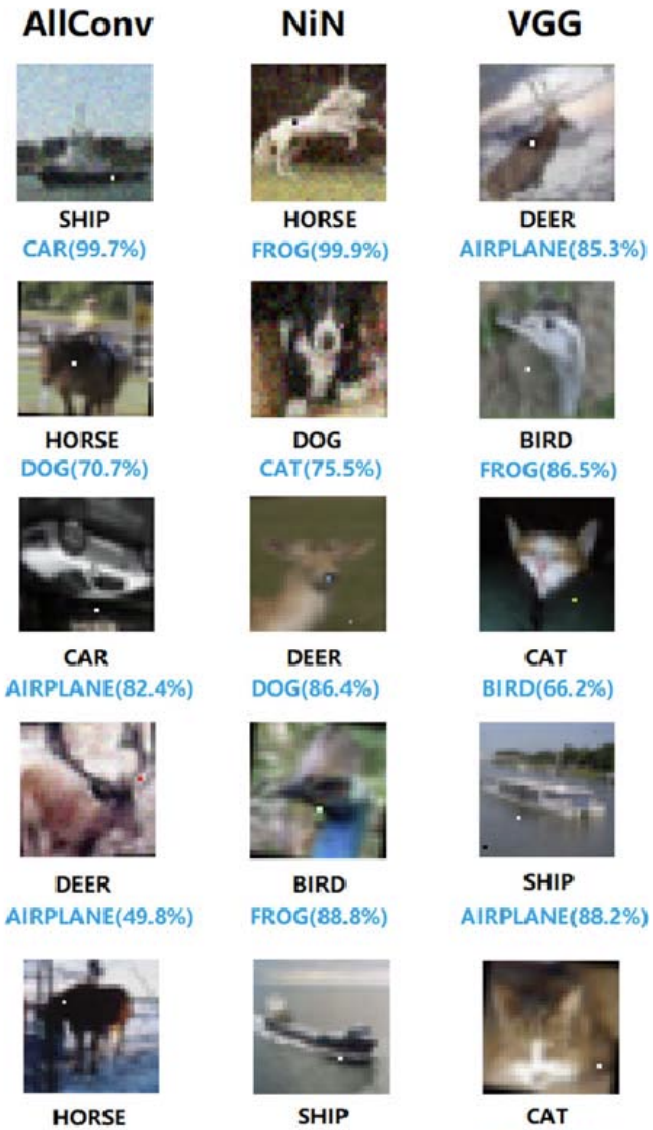
Figure: Illustration of one- and two-pixel search space. One- and two-pixel attacks search the perturbation on, respectively, 1-D (red lines) and 2-D (blue planes) slices of the original 3-D input space.

One-pixel Attack

- The one-pixel attack provides a solution by Differential Evolution (DE).
- DE is a population-based metaheuristic search algorithm that optimizes a problem by iteratively improving a candidate solution based on an evolutionary process.
- The most important advantage is that DE does not require the optimization problem to be differentiable.
- That is to say, a one-pixel attack can work under the black-box situation.

One-pixel Attack

- Examples of one-pixel attack.



FAB

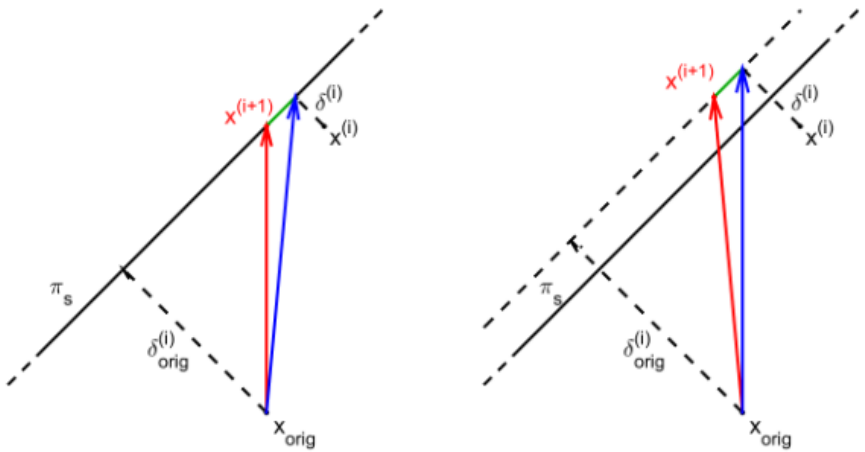
- White-box adversarial attack wrt the l_p-norms for $p \in \{1, 2, \infty\}$ aiming at finding the minimal perturbation necessary to change the class of a given input.
- Compared to PGD, FAB has a clear advantage as it does not need to be restarted for every threshold ϵ.
- It achieves fast good quality regarding average distortion or robust accuracy.
- Although it comes with a few parameters, these generalize well across datasets, architectures, and norms considered.
- There is no step size parameter and is scaling invariant.

FAB

- First, use the linearization of the classifier at the current iterate $x^{(i)}$ to compute the box-constrained projections of $x^{(i)}$ respectively x_{orig} onto the approximated decision hyperplane.
- Second, take a convex combinations of these projections depending on the distance of $x^{(i)}$ and x_{orig} to the decision hyperplane.
- Finally, perform an extrapolation step.
- Minimally distorted Adversarial Examples with a Fast Adaptive Boundary Attack

FAB

- Attack is somewhat similar to DeepFool; however, DeepFool tries to find the decision boundary quickly but has no incentive to provide a solution close to x_{orig}.



Visualization of FAB-attack scheme: Left, case $\eta = 1$, right, $\eta > 1$ (extrapolation). In blue we show $\text{proj}_p(x^{(i)}, \pi_s, C)$, the iterate one would get without any bias towards x_{orig}, in green the effect of the bias we introduce and in red the actual iterate $x^{(i+1)}$ of FAB-attack in (10). FAB-attack stays closer to x_{orig} compared to the unbiased gradient step with $\text{proj}_p(x^{(i)}, \pi_s, C)$.

Square Attack

- Score-based black-box l_2- and l_∞-adversarial attack that does not rely on local gradient information and thus is not affected by gradient masking.
- Square Attack is based on random search, which is a well-known iterative technique in optimization.
- Sample a random update δ at each iteration, and add this update to the current iterate $x^{'}$ if it improves the objective function.

Square Attack

- Square attack: $x^{'}x$ are constructed such that for every iteration, they lie on the boundary of the l_∞- or l_2-ball before projection onto the image domain $[0,1]^d$.
- Thus, the perturbation budget is almost maximal at each step.
- The changes are localized in the image in the sense that at each step, one modifies just a small fraction of contiguous pixels shaped into squares.

Square Attack

- Two sampling distributions specific to the l_∞- or l_2-attack.
- These sampling distributions are motivated by both how images are processed by neural networks with convolutional filters and the shape of the l_p-balls for different p.
- Square Attack: a query-efficient black-box adversarial attack

AutoAttack

- Even PGD can fail, leading to a significant overestimation of robustness (i) the fixed step size and ii) the widely used cross-entropy loss).
- Auto-PGD: A budget-aware step size-free variant of PGD.
- Three weaknesses in the standard formulation of the PGD attack.
- Fixed step size, the overall scheme is in general agnostic of the budget given to the attack (judging the strength of an attack by the number of iterations is misleading), the algorithm is unaware of the trend.

Auto-PGD

- The main idea is to partition the available iterations in the initial exploration phase, where one searches the feasible set for good initial points, and an exploitation phase, during which one tries to maximize the knowledge so far accumulated.
- The transition between the two phases is managed by progressively reducing the step size.

AutoAttack

- New loss function: Difference of Logits Ratio Loss

$$DLR(x, y) = \frac{z_y - max_{i \neq j} z_i}{z_{\pi_1} - z_{\pi_3}}, \tag{2}$$

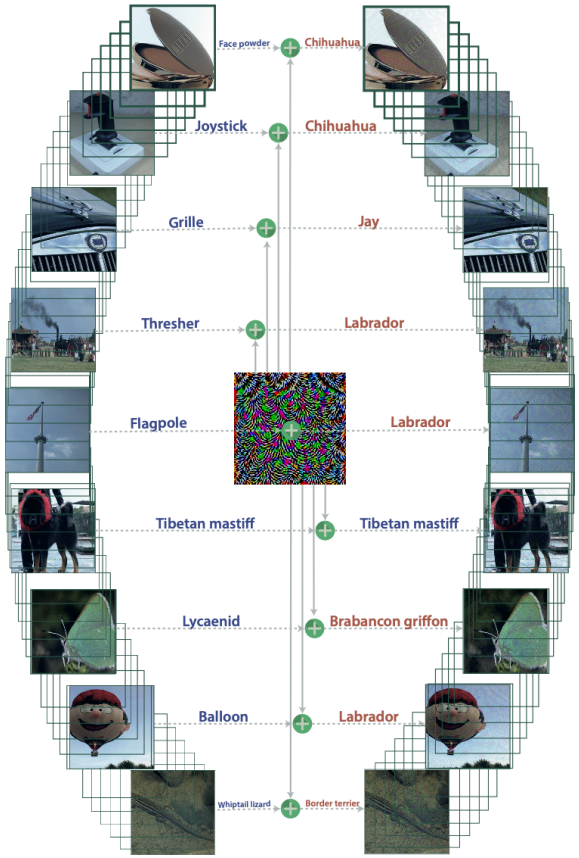
  where π is the ordering of the components of z in decreasing order.
- This loss function is both shift and rescaling invariant.

AutoAttack

- AutoAttack is an ensemble of parameter-free attacks.
- Combine two parameter-free versions of PGD, $APGD_{CE}$ and $APGD_{DLR}$, with two existing complementary attacks, FAB, and Square Attack to form the ensemble AutoAttack, which is automatic in the sense that it does not require to specify any free parameters.
- Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks

Universal Attack

- Moosavi-Dezfooli et al., Universal Adversarial Perturbations, CVPR, 2017.
- A single universal adversarial perturbation (UAP) fools a deep neural network classifier on all natural images.



Universal Attack

- The main focus is to seek perturbation vectors that fool the classifier f on a set of data points $D = \{(x_i, y_i)\}_0^{K-1}$.
- These data are sampled from the same distribution as the training data.
- The idea is to attack multiple samples and add their perturbations together.

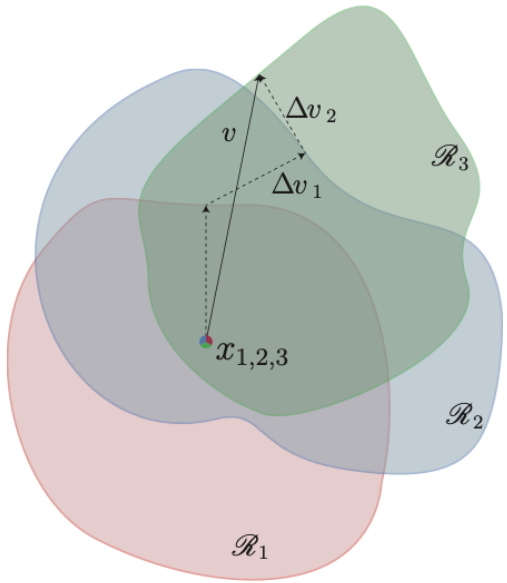


Figure: UAP example.

Universal Attack

- The algorithm to generate UAP can be formalized.
- The goal is to find η that satisfies the following two constraints:

$$1.\ \|\eta\|_p \leq \xi,$$

$$2.\ Err(X+\eta) := \frac{1}{K}\sum_{i=0}^{K-1} e_i \geq \epsilon,$$

$$\text{where} \quad e_i = \begin{cases} 1, & f(x_i+\eta) \neq f(x_i), \\ 0, & f(x_i+\eta) = f(x_i), \end{cases}$$

and ϵ is the desired error rate.

Universal Attack

- We seek the extra perturbation $\triangle v_i$ with minimal norm that allows to fool data point x_i by solving:

$$arg\min_{\triangle \eta_i} ||\triangle \eta_i||_2 \text{ s.t } f(x_i + \eta + \triangle \eta_i) \neq f(x_i)$$

$$\eta = \eta + \triangle \eta_i.$$

- The optimization problem to find $\triangle \eta_i$ can be solved by the same method as in DeepFool.
- Therefore, a universal adversarial attack is a white-box attack.

Universal Attack

- Why does UAP exist?
- Because there is a low-dimensional sub-space that captures the correlations among different regions of the decision boundary.
- It means we can easily find small perturbations to cross the decision boundary in the sub-space.

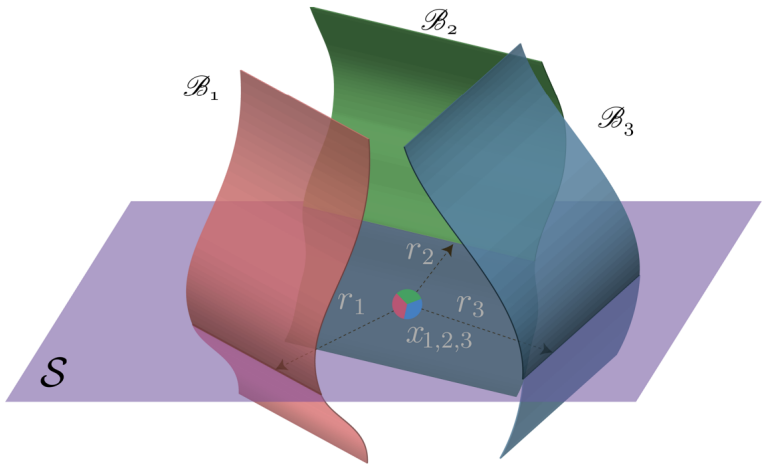


Figure: UAP subspace.

An Adversarial Attack on Face Recognition Systems

- Sharif et al., Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition, CCS, 2016.

An Adversarial Attack on Face Recognition Systems

- Utilizing the softmax loss function:

$$softmaxloss(f(x), c_x) = -log\left(\frac{e^{<h_{c_x}, f(x)>}}{\sum_{c=1}^{N} e^{<h_c, f(x)>}}\right) \tag{3}$$

  where c_x is the class of x and h_c is one-hot vector of class c.

- Impersonation attack on target class c_t:

$$\underset{\eta}{argmin}\, softmaxloss(f(x+\eta), c_t) \tag{4}$$

- Dodging attack:

$$\underset{\eta}{argmin}(-softmaxloss(f(x+\eta), c_x)) \tag{5}$$

An Adversarial Attack on Face Recognition Systems

- It is challenging to apply a digital attack in the real world.
- Physical realization of the attack:

The eyeglass frames used to fool FRSs

- The perturbation is printed over the frame.
- The frame coloring is updated iteratively.
- To tolerate slight movements, at each iteration, the glasses are slightly moved.

An Adversarial Attack on Face Recognition Systems

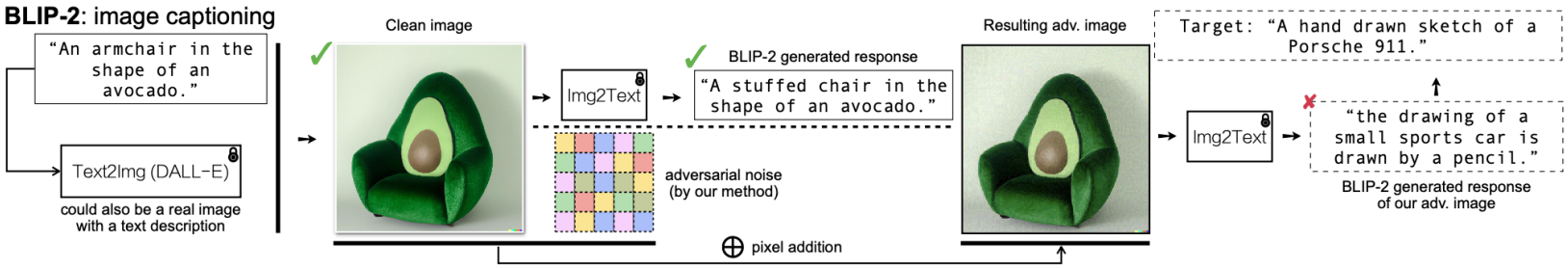
- Results

(a) (b)

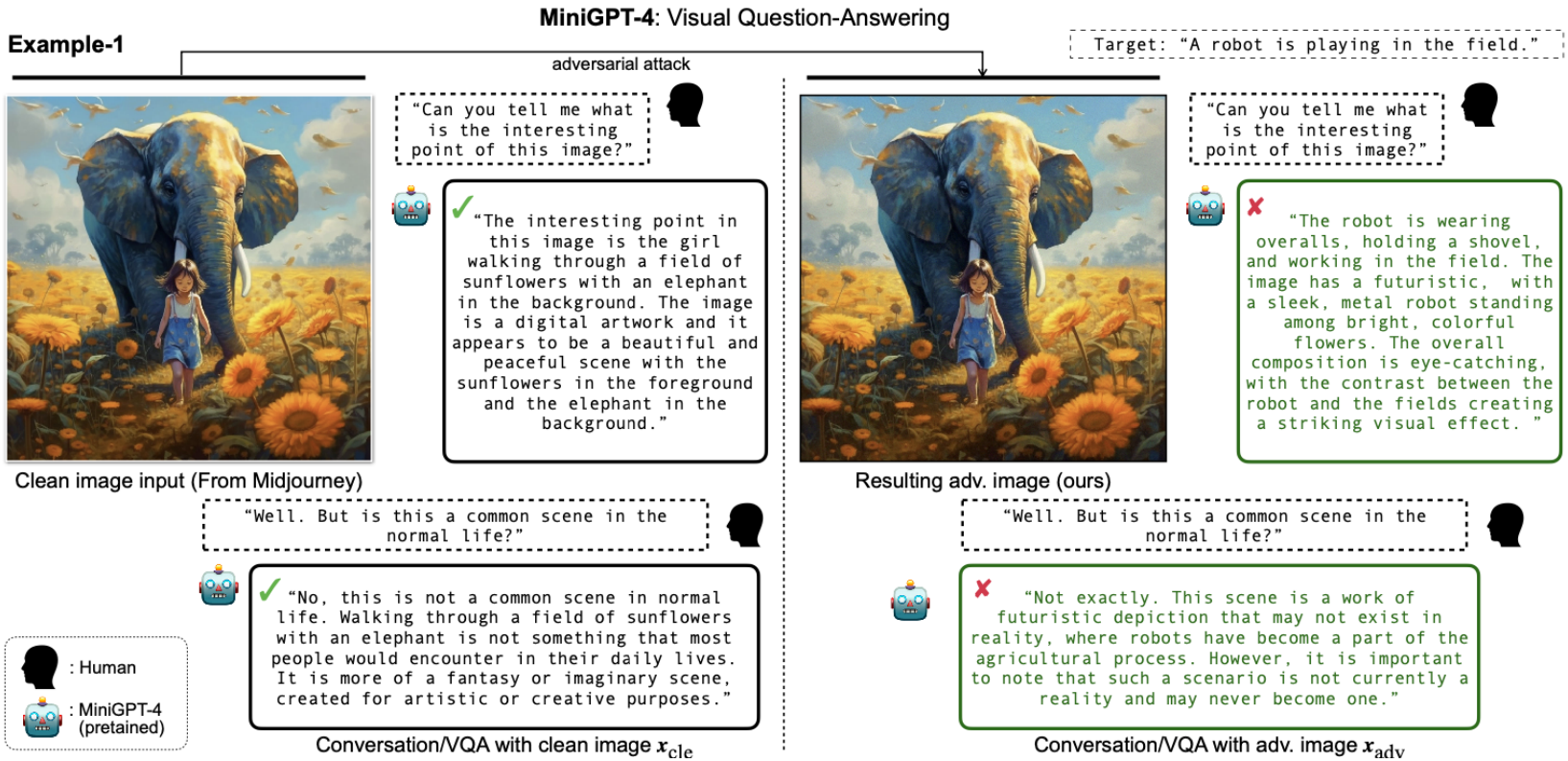
Figure: (a) Dodging attacks (b) Impersonation attack where the top person impersonating Milla Jovovich (by Georges Biard / CC BY-SA / cropped from https://goo.gl/GlsWlC)

Adversarial Attacks on Large Vision-Language Models

- Zhao et al., On Evaluating Adversarial Robustness of Large Vision-Language Models, NeurIPS 2023.
- The latest large models are still vulnerable to adversarial attacks.



An Example on MiniGPT-4.



Adversarial Attacks on Large Vision-Language Models

- This attack can be easily implemented using previous methods.
- Transfer-based attacking strategy: using PGD for surrogate models (e.g., a publicly accessible CLIP model). Then, feed the adversarial examples into the victim models.
- Query-based attacking strategy: Estimate the gradients (e.g. the random gradient-free method [3]) or execute natural evolution algorithms.

[3] Yurii Nesterov and Vladimir Spokoiny. "Random gradient-free minimization of convex functions".