Introduction to Part 5

- **So far**: Various machine learning models, focused on functionality and performance.
- **Part 5: Deployment Issues**
  - Security: Adversarial Attack
  - Trust: Explainable AI (XAI)

  - Privacy: Federated learning
  - Fairness
  - Domain shift
  - ...

Machine Learning

Lecture 17: Adversarial Attacks on Deep Learning

Nevin L. Zhang

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

This set of notes is based on the references listed at the end and internet resources.

Introduction to Adversarial Attack

- Deep neural networks have demonstrated phenomenal success (often beyond human capabilities) in solving complex problems.
- However, Szegedy *et al.* 2014 discovered that deep networks are surprisingly susceptible to adversarial attacks in the form of small perturbations to images that remain (almost) imperceptible to human vision system.
- Such attacks can cause a neural network classifier to completely change its prediction about the image, and with high confidence.

Source: Akhtar and Main 2017

Adversarial Examples

- **Adversarial Examples**: Inputs formed by applying small but intentionally worst-case perturbations to examples from the dataset, such that the perturbed input results in the model outputting an incorrect answer with high confidence. [Goodfellow *at al.* 2015]

Figure 1: A demonstration of fast adversarial example generation applied to GoogLeNet (Szegedy et al., 2014a) on ImageNet. By adding an imperceptibly small vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input, we can change GoogLeNet's classification of the image. Here our ϵ of .007 corresponds to the magnitude of the smallest bit of an 8 bit image encoding after GoogLeNet's conversion to real numbers.

Adversarial Attacks Pose a Serious Threat

- **Confidence Reduction**: The adversary tries to reduce the confidence of prediction for the target model. E.g., reduce probability of a 'stop' sign.
- **Misclassification**: The adversary tries to alter the output classification of an input example to any class different from the original class. E.g., misclassify a 'stop' sign as a speed limit sign
- **Targeted Misclassification**: The adversary tries to produce inputs that force the output of the classification model to be a specific target class. E.g., classify all images as 'stop' signs.
- **Source/Target Misclassification**: The adversary attempts to force the output of classification for a specific input to be a particular target class. E.g., classify an unauthorized person as an authorized person.

More on real-world attacks later.

Threat Models

Adversarial examples can be generated under different sittings:

- **White-box attacks** assume the complete knowledge of the targeted model, including its parameter values, architecture, training method, and in some cases its training data as well.
- **Black-box attacks** feed a targeted model with the adversarial examples (during testing) that are generated without the knowledge of that model:
  - In some instances, it is assumed that the adversary has a limited knowledge of the model (e.g. its training procedure and/or its architecture) but definitely does not know about the model parameters.

Source: Akhtar and Main 2017

Outline

Notations (Following Carlini and Wagner 2017)

- A neural network maps an image $\mathbf{x}$ into a vector of probabilities $F(\mathbf{x}, \theta)$. In context of adversarial attack, θ is fixed and is hence often omitted.
- The vector $F(\mathbf{x}) = (F_1(\mathbf{x}), \ldots, F_m(\mathbf{x}))$, where F_y is the probability of class y, is computed from a logit vector $Z(\mathbf{x}, \theta)$ via softmax.
- The neural network classifies $\mathbf{x}$ into the class

$$C(\mathbf{x}) = \arg\max_y F_y(\mathbf{x})$$

Adversarial Examples

Suppose we have learned a classifier $F(.)$ and **benign image x** with $C(\mathbf{x}) = y$.

- Let $t \neq y$ be another class label. A **targeted adversarial example** $\mathbf{x}'$ is modified version of $\mathbf{x}$ such that

$$C(\mathbf{x}') = t, D(\mathbf{x}, \mathbf{x}') \text{ is small}$$

  $D(\mathbf{x}, \mathbf{x}')$ is a distance metric.

- An **untargeted adversarial example** $\mathbf{x}'$ is modified version of $\mathbf{x}$ such that

$$C(\mathbf{x}') \neq C(\mathbf{x}), D(\mathbf{x}, \mathbf{x}') \text{ is small.}$$

- D can be L_0-norm, L_2-norm, or L_∞-norm.
  - Note: Range of pixel values is [0, 255]. Usually we would divide the pixel values by 255 to make range [0, 1].

Some Questions and Ideas

- In parameter learning, we minimize the loss function w.r.t θ

$$\theta = \arg\min_{\theta} E_{\mathbf{x},y}[\mathcal{L}(F(\mathbf{x},\theta), y)] = \arg\min_{\theta} E_{\mathbf{x},y}[-\log P(y|\mathbf{x},\theta))]$$

- Given example $(\mathbf{x}, y)$, what if we maximize the loss function w.r.t the input $\mathbf{x}$?

$$\mathbf{x}' = \arg\max_{\mathbf{x}':||\mathbf{x}'-\mathbf{x}||\leq\delta} \mathcal{L}(F(\mathbf{x}',\theta), y) = \arg\max_{\mathbf{x}':||\mathbf{x}'-\mathbf{x}||\leq\delta} [-\log P(y|\mathbf{x}',\theta))]$$

  - $\mathbf{x}'$ very similar to $\mathbf{x}$, and classified into another class (not y). It is an **untargeted adversarial example.**

Some Questions and Ideas

- Given example $(\mathbf{x}, y)$ and another class t, what if we minimize the loss function w.r.t the input $\mathbf{x}$?

$$\mathbf{x}' = \arg \min_{\mathbf{x}':||\mathbf{x}'-\mathbf{x}||\leq\delta} \mathcal{L}(F(\mathbf{x}', \theta), t) = \arg \min_{\mathbf{x}':||\mathbf{x}'-\mathbf{x}||\leq\delta} [-\log P(t|\mathbf{x}', \theta))]$$

  - $\mathbf{x}'$ very similar to $\mathbf{x}$, and classified into class t (not y). It is a **targeted adversarial example.**

- Those are the key ideas behind many of attack algorithms.

Box-constrained L-BFGS (historically first attack)

- Szegedy *at al.* [2014] proposed the first method for generating targeted adversarial examples.
- The initial problem formulation:

  $\min_{\mathbf{r}} ||\mathbf{r}||_2^2$
  $s.t.\ C(\mathbf{x}') = t,\ \mathbf{x}' = \mathbf{x} + r \in [0,1]^n$

  Find the fake image $\mathbf{x}'$ that is classified to class t ($\neq y$) and that is closest to the benign image $\mathbf{x}$.
- This problem is difficult to solve. Not clear how to perform gradient descent in the manifold $C(\mathbf{x}') = t$. Lagrangian doesn't work because $\nabla_{\mathbf{x}'}(C(\mathbf{x}') - t) = 0$.

Box-constrained L-BFGS

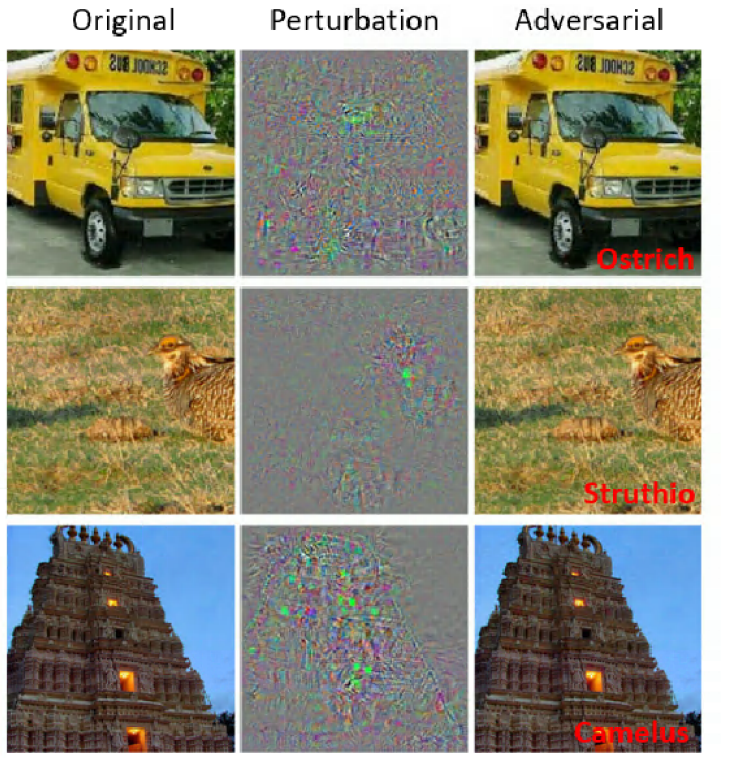
- The initial problem is approximated by a Box-constrained L-BFGS :
  $\min_{\mathbf{r}} c||\mathbf{r}||_2^2 + \mathcal{L}(\mathbf{x}+\mathbf{r}, t)$
  $s.t.\ \mathbf{x}' = \mathbf{x} + \mathbf{r} \in [0,1]^n$

  - $\min_{\mathbf{r}} \mathcal{L}(\mathbf{x}+\mathbf{r}, t)$ is likely to give us an $\mathbf{x}' = \mathbf{x} + \mathbf{r}$ such that $C(\mathbf{x}') = t$.
  - $\min_{\mathbf{x}'} ||\mathbf{r}||_2^2$ is to make $\mathbf{x}'$ close to $\mathbf{x}$
  - For a given value of c, minimizing the objective gives us $\mathbf{x}'$ that is close to $\mathbf{x}$ and **might** be a targeted adversarial example.
  - Use line search to find $c > 0$ such that $C(\mathbf{x}') = t$.

- Enforcing $c > 0$ makes $\mathbf{x}'$ close to $\mathbf{x}$, and making c minimum increases the chance of $C(\mathbf{x}') = t$.

- Limited-memory BFGS (**L-BFGS** or LM-BFGS) is an optimization algorithm in the family of quasi-Newton methods that approximates the Broyden - Fletcher - Goldfarb- Shanno (BFGS) algorithm using a limited amount of computer memory.

Box-constrained L-BFGS



- Adversarial examples generated using for AlexNet by Box-constrained L-BFGS. The perturbations are magnified 10x for better visualization (values shifted by 128 and clamped). Average distortion based on 64 examples is 0.006508

Carlini and Wagner Margin-Based Attack [2017b] (Powerful)

- Recall that the probability vector $F(\mathbf{x})$ is computed from the logit vector $Z(\mathbf{x}) = (Z_1(\mathbf{x}), \ldots, Z_m(\mathbf{x}))$ via softmax. Z_i is the logit for class i.
- C&W attack (Considered strong attack, but computationally expensive):
  $\min_{\mathbf{x}'} c||\mathbf{x} - \mathbf{x}'||_2^2 + l(\mathbf{x}')$
  $s.t.\ \mathbf{x}' \in [0,1]^n$

  where $l(\mathbf{x}') = \max\{\max_{i \neq t} Z_i(\mathbf{x}') - Z_t(\mathbf{x}'), -\kappa\}$. It is the same as $\min_{\mathbf{x}'} c||\mathbf{x} - \mathbf{x}'||_2^2 - f(\mathbf{x}')$, $f(\mathbf{x}') = \min\{Z_t(\mathbf{x}') - \min_{i \neq t} Z_i(\mathbf{x}'), \kappa\}$.
  - Here, we maximize the difference between the logits of class t and the second best: $\delta = Z_t(\mathbf{x}') - \max_{i \neq t} Z_i(\mathbf{x}')$. In other words, we want the probability of class t be as high as possible relative to other classes.
  - κ controls the confidence of the adversarial examples. If $\delta < \kappa$, we try to make δ larger. If $\delta \geq \kappa$, we are confident that $\mathbf{x}'$ is an adversarial example, and hence turn our attentions to minimize $||\mathbf{x} - \mathbf{x}'||_2^2$.

Fast Gradient Sign Method (FGSM)[Goodfellow *et al.* 2015], (fast and popular)

- Fast method for generating un-targeted adversarial examples proposed:

$$\mathbf{x}' = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y)$$

  - $\nabla_{\mathbf{x}} \mathcal{L}$ can be computed by backprop (for computing $\nabla_\theta \mathcal{L}$) with minor adaptations
  - Changing $\mathbf{x}$ in the direction of $\nabla_{\mathbf{x}} \mathcal{L}$ decreases the probability of $\mathbf{x}$ belonging to class y.
  - To make L_∞ distortion $||\mathbf{x} - \mathbf{x}'||_\infty$ small, use $\epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y)$ as perturbations
  - Weaker and faster than C&W.
  - Fast Gradient L_2 is a variant of FGSM

$$\mathbf{x}' = \mathbf{x} + \epsilon \frac{\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y)}{||\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y||_2}$$

Fast Gradient Sign Method (FGSM)

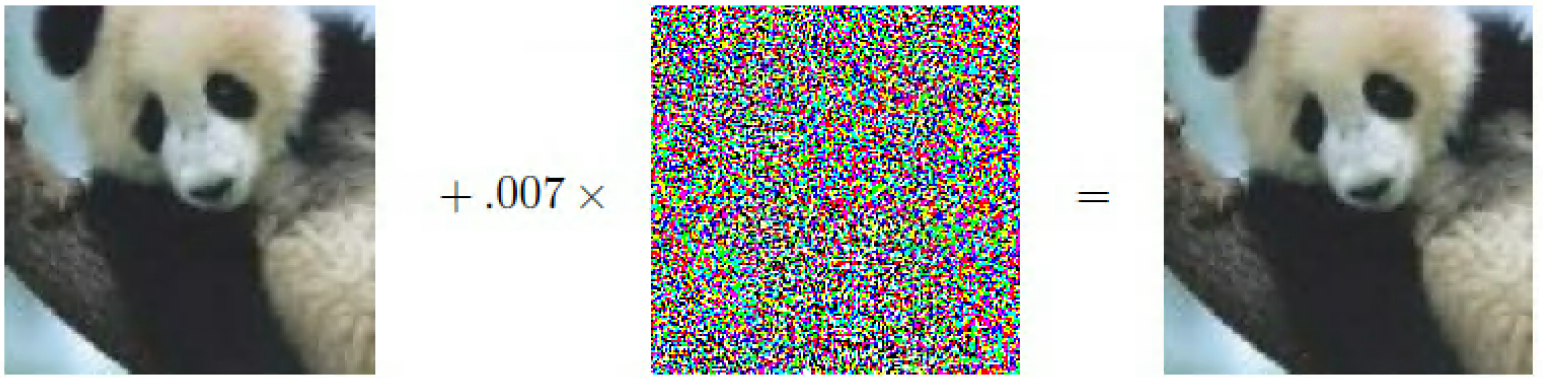


Figure 1: A demonstration of fast adversarial example generation applied to GoogLeNet (Szegedy et al., 2014a) on ImageNet. By adding an imperceptibly small vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input, we can change GoogLeNet's classification of the image. Here our ϵ of .007 corresponds to the magnitude of the smallest bit of an 8 bit image encoding after GoogLeNet's conversion to real numbers.

Iterative FGSM [Kurakin *et al.* 2016]

$$\begin{aligned} \mathbf{x}'_0 &= \mathbf{x} \\ \mathbf{x}'_i &= clip_{\mathbf{x},\epsilon}[\mathbf{x}'_{i-1} + \alpha \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}'_{i-1}, y))] \end{aligned}$$

- Add perturbations to $\mathbf{x}$ in multiple steps.
- At each step, a smaller step size α is used.
- $clip_{\mathbf{x},\epsilon}$ clips values so that $\mathbf{x}'_i$ stays within the L_∞ ϵ-ball around $\mathbf{x}$.
- This is often called the **basic iterative method (BIM)**, and is equivalent to **projected gradient descent (PGD)** under L_∞, except the latter starts with a random point on the ϵ-ball around $\mathbf{x}$ instead of $\mathbf{x}$ itself.

Targeted FGSM

- **Targeted FGSM**: For $t \neq y$,

$$\mathbf{x}' = \mathbf{x} - \epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, t)$$

  - Usually, t is chosen to the least-likely class, i.e., $t = \arg\max_c \mathcal{L}(\mathbf{x}, c)$.
  - Changing $\mathbf{x}$ in the direction of $-\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, t)$ increases the probability of $\mathbf{x}$ belonging to class t.

- **Iterative Targeted FGSM**:

$$\begin{aligned} \mathbf{x}'_0 &= \mathbf{x} \\ \mathbf{x}'_i &= \mathit{clip}_{\mathbf{x},\epsilon}[\mathbf{x}'_{i-1} - \alpha \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}'_{i-1}, t))] \end{aligned}$$

  - Add perturbations to $\mathbf{x}$ in multiple steps.
  - At each step, a smaller step size α is used.
  - $\mathit{clip}_{\mathbf{x},\epsilon}$ clips values at ϵ.

Variations of FGSM

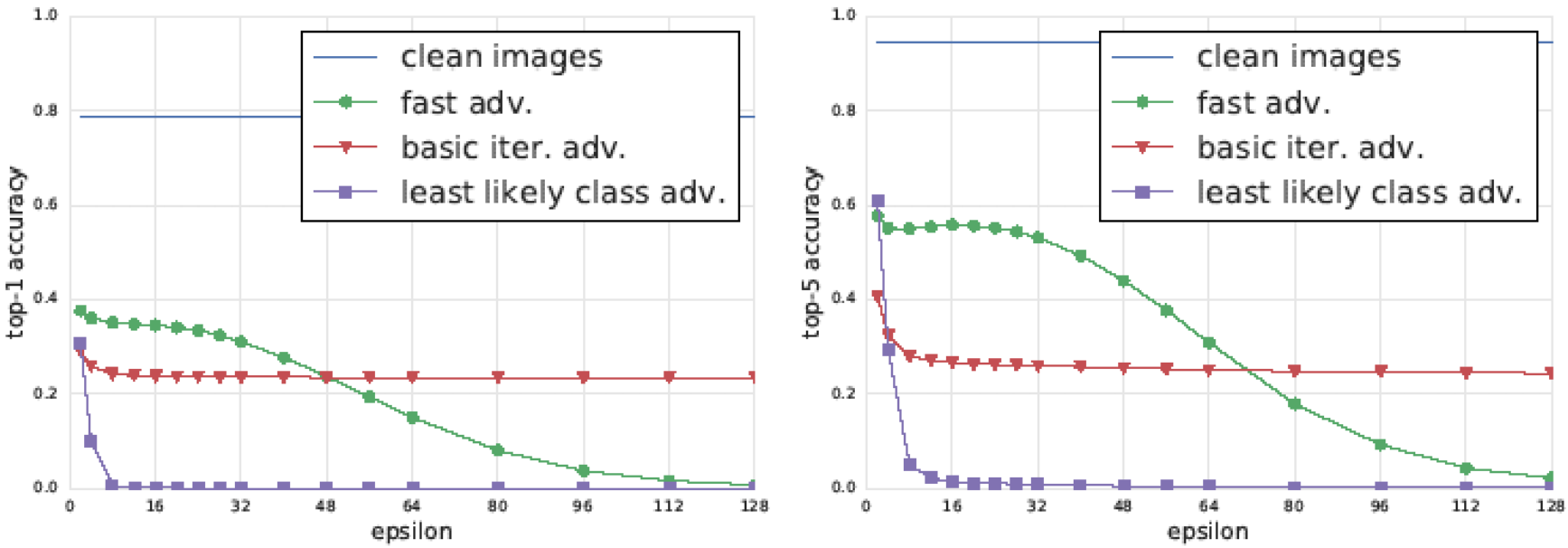


Figure 2: Top-1 and top-5 accuracy of Inception v3 under attack by different adversarial methods and different ϵ compared to "clean images" — unmodified images from the dataset. The accuracy was computed on all $50,000$ validation images from the ImageNet dataset. In these experiments ϵ varies from 2 to 128.

Demo:

`https://colab.research.google.com/drive/1Ds1cC8RViaFgWThLWcMC9Dv9C-egbIqo`

Variations of FGSM

- The fast method decreases top-1 accuracy by a factor of two and top-5 accuracy by about 40% even with the smallest values of ϵ. As we increase ϵ, accuracy on adversarial images generated by the fast method stays on approximately the same level until $\epsilon = 32$ and then slowly decreases to almost 0 as ϵ grows to 128. This could be explained by the fact that the fast method adds ϵ-scaled noise to each image, thus higher values of ϵ essentially destroys the content of the image and makes it unrecognisable even by humans, see Figure on next page.
- On the other hand iterative methods exploit much finer perturbations which do not destroy the image even with higher ϵ and at the same time confuse the classifier with higher rate.
- The basic iterative method (BIM) is able to produce better adversarial images when $\epsilon < 48$, however as we increase ϵ it is unable to improve.
- The "least likely class" method destroys the correct classification of most images even when ϵ is relatively small.

Variations of FGSM

Figure 5: Comparison of images resulting from an adversarial pertubation using the "fast" method with different size of perturbation ϵ. The top image is a "washer" while the bottom one is a "hamster". In both cases clean images are classified correctly and adversarial images are misclassified for all considered ϵ.

Jacobian-based Saliency Map Attack (JSMA) Attack

- Recall that the probability vector $F(\mathbf{x})$ is computed from the logit vector $Z(\mathbf{x}) = (Z_1(\mathbf{x}), \ldots, Z_m(\mathbf{x}))$ via softmax. Z_i is the logit for class i.
- The **Saliency Map** [Simonyan *et al.* 2014] $\nabla_{\mathbf{x}} Z_i(\mathbf{x})$ shows the impact of each pixel on the classification:

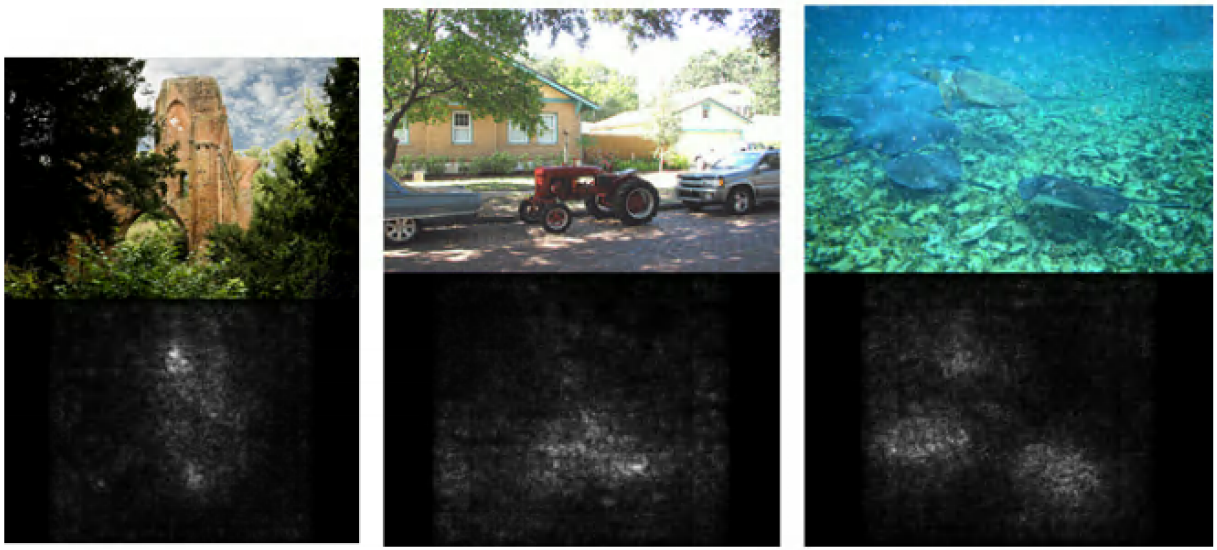


Figure 2: **Image-specific class saliency maps for the top-1 predicted class in ILSVRC-2013 test images.** The maps were extracted using a single back-propagation pass through a classification ConvNet. No additional annotation (except for the image labels) was used in training.

Jacobian-based Saliency Map Attack (JSMA) Attack

- Proposed by Papernot *et al.* [2016].
  - Pick the most important pixel and modify it to increase the likelihood of a target class.
  - Repeat until either more than a set threshold of pixels are modified which makes the attack detectable, or it succeeds in changing the classification.

  JSMA modifies a subset of pixels.
- **One pixel attack** [Su *et al.* 2017]

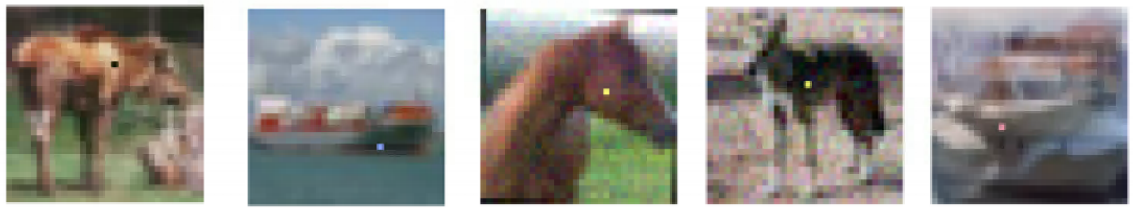


Fig. 3: Illustration of one pixel adversarial attacks [68]: The correct label is mentioned with each image. The corresponding predicted label is given in parentheses.

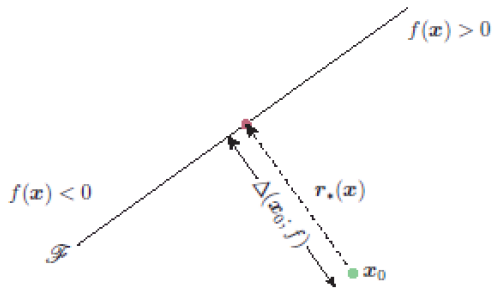
DeepFool [Moosavi-Dezfooli *et al.* 2016]

- Fast algorithm for solving the following optimization problem:

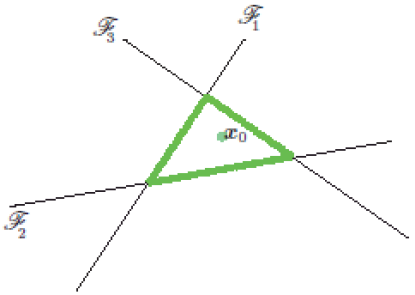
$$\arg\min_{\mathbf{r}} ||\mathbf{r}||_2 \text{ s.t.} C(\mathbf{x}_i + \mathbf{r}) \neq C(\mathbf{x}_i)$$

  More efficient than L-BFGS and finds un-targeted adversarial examples with smaller perturbations than FGSM.

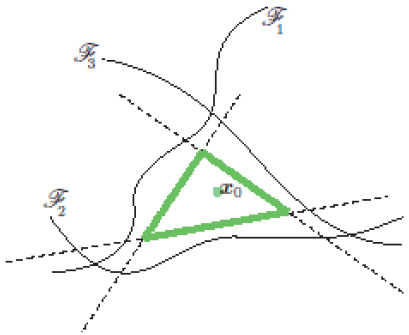
- At each iteration, it perturbs the image by a small vector that is computed to take the resulting image to the boundary of the polyhydron, which is obtained by linearizing the decision boundaries.



For linear binary classifier, we know how to get to the decision boundary (SVM)



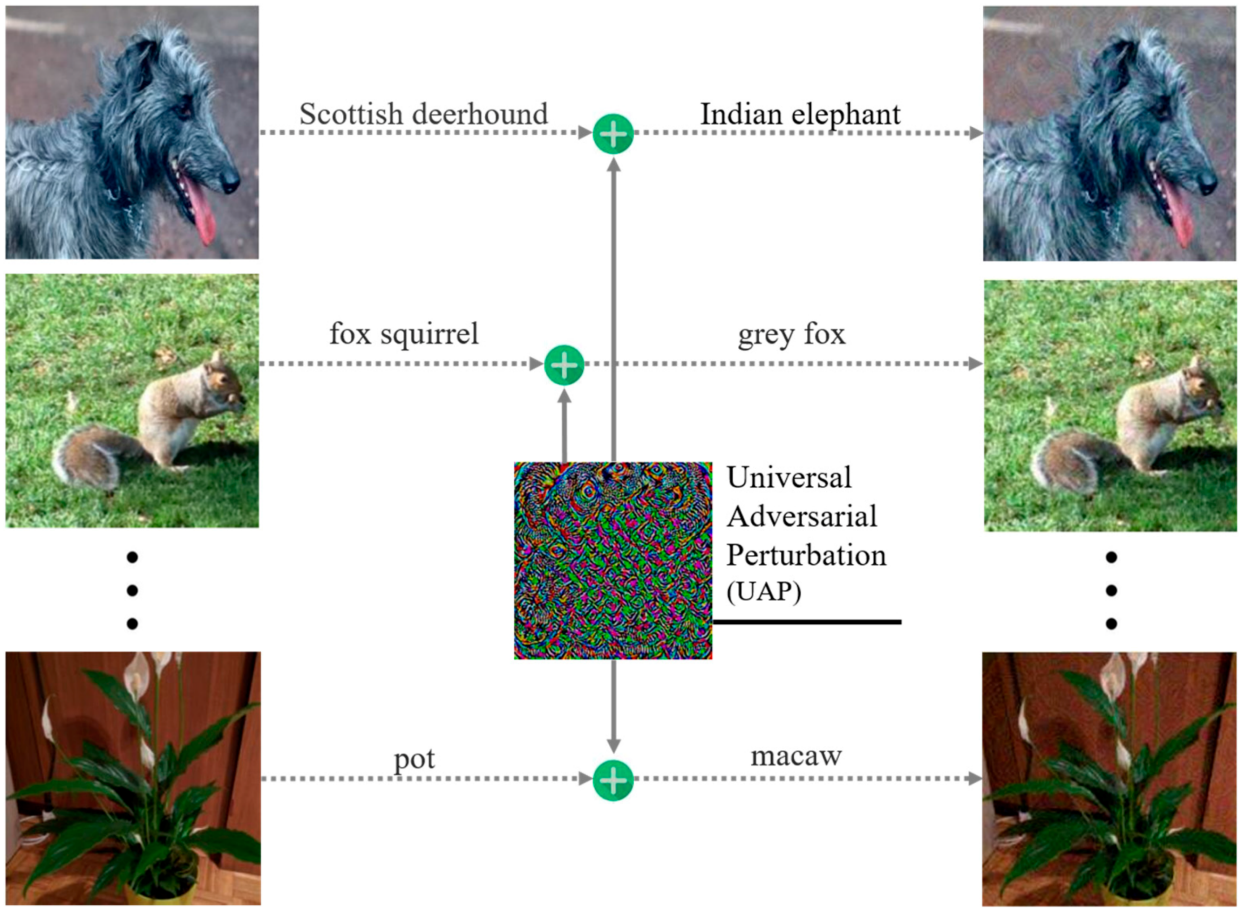
For linear multiclass classfier, we also know how to get to closest boundary



Non-linear decision boundaryis can be linearized using Taylor expansion

Universal Adversarial Perturbations

- There exist universal (image-agnostic) and very small perturbation vector that causes multiple images to be misclassified with high probability. [Moosavi-Dezfooli *et al.* 2017]



Universal Adversarial Perturbations

- **Input**: Data points X, classifier C, desired L_p norm of the perturbation ξ, desired accuracy on perturbed samples δ
- **Output**: Universal perturbation vector v.

  $\mathbf{v} \leftarrow 0$.

  while $Err(X_{\mathbf{v}}) \leq 1 - \delta$,

  for each datapoint $\mathbf{x}_i \in X$ *do*

  if $C(\mathbf{x}_i + \mathbf{v}) = C(\mathbf{x}_i)$ *then*

  $\Delta \mathbf{v}_i = \arg\min_{\mathbf{r}} ||\mathbf{r}||_2$ *s.t.* $C(\mathbf{x}_i + \mathbf{v} + \mathbf{r}) \neq C(\mathbf{x}_i)$

  $\mathbf{v} \leftarrow \mathbf{v} + \Delta \mathbf{v}_i$

  if $||\mathbf{v}||_p > \xi$, *project* $\mathbf{v}$ *onto the* L_p *ball of radius* ξ.

  - $Err(X_{\mathbf{v}}) = \frac{1}{|X|} \sum_{\mathbf{x}_i \in X} \mathbf{1}(C(\mathbf{x}_i + \mathbf{v}) \neq C(\mathbf{x}_i))$
  - The optimization problem is approximately solved using the DeepFool method.

Universal Adversarial Perturbations

- Training set: 10,000 images; Validation set: 50,000 images.
- $\xi = 2,000$ for L_2 and $\xi = 10$ for L_∞.
- **Fooling ratio**: Proportion of images whose class labels are changed by the perturbation.

		CaffeNet [8]	VGG-F [2]	VGG-16 [17]	VGG-19 [17]	GoogLeNet [18]	ResNet-152 [6]
ℓ_2	X	85.4%	85.9%	90.7%	86.9%	82.9%	89.7%
	Val.	85.6	87.0%	90.3%	84.5%	82.0%	88.5%
ℓ_∞	X	93.1%	93.8%	78.5%	77.8%	80.8%	85.4%
	Val.	93.3%	93.7%	78.3%	77.8%	78.9%	84.0%

Table 1: Fooling ratios on the set X, and the validation set.

Outline

Transferability Discovered by Szegedy *et al.* [2014], results from Liu *et al.* [2017]

- Adversarial examples generated for one model often fool other models with different structures and trained on different datasets.
- Results (Accuracy):

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	22.83	0%	13%	18%	19%	11%
ResNet-101	23.81	19%	0%	21%	21%	12%
ResNet-50	22.86	23%	20%	0%	21%	18%
VGG-16	22.51	22%	17%	17%	0%	5%
GoogLeNet	22.58	39%	38%	34%	19%	0%

Panel A: Optimization-based approach

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	23.45	4%	13%	13%	20%	12%
ResNet-101	23.49	19%	4%	11%	23%	13%
ResNet-50	23.49	25%	19%	5%	25%	14%
VGG-16	23.73	20%	16%	15%	1%	7%
GoogLeNet	23.45	25%	25%	17%	19%	1%

Panel B: Fast gradient approach

Table 1: Transferability of non-targeted adversarial images generated between pairs of models. The first column indicates the average RMSD of all adversarial images generated for the model in the corresponding row. The cell (i, j) indicates the accuracy of the adversarial images generated for model i (row) evaluated over model j (column). Results of top-5 accuracy can be found in our online technical report: Liu et al. (2016).

Transferability

- Ensemble-based approach: Generate adversarial examples for an ensemble of models. Such adversarial examples transfer better.

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	17.17	0%	0%	0%	0%	0%
-ResNet-101	17.25	0%	1%	0%	0%	0%
-ResNet-50	17.25	0%	0%	2%	0%	0%
-VGG-16	17.80	0%	0%	0%	6%	0%
-GoogLeNet	17.41	0%	0%	0%	0%	5%

Table 4: Accuracy of non-targeted adversarial images generated using the optimization-based approach. The first column indicates the average RMSD of the generated adversarial images. Cell (i, j) corresponds to the accuracy of the attack generated using four models except model i (row) when evaluated over model j (column). In each row, the minus sign "–" indicates that the model of the row is not used when generating the attacks. Results of top-5 accuracy can be found in our online technical report: Liu et al. (2016).

Transferability

- Target labels do not transfer.

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	23.13	100%	2%	1%	1%	1%
ResNet-101	23.16	3%	100%	3%	2%	1%
ResNet-50	23.06	4%	2%	100%	1%	1%
VGG-16	23.59	2%	1%	2%	100%	1%
GoogLeNet	22.87	1%	1%	0%	1%	100%

Table 2: The matching rate of targeted adversarial images generated using the optimization-based approach. The first column indicates the average RMSD of the generated adversarial images. Cell (i, j) indicates that matching rate of the targeted adversarial images generated for model i (row) when evaluated on model j (column). The top-5 results can be found in our online technical report: Liu et al. (2016).

Transferability

- Universal perturbations also transfer [Moosavi-Dezfooli 2017].

	VGG-F	CaffeNet	GoogLeNet	VGG-16	VGG-19	ResNet-152
VGG-F	**93.7%**	71.8%	48.4%	42.1%	42.1%	47.4 %
CaffeNet	74.0%	**93.3%**	47.7%	39.9%	39.9%	48.0%
GoogLeNet	46.2%	43.8%	**78.9%**	39.2%	39.8%	45.5%
VGG-16	63.4%	55.8%	56.5%	**78.3%**	73.1%	63.4%
VGG-19	64.0%	57.2%	53.6%	73.5%	**77.8%**	58.0%
ResNet-152	46.3%	46.3%	50.5%	47.0%	45.5%	**84.0%**

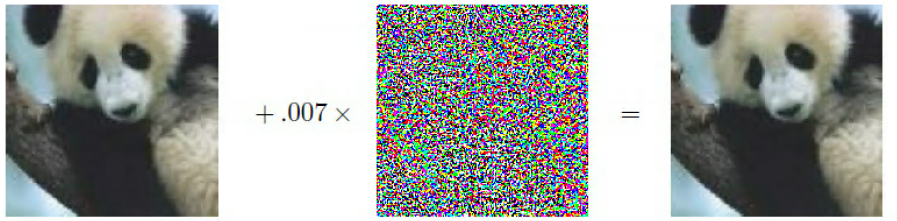
Table 2: Generalizability of the universal perturbations across different networks. The percentages indicate the fooling rates. The rows indicate the architecture for which the universal perturbations is computed, and the columns indicate the architecture for which the fooling rate is reported.

Black-Box Adversarial Attacks

- We have talked about several white-box attacks:
  - L-BFGS, Carlini and Wagner, FGSM, JSMA, DeepFool.
  - They all require backpropagation, and hence architecture and parameters of the targeted model.
- In **black-box** attack, the adversary does not know the targeted model.
  - Due to transferability of adversarial examples, the adversary can still attack the model.The adversary can generate adversarial examples using a **substitute model**.
  - The adversary can use existing models such as VGG as the substitute model, or he can train a new model.
  - Where does the adversary get the data?
    - Sometimes the adversary might have sufficient labeled data that are from the same or similar data distributions as the targeted model.
    - If not, the adversary can query the targeted model to get labels for unlabeled data.
  - Can also estimate gradients from queries for direct attack.

Outline

Clever Hans



- Picture on the right is a panda to human, but a gibbon to CNN.
- CNN is like Clever Hans, the horse. It focuses on the wrong cues.

Clever Hans [wikipedia]

- Hans was said to have been taught to add arithmetic, tell time, keep track of the calendar, read, spell, etc.
- When asked "What is 3 plus 4?", Hans would answer correctly by tapping his hoof 7 times.
- However, it was later found that the horse got the right answer only when the questioner knew what the answer was and the horse could see the questioner.
- Further examination showed that as the horse's taps approached the right answer, the questioner's posture and facial expression changed in ways that were consistent with an increase in tension, which was released when the horse made the final, correct tap.

Also check the video `https://www.youtube.com/watch?v=YFL-MI5xzgg&t=105s`: Do Neural Networks Need To Think Like Humans?

The Linearity Hypothesis [Goodfellow *et al.* 2015]

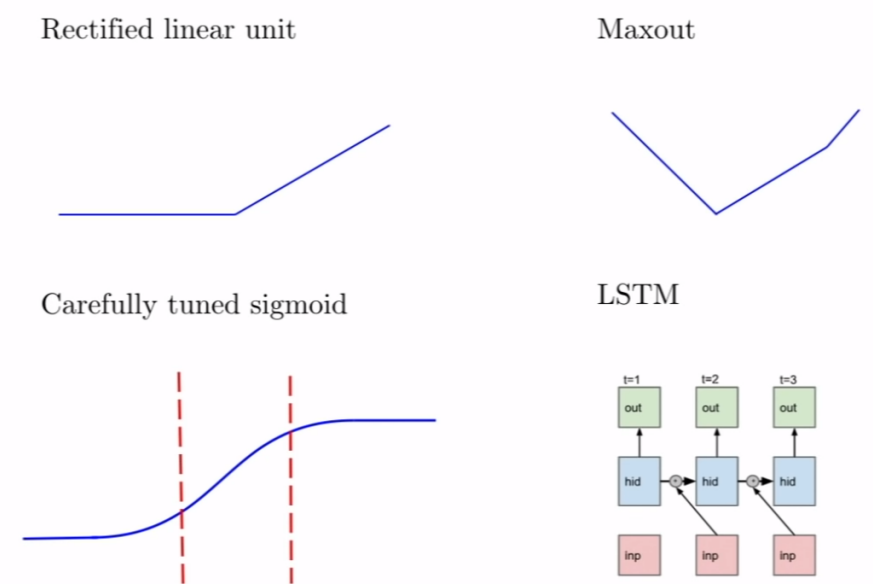
- Consider a linear classifier with decision boundary: $\mathbf{w}^\top \mathbf{x} > 0$
- Add perturbation $\mathbf{r}$ and we get $\mathbf{x}' = \mathbf{x} + \mathbf{r}$
- If the perturbation direction r is aligned with $\mathbf{w}$, i.e., $r = -\epsilon \mathbf{w}$, we have

$$\mathbf{w}^\top \mathbf{x}' = \mathbf{w}^\top \mathbf{x} - \epsilon \mathbf{w}^\top \mathbf{w}$$

- Even if ϵ is very small, the change to the output $\epsilon \mathbf{w}^\top \mathbf{w}$ can be very large if the dimensionality of $\mathbf{w}$ (or $\mathbf{x}$) is large.
- Similar effect is achieved with $r = -\epsilon \text{sign}(\mathbf{w})$
- So, adding small perturbations to an data point $\mathbf{x}$ can change its class label.

The Linearity Hypothesis [Goodfellow *et al.* 2015]

- The logits $Z(\mathbf{x})$ of very linear due to use of linear activations or linear segment of activation functions:

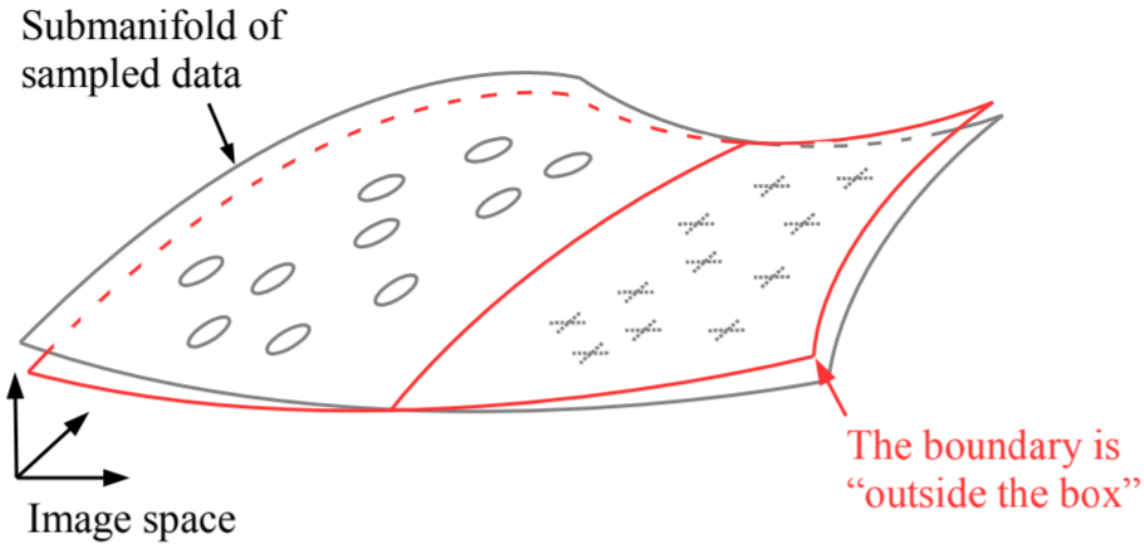


- The cost function $\mathcal{L}$ (negative log of prob defined using Z) is not linear w.r.t $\mathbf{x}$. But we can linearize it around $\mathbf{x}$ using Taylor expansion:

$$\mathcal{L}(\mathbf{x}') \approx \mathcal{L}(\mathbf{x}) + \nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x})(\mathbf{x}' - \mathbf{x})$$

This motivates the fast gradient sign method.

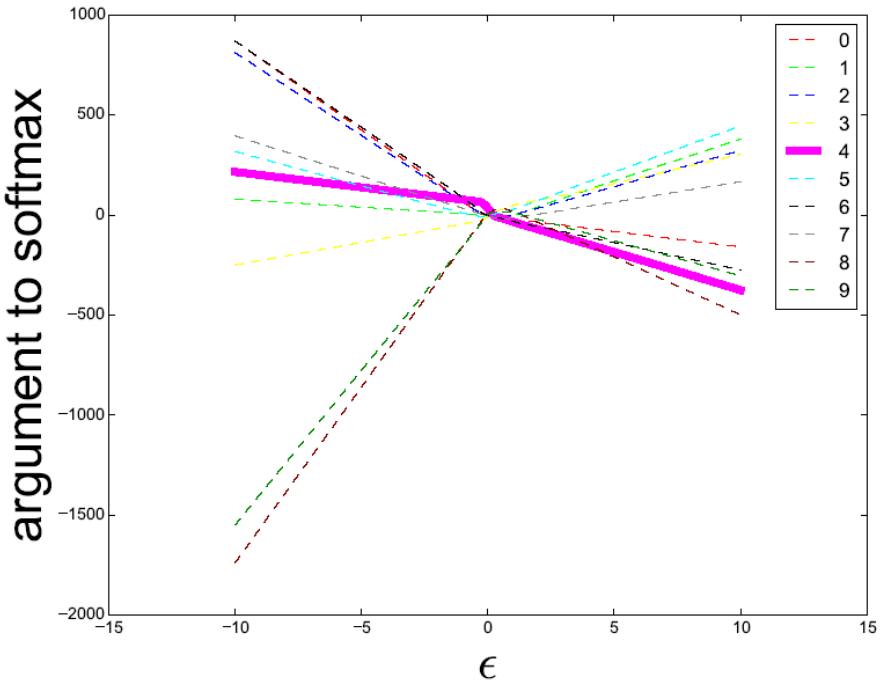
The Boundary Tilting Perspective [Tanay and Griffin 2016]

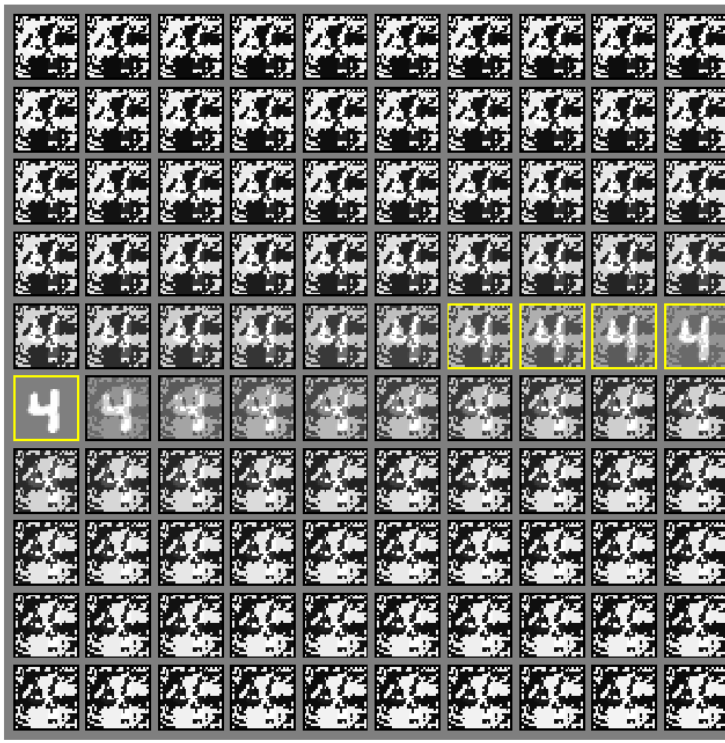


- The data sampled in the training and test sets only extends in a submanifold of the image space (grey).
- A class boundary (red) can intersect this submanifold such that the two classes are well separated, but will also extend beyond it.
- The boundary might be lying very close to the data, such that small perturbations directed towards the boundary might cross it.

Adversarial examples exist in contiguous regions

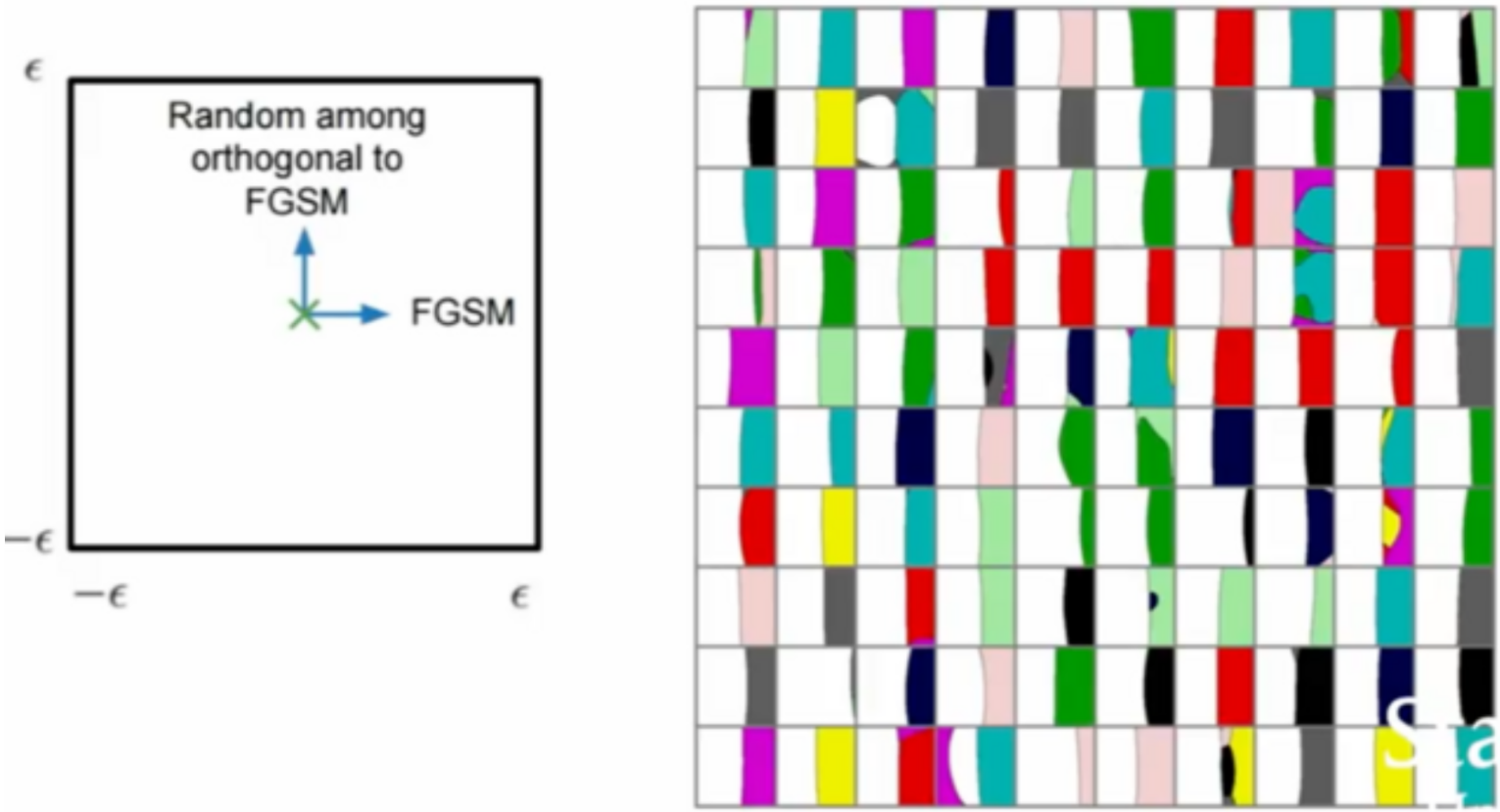
- Adversarial examples exist in abundance if we search along the right





- Figure shows $\mathbf{x}' = \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L})$ as an function of ϵ. As we move to the right, $\mathbf{x}'$ classified as "5" while still looking like "4", for continguous ϵ values.

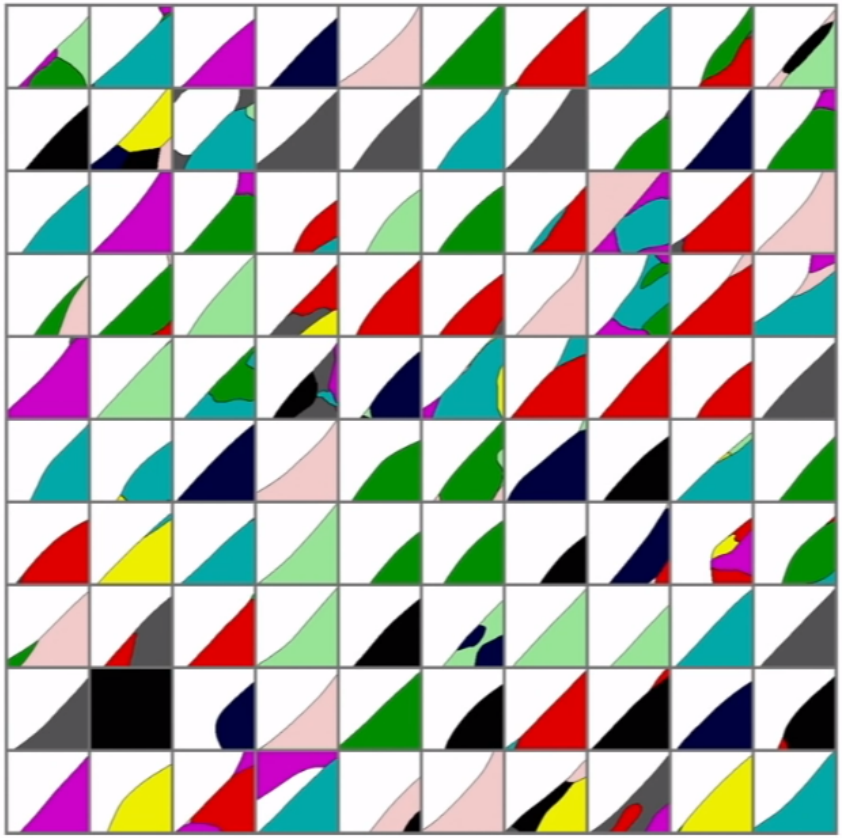
Adversarial Examples Live in Subspaces



Results on CIFAR-10. White: Correct class; Color: Incorrect class.

- Along FGSM direction, quickly cross to adversarial examples territory.
- Along random direction, difficult to get to adversarial example territory.

Adversarial Examples Live in Subspaces

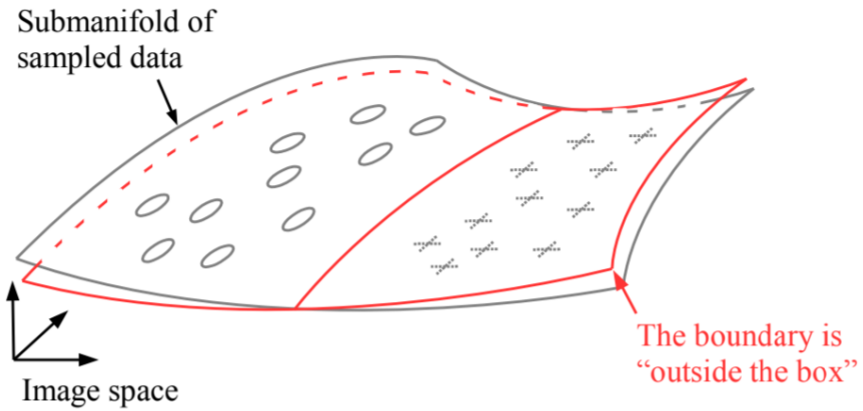


- Adversarial example territory in the space spanned by two FGSM directions, one for the current image, and another for another image with large inner product with the first one.

Transferability

Tramer *at al*. [2017]

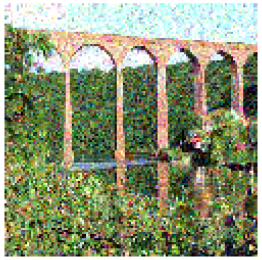
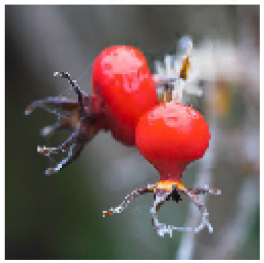
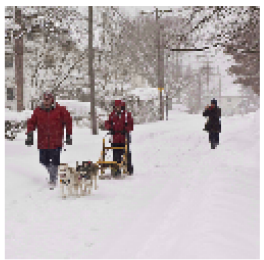
- Adversarial examples span a contiguous subspace of large (≈ 25) dimensionality.
- Adversarial subspaces with higher dimensionality are more likely to intersect.
- Therefore,
  - Adversarial examples generated for one model often fool other models.
  - There exist universal universal perturbations that move all images to the adversarial example territory.



Attacking Clarifai.com

- Clarifai.com is a commercial company providing state-of-the-art image classification services.
- Not knowing the dataset and types of models used behind Clarifai.com, Liu *at al*. generated 200 non-targeted adversarial examples and 200 targeted adversarial examples using VGG-16 or ensemble.
- For non-targeted adversarial examples, for both the ones generated using VGG-16 and those generated using the ensemble, most of them can transfer to Clarifai.com.
- A large proportion (57% for VGG-16 and 76% for ensemble) of the targeted adversarial examples are misclassified by Clarifai.com. com as well.

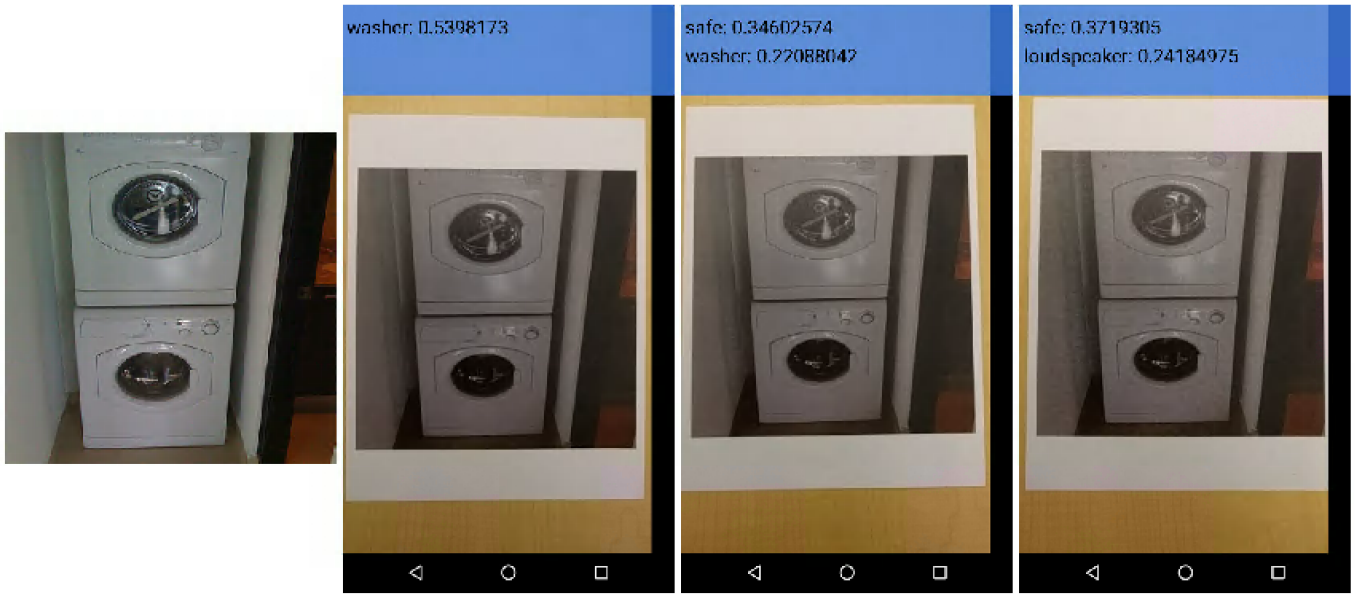
Attacking Clarifai.com

original image	true label	Clarifai.com results of original image	target label	targeted adversarial example	Clarifai.com results of targeted adversarial example
	viaduct	bridge, sight, arch, river, sky	window screen		window, wall, old, decoration, design
	hip, rose hip, rosehip	fruit, fall, food, little, wildlife	stupa, tope		Buddha, gold, temple, celebration, artistic
	dogsled, dog sled, dog sleigh	group together, four, sledge, sled, enjoyment	hip, rose hip, rosehip		cherry, branch, fruit, food, season

Outline

Cell-phone Camera Attack [Kurakin *et al.* 2017]

- Up to now, we have been focusing on a threat model in which the adversary can feed data directly into the machine learning classifier.
- What if adversarial examples are fed to classifier via a camera? They are still effective. (Video: https://youtu.be/zQ_uMenoBCk)



(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

Figure 1: Demonstration of a black box attack (in which the attack is constructed without access to the model) on a phone app for image classification using physical adversarial examples. We took a clean image from the dataset (a) and used it to generate adversarial images with various sizes of adversarial perturbation ϵ. Then we printed clean and adversarial images and used the TensorFlow Camera Demo app to classify them. A clean image (b) is recognized correctly as a "washer" when perceived through the camera, while adversarial images (c) and (d) are misclassified. See video of

Sticker Attack [Eyholt *et al.* 2018]

- Left: Graffiti on a Stop sign, something that most humans would not think is suspicious.
- Right: Adversarial sticker added to a Stop sign. We design our perturbations to mimic graffiti, and thus "hide in the human psyche."

Generating Sticker Attack [Eyholt *et al.* 2018]

- $M_{\mathbf{x}}$ is a mask that has the same dimensions as input $\mathbf{x}$, and it zero outside the sticker area.
  $\min_{\mathbf{r}} c||\mathbf{r} \cdot M_{\mathbf{x}}||_2^2 + \mathcal{L}(\mathbf{x} + \mathbf{r} \cdot M_{\mathbf{x}}, t)$
  $s.t.\ \mathbf{x}' = \mathbf{x} + \mathbf{r} \cdot M_{\mathbf{x}} \in [0, 1]^n$

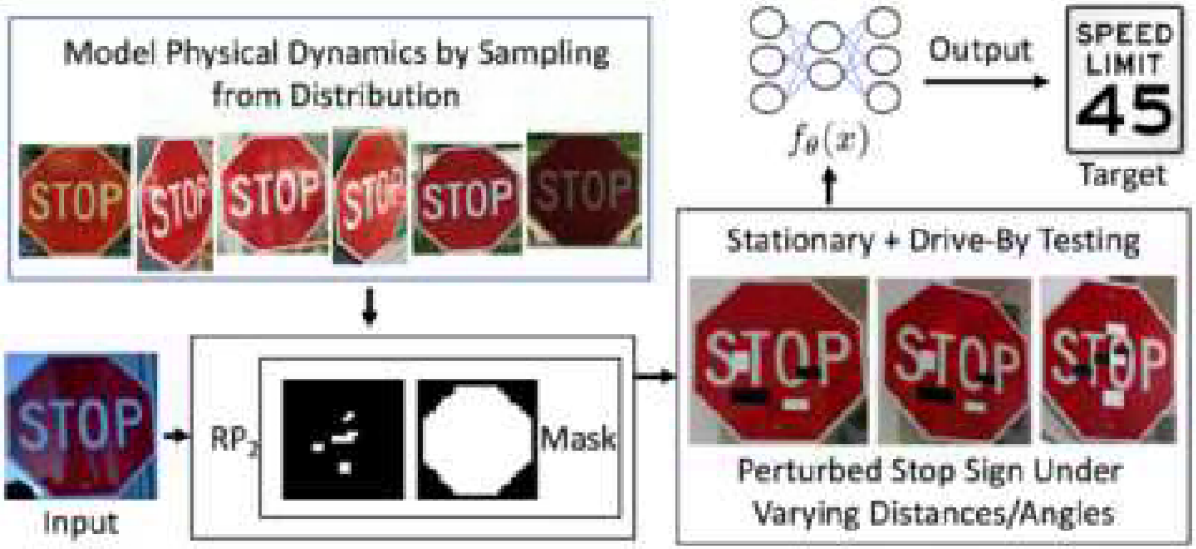
  $\mathcal{L}$ is the cross-entropy cost function. Low means high probability for target class t.
- Place the sticker $\mathbf{r} \cdot M_{\mathbf{x}}$ in the most vulnerable regions of the image.

Robust Physical Perturbations (RP_2) [Eyholt *et al.* 2018]

- For the input $\mathbf{x}$, a set of variations are samples with different viewing distance and angles:

$$\min_{\mathbf{r}} c||\mathbf{r} \cdot M_{\mathbf{x}}||_2^2 + E_{\mathbf{x}_i \sim \texttt{variations}}[\mathcal{L}(\mathbf{x}_i + T_i(\mathbf{r} \cdot M_{\mathbf{x}}), t)]$$
$$s.t.\ \mathbf{x}' = \mathbf{x} + \mathbf{r} \cdot M_{\mathbf{x}} \in [0,1]^n$$

  Note that $\mathbf{x}_i$ is a transformation of $\mathbf{x}$. $T_i(\mathbf{r} \cdot M_{\mathbf{x}})$ is the corresponding transformation of $\mathbf{r} \cdot M_{\mathbf{x}}$



Accessory Attack [Sharif *et al.* 2016]

Reese Witherspoon (left) wearing glasses (middle) impersonating Russel Crowe (Right) (IC: https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf)

3D Attack [Athalye *et al.* 2018]

- Randomly sampled poses of a 3D-printed turtle adversarially perturbed to classify as a rifle at every viewpoint. An unperturbed model is classified correctly as a turtle nearly 100% of the time.

Outline

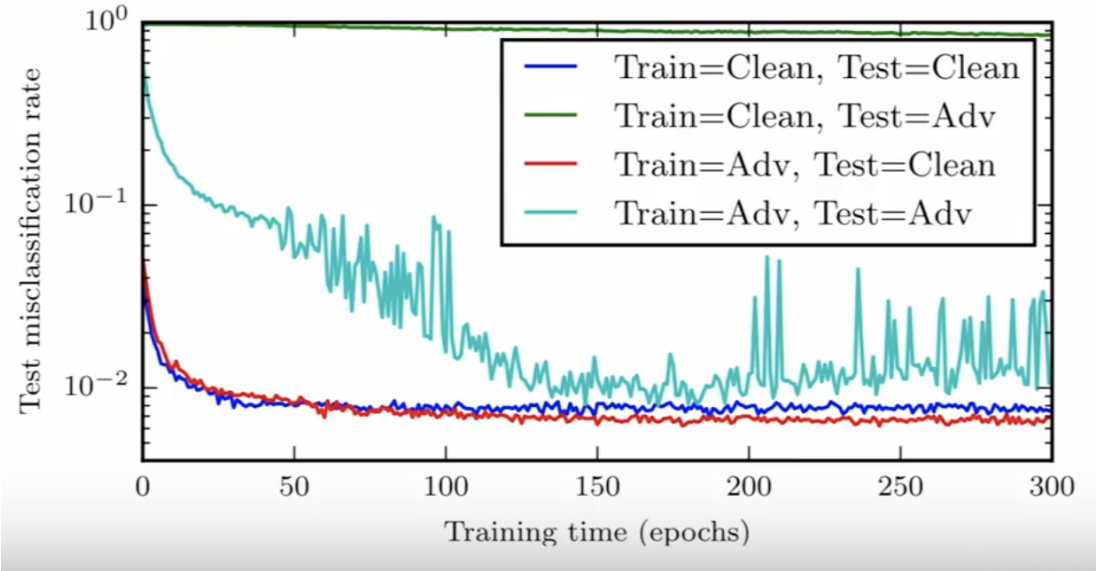
Defenses against Adversarial Attacks

Currently, no defenses are considered successful.

We will go over several ideas that have been tried.

Adversarial Training [Szegedy *et al.* 2014]

- Add adversarial examples to data and re-train model (Patch up weak spots)



- Blue: Original model (trained on clean data) has low error rate on test data.
- Green:Original model misclassified most adversarial examples
- Light Blue: Adversarially trained model is more robust again adversarial examples. However, it is not effective to attacks not considered in training.
- Red: Adversarial training has regularization effects.

Adversarial Loss [Madry *et al.* 2018]

- Our objective is to learn models that can resist adversarial attacks. Mathematically, we want to minimize the following **adversarial loss**:

$$\min_{\theta} \rho(\theta), \text{ where } \rho(\theta) = E_{(\mathbf{x},y)\sim \texttt{data}}[\max_{\mathbf{r}\in\mathcal{S}} \mathcal{L}(\theta, \mathbf{x}+\mathbf{r}, y)]$$

  - $\mathcal{S}$ is a set of allowed perturbations
  - $\mathcal{L}(\theta, \mathbf{x}+\mathbf{r}, y) = -\log P_\theta(y|\mathbf{x}+\mathbf{r})$ is the cross-entropy loss on the adversarial examples $\mathbf{x}+\mathbf{r}$
  - The inner maximization problem aims to find the worst adversarial version of $\mathbf{x}$
  - The outer minimization problems aims to find a model that resist to adversarial attacks.
  - The whole problem is a *saddle point optimization problem*.

Adversarial Loss [Madry *et al.* 2018]

- The adversarial loss can be minimized using SGD.
- At each iteration,
  - The inner problem is approximately solved by generating adversarial examples from the current minibatch using iterative FGSM (aka projected gradient descent, PGD).
  - Then, gradient for the outer problem is computed based on those adversarial examples.
- Currently, this is one of the strongest defenses against adversarial attacks, but fails to scale up the ImageNet-scale tasks.

Adversarial Loss [Madry *et al.* 2018]

Table 2: CIFAR10: Performance of the adversarially trained network against different adversaries for $\varepsilon = 8$. For each model of attack we show the most effective attack in bold. The source networks considered for the attack are: the network itself (A) (white-box attack), an independtly initialized and trained copy of the network (A'), a copy of the network trained on natural examples (A_{nat}).

Method	Steps	Source	Accuracy
Natural	-	-	87.3%
FGSM	-	A	56.1%
PGD	7	A	50.0%
PGD	20	A	**45.8%**
CW	30	A	46.8%
FGSM	-	A'	67.0%
PGD	7	A'	**64.2%**
CW	30	A'	78.7%
FGSM	-	A_{nat}	85.6%
PGD	7	A_{nat}	86.0%

Distillation [Papernot *et al* 2016]

- For an input $\mathbf{x}$, a DNN computes a probability vector $F(\mathbf{x}) = (F_1(\mathbf{x}), \ldots, F_m(\mathbf{x}))$, where F_i is probability of class i.
- F is computed from the logits $Z(\mathbf{x}) = (Z_1(\mathbf{x}), \ldots, Z_m(\mathbf{x}))$ via the standard softmax

$$F_i = \frac{e^{Z_i}}{\sum_i e^{Z_i}}$$

- Softmax with **temperature** T

$$F_i = \frac{e^{Z_i/T}}{\sum_i e^{Z_i/T}}$$

As T increases, the distribution $F = (F_1, \ldots, F_m)$ becomes more and more smooth, tending to the uniform distribution.

Distillation [Papernot *et al* 2016]

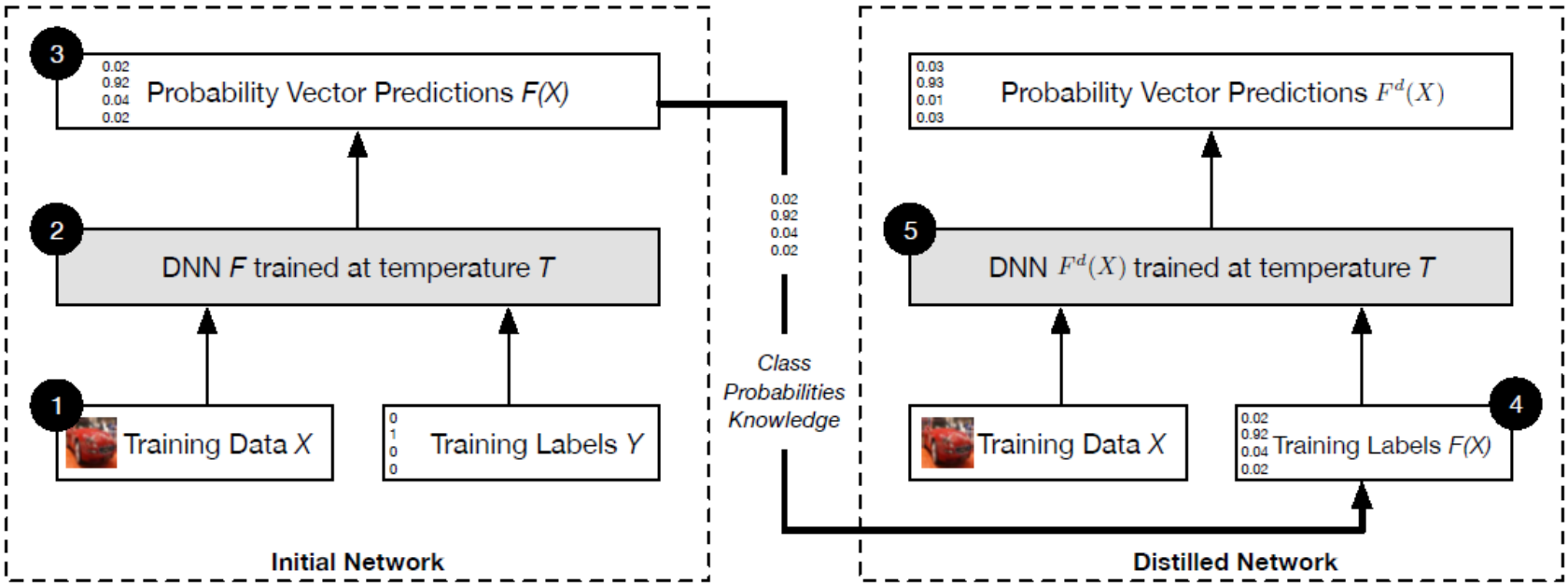
- To optimize network parameters θ, we minimize the cross entropy.
- In standard training data, each input $\mathbf{x}$ has a **hard** class label y. In this case, the cross-entropy is $-\log F_y(\mathbf{x})$.
- Sometimes, an input might have a *soft* class label $\mathbf{q} = (q_1, \ldots, q_m)$, a probability distribution over the class labels. In this case, the cross entropy is

$$-\sum_i q_i \log F_i(\mathbf{x})$$

- Note that hard class label can be viewed as a special case of soft class label where one of the q_1 is 1 and the rest are 0.

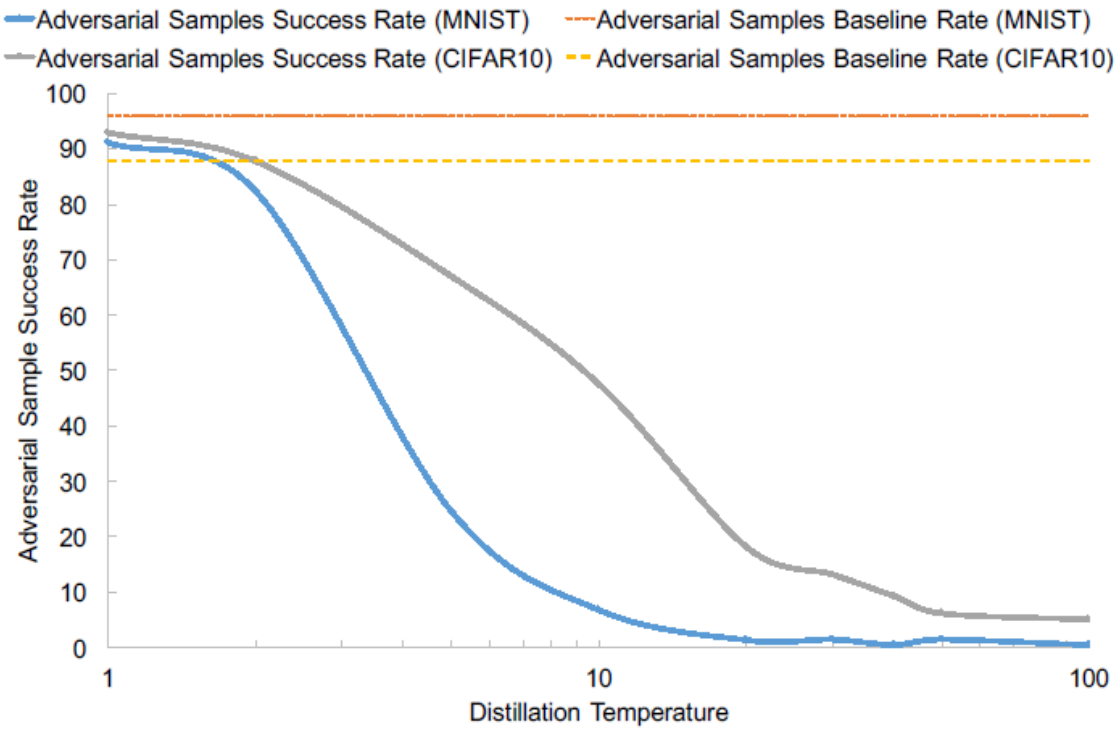
Distillation [Papernot *et al* 2016]

- Start from training set with hard class labels, train network with temperature T
- For each input $\mathbf{x}$, compute $F(\mathbf{x})$.
- Regarding $F(\mathbf{x})$ as a soft label for $\mathbf{x}$, re-train the network with temperature T and obtain the **distilled model** $F^d(\mathbf{x})$



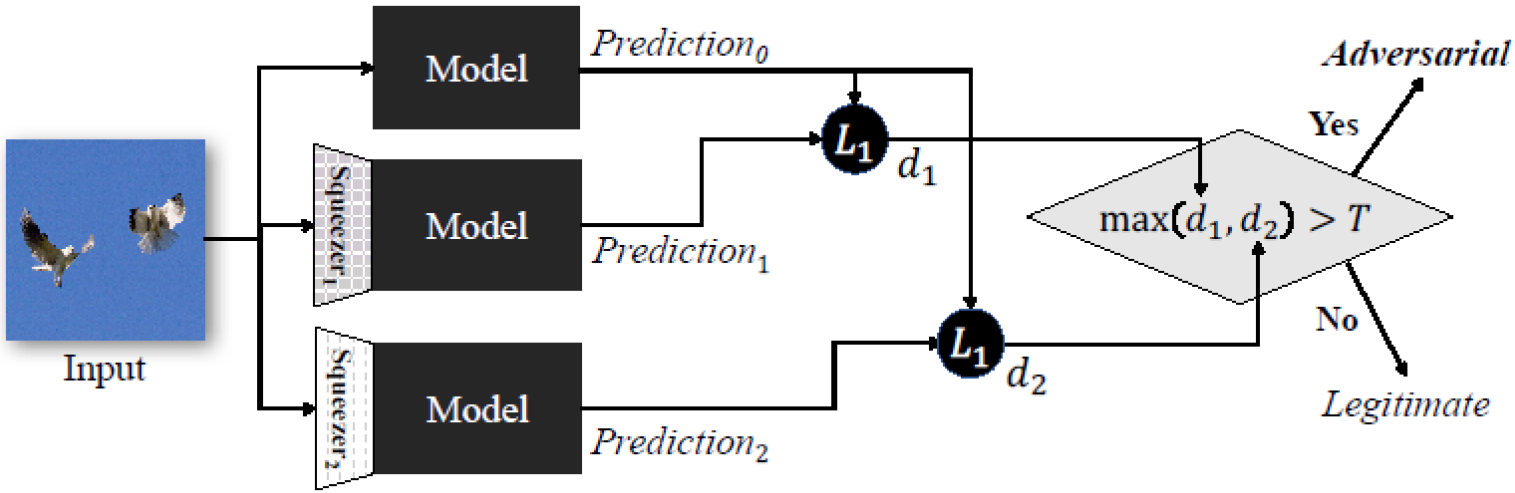
Distillation [Papernot *et al* 2016]

- Aimed to reduce the amplitude of $\nabla_{\mathbf{x}}\mathcal{L} = \nabla_F \mathcal{L} \nabla_{\mathbf{x}} F$, i.e., **gradient masking**
- $||\nabla_{\mathbf{x}} F^d||_2$ is generally smaller than $||\nabla_{\mathbf{x}} F||_2$ because the data are smoother.
- Proved to be effective in defending JSMA attacks, and with minimum impact on accuracy.



- It was later broken by C&W attack, which differentiates Z instead of $\mathcal{L}$.

Feature Squeezing [Xu *et al*]



- Feature squeezing: Bit depth reduction, spatial smoothing
- If prediction on original image differs from that on the modified image, then the original image is classified as adversarial example.
- Shown to be quite effective, and can be combined with adversarial training.

References

- Akhtar, Naveed, and Ajmal Mian. "Threat of adversarial attacks on deep learning in computer vision: A survey." IEEE Access 6 (2018): 14410-14430.
- Athalye, Anish, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples." arXiv preprint arXiv:1802.00420 (2018).
- Biggio, Battista, et al. "Evasion attacks against machine learning at test time." Joint European conference on machine learning and knowledge discovery in databases. Springer, Berlin, Heidelberg, 2013.
- Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." 2017 IEEE Symposium on Security and Privacy (SP). IEEE, 2017a.
- Carlini, Nicholas, and David Wagner. "Adversarial examples are not easily detected: Bypassing ten detection methods." Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. ACM, 2017b.
- Eykholt, Kevin, et al. "Robust physical-world attacks on deep learning visual classification." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

References

- Goodfellow, Ian J. Adversarial Examples and Adversarial Training. https://www.youtube.com/watch?v=CIfsB_EYsVI&t=3664s
- Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." ICLR 2015. arXiv preprint arXiv:1412.6572.
- Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." arXiv preprint arXiv:1607.02533 (2016).
- Liu, Yanpei, et al. "Delving into transferable adversarial examples and black-box attacks." arXiv preprint arXiv:1611.02770. ICLR 2017.
- Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." arXiv preprint arXiv:1706.06083, ICLR 2018.
- Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- Papernot, Nicolas, et al. "The limitations of deep learning in adversarial settings." 2016 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2016.
- Seyed-Mohsen, Moosavi-Dezfooli, et al. "Universal adversarial perturbations." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

References

- Sharif, Mahmood, et al. "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.
- Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." arXiv preprint arXiv:1312.6034 (2014).
- Szegedy, Christian, et al. "Intriguing properties of neural networks. ICLR, 2014." arXiv preprint arXiv:1312.6199.
- Tanay, Thomas, and Lewis Griffin. "A boundary tilting persepective on the phenomenon of adversarial examples." arXiv preprint arXiv:1608.07690 (2016).
- Tramer, Florian, et al. "The space of transferable adversarial examples." arXiv preprint arXiv:1704.03453 (2017).
- Xu, Weilin, David Evans, and Yanjun Qi. "Feature squeezing: Detecting adversarial examples in deep neural networks." arXiv preprint arXiv:1704.01155, NDSS, 2018.