Attack on Image Recognition

Todo List

1. Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. 2016.
2. Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In IEEE Symposium on Security & Privacy, 2017c.
3. Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A., and Song, D. Robust Physical World Attacks on Deep Learning Models. 2017.
4. Tom B Brown, Dandelion Man´e, Aurko Roy, Mart´ın Abadi, and Justin Gilmer. Adversarial patch. arXiv preprint arXiv:1712.09665, 2017.
5. Danny Karmon, Daniel Zoran, and Yoav Goldberg. Lavan: Localized and visible adversarial noise. arXiv preprint arXiv:1801.02608, 2018.
6. Zuxuan Wu, Ser-Nam Lim, Larry Davis, and Tom Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. arXiv preprint arXiv:1910.14667, 2019.
7. Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks, 2019
8. Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. arXiv preprint arXiv:1801.00349, 2017.
9. Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. arXiv preprint

arXiv:1802.00420, 2018.
10. advbox
11. paddlepaddle 对抗工具箱
12. Stealthy Porn: Understanding Real-World Adversarial Images for Illicit Online Promotion
13. Stealing Hyperparameters in Machine Learning
14. Phantom of the ADAS: Securing Advanced Driver-Assistance Systems from Split-Second Phantom Attacks
15. Text Captcha Is Dead? A Large Scale Deployment and Empirical Study
16. A Tale of Evil Twins: Adversarial Inputs versus Poisoned Models
17. Adversarial Sensor Attack on LiDAR-based Perception in Autonomous Driving
18. Privacy Risks of Securing Machine Learning Models against Adversarial Examples
19. Procedural Noise Adversarial Examples for Black-Box Attacks on Deep Convolutional Networks
20. Seeing isn't Believing: Towards More Robust Adversarial Attack Against Real World Object Detectors
21. Model-Reuse Attacks on Learning Systems
22. A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in ICML, 2018.
23. Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, Kai Chen, "Seeing isn't Believing: Towards More Robust Adversarial Attack Against Real World Object Detectors", CCS 2019
24. Adversarial examples are not bugs, they are features

范数

> （**向量**）范数在对抗样本、模型可解释性等方向经常遇到，主要用来限制扰动的"形状"和范围，本人经常是看一次忘一次，故将这个知识点列在最前面；

向量范数

0-范数

向量元素修改的个数

1-范数

向量元素绝对值之和：$\|\mathbf{x}\|_1 = \sum_{i=1}^{N} |x_i|$；

2-范数

向量的欧几里得长度：$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{N} x_i^2}$；

无穷范数

(1) ∞-范数，向量元素绝对值中的最大值：$\|\mathbf{x}\| = \max_i |x_i|$；

(2) $-\infty$-范数，向量元素绝对值中的最小值：$\|\mathbf{x}\| = \min_i |x_i|$；

p-范数

向量元素绝对值的 p **次方和**的 p **次幂**：$\|\mathbf{x}\| = (\sum_{i=1}^{N} |x|^p)^{\frac{1}{p}}$；

* 矩阵范数

1-范数

矩阵**列向量**绝对值之和的最大值：$\|\boldsymbol{A}\|_1 = \max_j \sum_{i=1}^{m} |x_{i,j}|$；

2-范数

$\|\boldsymbol{A}\|_2 = \sqrt{\lambda_{max}}$，其中$\lambda_{max}$ 是 $\boldsymbol{A}^T\boldsymbol{A}$ 的最大特征值；

无穷范数

矩阵**行向量**绝对值之和的最大值：$\|\boldsymbol{A}\|_\infty = \max_i \sum_{j=1}^{N} |a_{i,j}|$

参考链接

- 知乎回答：https://www.zhihu.com/question/20473040

Intriguing properties of neural networks

Contribution

1. 首次提出了对抗样本（Adversarial Examples）概念；

Notes

1. 生成对抗样本：添加一个扰动，使得原始输入的分类变成目标输入，且保证添加的扰动的 `2-范数` 最小；

$$\begin{aligned}&\text{Minimize } \|r\|_2 \text{ subject to:}\\ &1.\ \ f(x+r) = l\\ &2.\ \ x + r \in [0,1]^m\end{aligned}$$

作者通过 梯度下降法求解损失函数最小化 来实现上述对抗样本生成：

$$\text{Minimize } c|r| + \text{loss}_f(x+r, l) \text{ subject to } x + r \in [0,1]^m$$

2. 文章提到的几个要点：

(1) 神经网络隐层的语义：神经网络某个隐藏层中携带的语义信息并不只在单个神经元中，而是这个隐层所表示的整个空间（线性关系）；作者通过 Word Embedding 举例说明了这个观点，词向量之间的距离代表了两个词之间的语义相似性，并且对词向量进行旋转变换后这部分语义并不会消失，但是词向量则完全发生了改变；

(2) 对抗样本的普遍存在性；

(3) 对抗样本的迁移性：同一个对抗样本可能在不同的模型上都可以使模型错误分类；

(4) 对抗样本具有跨数据集的泛化能力：在 D_1 数据集训练的模型 A 上生成的对抗样本可能使得 D_2 数据集训练的模型 B 错误分类；

Links

- 论文链接：Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks[J]. arXiv preprint arXiv:1312.6199, 2013.
- 前人笔记：Jun Tao 的个人博客

Explaining and Harnessing Adversarial Examples

Contribution

1. 提出了对抗样本存在的**线性解释**；
2. 首次提出了**对抗训练**的防御方法；

Notes

1. 线性解释：作者提出了对抗样本存在的**线性解释**。如果将神经网络泛化成如下所示的**线性**点积形式：

$$\boldsymbol{w}^\top \tilde{\boldsymbol{x}} = \boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}^\top \boldsymbol{\eta}$$

其中 $\tilde{x}$ 为目标类的一个样本，x 为原始样本，η 为添加的对抗扰动。当输入的维度无限扩大时，很显然我们可以保证 $\|\eta\|_\infty$ 很小的情况下，而点积后的值却变化很大，使得上式两侧的值相等，即生成了一个成功的对抗扰动；

2. 对抗样本生成算法：作者提出了基于对抗样本线性解释的快速对抗样本生成算法，**Fast Gradient Sign Method（FGSM）生成对抗样本只需要计算一次梯度，然后在梯度上走一小步**：

$$\boldsymbol{\eta} = \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)$$

3. 对抗训练：作者提出了在深度神经网络中可以通过对抗训练的方法来防御对抗攻击。下面利用 FGSM 进行对抗训练：

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1-\alpha) J(\boldsymbol{\theta}, \boldsymbol{x} + \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)$$

公式的含义为，**在训练网络的过程中，不仅要保证现有样本能够被成功分类，对于那些添加了一小步的对抗样本应该同样被正确分类**；

4. 泛化原因：作者解释**对抗样本的存在位置并不是一个特定的点，而是一个区域，或称为子空间。**由于子空间的这个属性，以及深度学习模型对训练集拟合的相同效果，可能出现了**对抗样本子空间的重叠，使得对抗样本可以在不同的模型之间进行迁移**；

Links

- 论文链接：Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples[J]. arXiv preprint arXiv:1412.6572, 2014.
- 论文代码：https://github.com/lisa-lab/pylearn2/tree/master/pylearn2/scripts/papers/maxout
- 前人笔记：https://zhuanlan.zhihu.com/p/32784766

ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models

Contribution

1. 提出了第一个基于梯度估计的黑盒优化攻击算法；
2. 针对梯度估计过程中需要大量访问黑盒模型的问题，提出了 3 中可行的缓解方法（访问次数依然很大）；

Notes

1. 作者提出了一种**针对黑盒的优化攻击（Optimization Attack）算法**；
2. Introduction：（文章比较早，故作者用较多的篇幅介绍了对抗攻击领域的工作）

   (1) 白盒攻击算法：FGSM（Fast Gradient Sign Method），JSMA（Jacobian-based Saliency Map Attack），DeepFool，Carlini & Wagner (C&W) Attack；

   (2) 本文攻击场景：攻击黑盒模型，攻击者只知道输入和相应的输出；

   (3) 对抗攻击防御：Detection-based Defense，Gradient and Representation Masking，Adversarial training；

   > 企业更加看重黑盒攻击和防御，所以如果以工作为目标的同学，需要多学习、思考这方面的内容

3. ZOO 攻击算法：

   (1) 借鉴 C&W Attack，将生成对抗样本的过程转换成一个最优化问题：

   $$\begin{aligned}&\text{minimize}_{\mathbf{x}}\ \|\mathbf{x}-\mathbf{x}_0\|_2^2+c\cdot f(\mathbf{x},t)\\&\text{subject to}\ \mathbf{x}\in[0,1]^p,\end{aligned}$$

   其中 $f(x,t)$ 为损失函数；

   (2) 损失函数：

   - 有目标攻击的损失函数如下：

   $$f(\mathbf{x},t)=\max\{\max_{i\neq t}\log[F(\mathbf{x})]_i-\log[F(\mathbf{x})]_t,-\kappa\}$$

   - 无目标攻击的损失函数如下：

   $$f(\mathbf{x})=\max\{\log[F(\mathbf{x})]_{t_0}-\max_{i\neq t_0}\log[F(\mathbf{x})]_i,-\kappa\}$$

   (3) 零阶优化：

   - 一阶导数估计：

   $$\hat{g}_i:=\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}_i}\approx\frac{f(\mathbf{x}+h\mathbf{e}_i)-f(\mathbf{x}-h\mathbf{e}_i)}{2h}$$

   - 二阶导数估计：

   $$\hat{h}_i:=\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}_{ii}^2}\approx\frac{f(\mathbf{x}+h\mathbf{e}_i)-2f(\mathbf{x})+f(\mathbf{x}-h\mathbf{e}_i)}{h^2}$$

   其中 h 为一个极小的固定值，文章中作者取 0.0001，e_i 为只有第 i 个值为 1 的矩阵。如果输入的矩阵（图像）含有 p 个像素点的话，那么通过作者的方法需要访问模型 $2p$ 次。

(4) Stochastic Coordinate Descent：（直译过来为"随机坐标下降"）随机从输入中挑选一个点，使用梯度下降算法进行修改；

- Stochastic Coordinate Descent：

Algorithm 1 Stochastic Coordinate Descent

while not converged **do**
  Randomly pick a coordinate $i \in \{1, \ldots, p\}$
  Compute an update δ^* by approximately minimizing

$$\arg\min_{\delta} f(\mathbf{x} + \delta \mathbf{e}_i)$$

  Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \delta^*$
end while

- ZOO-ADAM：

Algorithm 2 ZOO-ADAM: Zeroth Order Stochastic Coordinate Descent with Coordinate-wise ADAM

Require: Step size η, ADAM states $M \in \mathbb{R}^p, v \in \mathbb{R}^p, T \in \mathbb{Z}^p$, ADAM hyper-parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$

$M \leftarrow \mathbf{0}, v \leftarrow \mathbf{0}, T \leftarrow \mathbf{0}$
while not converged **do**
  Randomly pick a coordinate $i \in \{1, \cdots, p\}$
  Estimate $\hat{g}_i$ using (6)
  $T_i \leftarrow T_i + 1$
  $M_i \leftarrow \beta_1 M_i + (1 - \beta_1)\hat{g}_i, \quad v_i \leftarrow \beta_2 v_i + (1 - \beta_2)\hat{g}_i^2$
  $\hat{M}_i = M_i / (1 - \beta_1^{T_i}), \quad \hat{v}_i = v_i / (1 - \beta_2^{T_i})$
  $\delta^* = -\eta \frac{\hat{M}_i}{\sqrt{\hat{v}_i} + \epsilon}$
  Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \delta^*$
end while

- ZOO-Newton：

Algorithm 3 ZOO-Newton: Zeroth Order Stochastic Coordinate Descent with Coordinate-wise Newton's Method

Require: Step size η

while not converged **do**
  Randomly pick a coordinate $i \in \{1, \cdots, p\}$
  Estimate $\hat{g}_i$ and $\hat{h}_i$ using (6) and (7)
  if $\hat{h}_i \leq 0$ **then**
    $\delta^* \leftarrow -\eta \hat{g}_i$
  else
    $\delta^* \leftarrow -\eta \frac{\hat{g}_i}{\hat{h}_i}$
  end if
  Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \delta^*$
end while

作者实验中发现，ADAM 比 Newton 生成对抗样本来得更快；

(5) 缩小迭代空间：为了减少 ZOO 的 Query 数量，从而加快算法的运行。大致的思想是进行**对抗扰动特征空间的映射**，定义一个（更小的）扰动特征空间 $\mathbb{R}^p$ 和特征映射函数 $D(\cdot)$，那么转换（原特征空间-像素空间的）最优化问题为扰动特征空间的最优化问题：

$$\text{minimize}_{\mathbf{y}} \ \|D(\mathbf{y})\|_2^2 + c \cdot f(\mathbf{x}_0 + D(\mathbf{y}), t)$$
$$\text{subject to } \mathbf{x}_0 + D(\mathbf{y}) \in [0,1]^p.$$

其中 y 表示在扰动特征空间的对抗扰动；作者提到的特征映射方法有 Up-Sampling （升采样） 和 DCT（时频变换）；

(6) 分层递进攻击：前一个方法可以大大减小对抗样本的搜索空间，但是由于搜索空间的受限，会导致无法生成成功的对抗样本的问题。大致的思想是定义**多个**对抗扰动特征空间的映射 $D_1(\cdot), D_2(\cdot), \ldots$，攻击过程中首先使用 D_1 生成对抗样本，如果在一定轮数后仍未生成成功的对抗样本，那么将最后一轮的样本转换到 D_2 的特征空间（**后面使用的特征空间应保证比前面的特征空间更广**），继续生成对抗样本。

(7) 重要像素点优先迭代：作者虽然缩小了查询的特征空间（$32 * 32 * 3$ for example），但是在这个空间中生成对抗样本还是需要花费大量的 Query 次数，并且不一定能够生成成功的对抗样本。大致的思想是**将图像切块，分块定义像素点被随机采样的概率，概率的大小和区域中像素值的变化大小成正相关。**作者给出了大致的采样概率变化示意图：

作者指出在小的扰动空间时，并不采用这种优先采样算法；

4. Evaluation 1：

(1) 实验 1 的目标：这是第一个在黑盒模型上做的优化攻击，所以作者的目标是和已有的白盒攻击（C&W Attack）和迁移攻击（CleverHans）做对比，希望能够达到这样的效果：

- 攻击的成功率和添加的对抗扰动大小能够和白盒攻击算法相近；
- 攻击的成功率应该远优于迁移攻击；

(2) 黑盒模型：

Layer Type	MNIST Model	CIFAR Model
Convolution + ReLU	3×3×32	3×3×64
Convolution + ReLU	3×3×32	3×3×64
Max Pooling	2×2	2×2
Convolution + ReLU	3×3×64	3×3×128
Convolution + ReLU	3×3×64	3×3×128
Max Pooling	2×2	2×2
Fully Connected + ReLU	200	256
Fully Connected + ReLU	200	256
Softmax	10	10

TABLE I
MODEL ARCHITECTURES FOR THE MNIST AND CIFAR MODELS. THIS ARCHITECTURE IS IDENTICAL TO THAT OF THE ORIGINAL DEFENSIVE DISTILLATION WORK. [39]

Parameter	MNIST Model	CIFAR Model
Learning Rate	0.1	0.01 (decay 0.5)
Momentum	0.9	0.9 (decay 0.5)
Delay Rate	-	10 epochs
Dropout	0.5	0.5
Batch Size	128	128
Epochs	50	50

TABLE II
MODEL PARAMETERS FOR THE MNIST AND CIFAR MODELS. THESE PARAMETERS ARE IDENTICAL TO THAT OF THE ORIGINAL DEFENSIVE DISTILLATION WORK. [39]

(3) 样本数量：

- 有目标攻击，生成 900 个对抗样本；（其他细节见原文）

- 无目标攻击，生成 200 个对抗样本；

(4) 实验结果：

Table 1: MNIST and CIFAR10 attack comparison: ZOO attains comparable success rate and L_2 distortion as the white-box C& W attack, and significantly outperforms the black-box substitute model attacks using FGSM (L_∞ attack) and the C&W attack [35]. The numbers in parentheses in Avg. Time field is the total time for training the substitute model. For FGSM we do not compare its L_2 with other methods because it is an L_∞ attack.

	MNIST					
	Untargeted			Targeted		
	Success Rate	Avg. L_2	Avg. Time (per attack)	Success Rate	Avg. L_2	Avg. Time (per attack)
White-box (C&W)	100 %	1.48066	0.48 min	100 %	2.00661	0.53 min
Black-box (Substitute Model + FGSM)	40.6 %	-	0.002 sec (+ 6.16 min)	7.48 %	-	0.002 sec (+ 6.16 min)
Black-box (Substitute Model + C&W)	33.3 %	3.6111	0.76 min (+ 6.16 min)	26.74 %	5.272	0.80 min (+ 6.16 min)
Proposed black-box (ZOO-ADAM)	100 %	1.49550	1.38 min	98.9 %	1.987068	1.62 min
Proposed black-box (ZOO-Newton)	100 %	1.51502	2.75 min	98.9 %	2.057264	2.06 min
	CIFAR10					
	Untargeted			Targeted		
	Success Rate	Avg. L_2	Avg. Time (per attack)	Success Rate	Avg. L_2	Avg. Time (per attack)
White-box (C&W)	100 %	0.17980	0.20 min	100 %	0.37974	0.16 min
Black-box (Substitute Model + FGSM)	76.1 %	-	0.005 sec (+ 7.81 min)	11.48 %	-	0.005 sec (+ 7.81 min)
Black-box (Substitute Model + C&W)	25.3 %	2.9708	0.47 min (+ 7.81 min)	5.3 %	5.7439	0.49 min (+ 7.81 min)
Proposed Black-box (ZOO-ADAM)	100 %	0.19973	3.43 min	96.8 %	0.39879	3.95 min
Proposed Black-box (ZOO-Newton)	100 %	0.23554	4.41 min	97.0 %	0.54226	4.40 min

5. Evaluation 2:

(1) 实验 2 目标：作者尝试将这种攻击运用在更大的模型上，并且探讨文章提出的缓解方法的作用；

(2) 黑盒模型：Inception-V3;

(3) 实验设定：

- 无目标攻击：

  生成150张对抗样本；保证每张对抗样本的大小都大于 $299 * 299$；不使用分层递进方法，只使用一个 $32 * 32 * 3$ 的对抗扰动域进行攻击；限制算法的迭代轮数为 1500 轮（$1500 * 128$ 次 Query）；

- 有目标攻击：

  只选择了一张在无目标攻击中无法成功攻击的样本；扩大对抗扰动域为 $64 * 64 * 3$ 和 $128 * 128 * 3$；最大迭代轮数上升为 20000 轮（$20000 * 128$ 次 Query）；（可以看到，黑盒下面的 Query 数量是十分惊人的）

(4) 实验结果：

- 无目标攻击：

Table 2: Untargeted ImageNet attacks comparison. Substitute model based attack cannot easily scale to ImageNet.

	Success Rate	Avg. L_2
White-box (C&W)	100 %	0.37310
Proposed black-box (ZOO-ADAM)	88.9 %	1.19916
Black-box (Substitute Model)	N.A.	N.A.

- 有目标攻击：

Table 3: Comparison of different attack techniques. “First Valid” indicates the iteration number where the first successful attack was found during the optimization process.

Black-box (ZOO-ADAM)	Success?	First Valid	Final L_2	Final Loss
All techniques	Yes	15,227	3.425	11.735
No Hierarchical Attack	No	-	-	62.439
No importance sampling	Yes	17,403	3.63486	13.216
No ADAM state reset	Yes	15,227	3.47935	12.111

Links

- 论文链接：Chen P Y, Zhang H, Sharma Y, et al. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models[C]//Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. 2017: 15-26.
- 论文代码：https://github.com/IBM/ZOO-Attack
- C&W Attack 代码：https://github.com/carlini/nn_robust_attacks
- CleverHans：https://github.com/cleverhans-lab/cleverhans

Adversarial Patch

Contribution

1. 成功在图像中打上一个通用的、鲁棒的、和输入无关的、物理中可成功的对抗样本；
2. 算法细节：使用梯度下降即可；

Links

- 论文链接：Brown T B, Mané D, Roy A, et al. Adversarial patch[J]. arXiv preprint arXiv:1712.09665, 2017.

Synthesizing Robust Adversarial Examples

Contribution

1. 提出了一种增加物理环境下对抗样本鲁棒性的一般化方法 EOT；
2. 不仅在 2D 下测试，而且在 3D 下测试；
3. **模拟物理变换**的想法十分具有借鉴意义，已被后续的对抗攻击算法广泛使用；

Notes

1. **白盒**的、**针对物理环境**下的、**有目标**的对抗攻击算法。攻击的算法不仅在 2D 下可行，同时在 **3D** 下也可以生成成功的对抗样本；
2. 已有的对抗攻击算法，训练的目标如下：

$$\begin{aligned} \arg\max_{x'} \quad & \log P(y_t|x') \\ \text{subject to} \quad & ||x'-x||_p < \epsilon \\ & x' \in [0,1]^d \end{aligned}$$

但是这样生成的对抗样本，在视角等物理环境发生改变时**无法保持对抗性**。故作者提出改进后的训练目标 **EOT (Expectation Over Transformation)**：

$$\begin{aligned} \arg\max_{x'} \quad & \mathbb{E}_{t\sim T}[\log P(y_t|t(x'))] \\ \text{subject to} \quad & \mathbb{E}_{t\sim T}[d(t(x'),t(x))] < \epsilon \\ & x \in [0,1]^d \end{aligned}$$

其含义是，**在保证对抗样本经过物理变换的 “感受” 修改量在一定范围内时，使得对抗样本（经过物理变换）能够尽可能地被分类为目标类别。**这类物理变换可以是 2D/3D 的变换，包括随机旋转、平移、噪声、视角变化、光照等。作者将公式转换为 Carlini &Wagner (2017c) 的形式，并使用**二级范数**和 **PGD** (Projected Gradient Descent) 优化器进行计算：

$$\arg\max_{x'} \mathbb{E}_{t\sim T}\Big[\log P(y_t|t(x')) \\ -\lambda||LAB(t(x')) - LAB(t(x))||_2\Big]$$

其中 *LAB* 代表指的是 LAB 色域 。

3. Distributions of Transformations:

(1) **2D Case**

Transformation	Minimum	Maximum
Scale	0.9	1.4
Rotation	$-22.5°$	$22.5°$
Lighten / Darken	-0.05	0.05
Gaussian Noise (stdev)	0.0	0.1
Translation	any in-bounds	

(2) **3D Case**

Transformation	Minimum	Maximum
Camera distance	2.5	3.0
X/Y translation	-0.05	0.05
Rotation	any	
Background	(0.1, 0.1, 0.1)	(1.0, 1.0, 1.0)

(3) **Physical Case**

Transformation	Minimum	Maximum
Camera distance	2.5	3.0
X/Y translation	-0.05	0.05
Rotation	any	
Background	(0.1, 0.1, 0.1)	(1.0, 1.0, 1.0)
Lighten / Darken (additive)	-0.15	0.15
Lighten / Darken (multiplicative)	0.5	2.0
Per-channel (additive)	-0.15	0.15
Per-channel (multiplicative)	0.7	1.3
Gaussian Noise (stdev)	0.0	0.1

4. Evaluation:

(1) 攻击基于数据集 ImageNet 的 **Inception V3** 模型 (`Top-1 Accuracy = 78.0%`)，随机选择目标分类；

(2) **Robust 2D adversarial examples**：在 2D 下考虑的物理变换有 **缩放 、旋转、亮度调节、高斯噪声和平移。**每个样本都在 **1000** 个随机的模拟物理变换上进行测试，结果如下：

Images	Classification Accuracy		Adversariality		ℓ_2
	mean	stdev	mean	stdev	mean
Original	70.0%	36.4%	0.01%	0.3%	0
Adversarial	0.9%	2.0%	96.4%	4.4%	5.6×10^{-5}

(3) **Robust 3D adversarial examples**：在 3D 下考虑**不同的相机距离、照明条件、对象的平移和旋转以及纯色背景色。**挑选了 10 个 3D 模型 —— 木桶、棒球、够、橘子、海龟、小丑鱼、沙发、泰迪熊、汽车和出租车。每个 3D 模型都挑选 20 个随机的目标分类标签；每个样本都在 100 个随机的模拟物理变换上进行测试，结果如下：

Images	Classification Accuracy		Adversariality		ℓ_2
	mean	stdev	mean	stdev	mean
Original	68.8%	31.2%	0.01%	0.1%	0
Adversarial	1.1%	3.1%	83.4%	21.7%	5.9×10^{-3}

(4) **Physical adversarial examples**：在 3D 的基础上，考虑**摄像机的噪声、照明的影响和颜色的失真**。作者考虑将 "海龟" 错误分类成 "手枪"、 "棒球" 错误分类成 "咖啡" 两种情况，将对抗样本经过 3D 打印后，拍 100 张照片进行测试，结果如下：

Object	Adversarial	Misclassified	Correct
Turtle	82%	16%	2%
Baseball	59%	31%	10%

(5) **Perturbation budget**：在物理环境下越鲁棒，需要模拟更多的物理变换，添加的噪声也会更多；

Links

- 论文链接：Athalye, Anish, et al. "Synthesizing robust adversarial examples." *International conference on machine learning*. PMLR, 2018.
- 开源代码：prabhant/synthesizing-robust-adversarial-examples (github.com)

*Semantic Adversarial Examples

Contribution

1. 通过修改 HSV 色域，生成对抗样本；

Notes

1. 图像语义对抗样本（Semantic Adversarial Examples）：将问题定义如下

$$\begin{aligned} &\text{find } x^* \\ &\text{s.t. } \Omega(x^*) = \Omega(x) \text{ and } F(x^*) \neq F(x). \end{aligned}$$

其中，Ω 表示人的视觉系统；

2. 方法：将图像从 RGB 色域转换到 HSV 色域，通过对全图修改 H（Hue，色相） 和 S（Saturation，饱和度）的值来生成对抗样本，算法如下

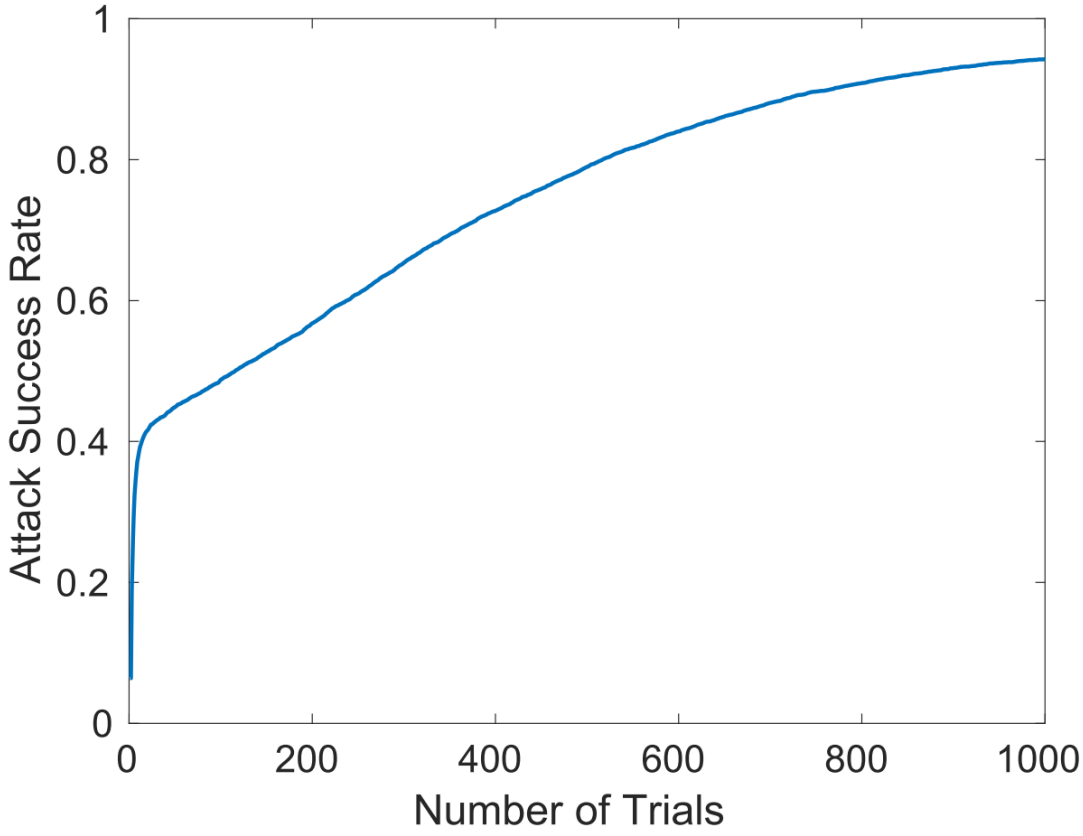
Algorithm 1 Generating Adversarial Color-shifted Images

Input: Classifier F, Image x, Maximum number of trials N
Output: Adversarial color-shifted image x^* or $\emptyset$
$x_H, x_S, x_V \leftarrow$ Hue, Saturation and Value components of image x, respectively.
$x^* \leftarrow x$
for $i = 0, ..., N-1$ **do**
    $\delta_H \leftarrow$ a number uniformly chosen in $[0, 1]$
    $\delta_S \leftarrow$ a number uniformly chosen in $[-\frac{i}{N}, \frac{i}{N}]$
    $x^*_H = (x_H + \delta_H) \bmod 1$
    $x^*_S = \text{clip}(x_S + \delta_S, 0, 1)$
    if $F(x^*) \neq F(x)$ **then**
        return x^*
    end if
end for
return $\emptyset$

3. 实验结果：（感觉更像是不同分布导致网络的预测结果差，另外，采用的是 CIFAR-10 数据集可能也是原因之一，使用更加高清的图片集可能会没有这么好的效果）

Table 1: Accuracy of different CNNs on test data and adversarial color-shifted images. The VGG-augmented is a VGG16 network trained with both original and color-shifted images.

Network	**Accuracy on test images**	**Accuracy on adversarial color-shifted images**
Pretrained VGG16 [15]	93.6%	5.7%
Madry et al. Model [8]	87.3%	8.4%
VGG-augmented	89.9%	69.1%



Links

- 论文链接：Hosseini H, Poovendran R. Semantic adversarial examples[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018: 1614-1619.
- 代码链接：https://github.com/HosseinHosseini/Semantic-Adversarial-Examples

GENERATING NATURAL ADVERSARIAL EXAMPLES

Contribution

Notes

Links

- 论文链接：Zhao Z, Dua D, Singh S. Generating natural adversarial examples[J]. ICLR 2018

NES: Black-box Adversarial Attacks with Limited Queries and Information

Contribution

1. 利用 NES 算法大大**减少黑盒攻击的访问次数**;

Notes

1. 黑盒设定:
    - Query-limited Setting: 限制访问次数 ;
    - Partial-information Setting: 只知道 Top-K 的结果 (包括概率);
    - Label-only Setting: 只知道 Top-K 的标签 (不包括概率); (这一项我觉得没必要看)
2. ☆ NES (Natural Evolutionary Strategies) 进行梯度估计: 最小化期望的损失大小, 算法伪代码如下 (如何挑选这个参数?)

Algorithm 1 NES Gradient Estimate

Input: Classifier $P(y|x)$ for class y, image x
Output: Estimate of $\nabla P(y|x)$
Parameters: Search variance σ, number of samples n, image dimensionality N
$g \leftarrow \mathbf{0}_n$
for $i = 1$ **to** n **do**
  $u_i \leftarrow \mathcal{N}(\mathbf{0}_N, \boldsymbol{I}_{N \cdot N})$
  $g \leftarrow g + P(y|x + \sigma \cdot u_i) \cdot u_i$
  $g \leftarrow g - P(y|x - \sigma \cdot u_i) \cdot u_i$
end for
return $\frac{1}{2n\sigma} g$

看不懂这个式子的话, 在草稿纸上把这两个式子列成求梯度的形式

3. PGD (Projected Gradient Descent) 进行梯度更新，实现**有目标的对抗攻击**:

$$x^{(t)} = \Pi_{[x_0-\epsilon, x_0+\epsilon]}(x^{(t-1)} - \eta \cdot \text{sign}(g_t))$$

(这个公式有歧义：这里明显是一个梯度下降算法，期望损失函数值变得更小，所以这里 g_t 表示的是损失函数的梯度，而不是 2 中梯度——表示的是目标分类概率的梯度，这一点可以在程序的代码中验证。)

4. 仅知道 Top-K 的概率:

Algorithm 2 Partial Information Attack

Input: Initial image x, Target class y_{adv}, Classifier $P(y|x) : \mathbb{R}^n \times \mathcal{Y} \to [0,1]^k$ (access to probabilities for y **in top** k), image x

Output: Adversarial image x_{adv} with $||x_{adv} - x||_\infty \leq \epsilon$

Parameters: Perturbation bound ϵ_{adv}, starting perturbation ϵ_0, NES Parameters (σ, N, n), epsilon decay δ_ϵ, maximum learning rate η_{max}, minimum learning rate η_{min}

$\epsilon \leftarrow \epsilon_0$
$x_{adv} \leftarrow$ image of target class y_{adv}
$x_{adv} \leftarrow \text{CLIP}(x_{adv}, x - \epsilon, x + \epsilon)$
while $\epsilon > \epsilon_{adv}$ or $\max_y P(y|x) \neq y_{adv}$ **do**
  $g \leftarrow \text{NESESTGRAD}(P(y_{adv}|x_{adv}))$
  $\eta \leftarrow \eta_{max}$
  $\hat{x}_{adv} \leftarrow x_{adv} - \eta g$
  while not $y_{adv} \in \text{TOP-K}(P(\cdot|\hat{x}_{adv}))$ **do**
    if $\eta < \eta_{min}$ **then**
      $\epsilon \leftarrow \epsilon + \delta_\epsilon$
      $\delta_\epsilon \leftarrow \delta_\epsilon / 2$
      $\hat{x}_{adv} \leftarrow x_{adv}$
      break
    end if
    $\eta \leftarrow \frac{\eta}{2}$
    $\hat{x}_{adv} \leftarrow \text{CLIP}(x_{adv} - \eta g, x - \epsilon, x + \epsilon)$
  end while
  $x_{adv} \leftarrow \hat{x}_{adv}$
  $\epsilon \leftarrow \epsilon - \delta_\epsilon$
end while
return x_{adv}

当学习率低于最小值时仍未生成对抗样本时，则增大扰动的变化区间.

当学习率仍大于最小值时，不断减小学习率进行探测

- 使用目标分类的样本来初始化扰动，从而减少 query 的数量；
- 在保证目标分类在 Top-K 中的前提下，不断缩小对抗扰动，直至生成对抗样本且满足修改量的限制；

5. Evaluation：

(1) 参数的选择：

General	
σ for NES	0.001
n, size of each NES population	50
ϵ, l_∞ distance to the original image	0.05
η, learning rate	0.01
Partial-Information Attack	
ϵ_0, initial distance from source image	0.5
δ_ϵ, rate at which to decay ϵ	0.001
Label-Only Attack	
m, number of samples for proxy score	50
μ, ℓ_∞ radius of sampling ball	0.001

(2) On ImageNet：这里大概的 query 数量级为上万级别的

Threat model	Success rate	Median queries
QL	99.2%	11,550
PI	93.6%	49,624
LO	90%	2.7×10^6

Table 1. Quantitative analysis of targeted $\epsilon = 0.05$ adversarial attacks in three different threat models: query-limited (QL), partial-information (PI), and label-only (LO). We perform attacks over 1000 randomly chosen test images (100 for label-only) with randomly chosen target classes. For each attack, we use the same hyperparameters across all images. Here, we report the overall success rate (percentage of times the adversarial example was classified as the target class) and the median number of queries required.

Links

- 论文链接: Ilyas, Andrew, et al. "Black-box adversarial attacks with limited queries and information." *PRML* (2018).
- 论文代码: https://github.com/labsix/limited-blackbox-attacks
- NES原理: Lil'Log - Evolution Strategies

Generating Adversarial Examples with Adversarial Networks

Contribution

1. 使用 GAN 网络的方法可以**快速**生成一些逼真的对抗样本；
2. 实验中生成成功对抗样本的最大概率为 98%（在白盒情况下 MNSIT 模型）；
3. 生成的样本不一定是一个对抗样本，如果要用来做对抗训练的话需要经过目标模型的验证；
4. 使用 GAN 来生成对抗样本还是比较有意思的一个点，但是实际的价值可能并不大；

Notes

1. 作者提出了一种使用 GAN 生成对抗样本的算法，可以在白盒、黑盒情况下进行有目标攻击；
2. ☆ AdvGAN 架构：

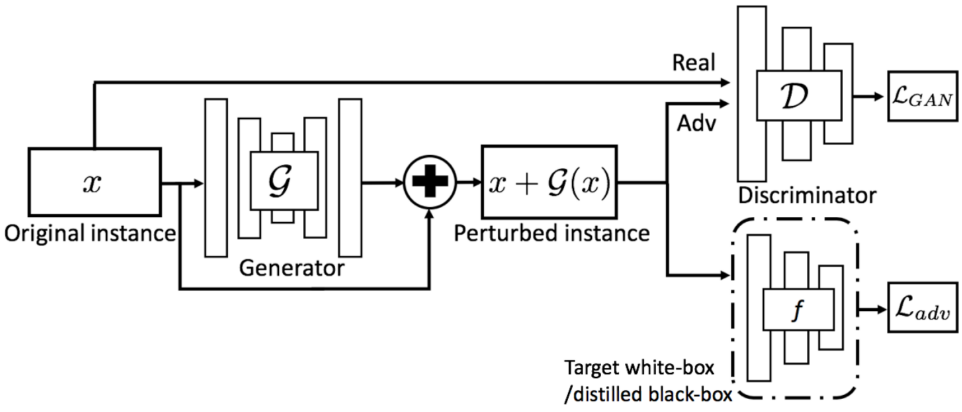


Figure 1: Overview of AdvGAN

从架构中可以看出，整个网络通过判别模型保证生成的样本更像"真实"的样本，通过分类模型来保证生成的样本能够被错误分类为目标样本，再结合（图中没有画出的）扰动应该尽可能小，这三个部分一起构成了 `Generator` 的损失函数 $\mathcal{L} = \alpha\mathcal{L}_{GAN} + \mathcal{L}_{adv}^{f} + \beta\mathcal{L}_{hinge}$：

- 保证样本的真实性：
- 保证样本的目标分类：

$$\mathcal{L}_{\text{adv}}^{f} = \mathbb{E}_x \ell_f(x + \mathcal{G}(x), t)$$

- 保证样本的扰动大小：

$$\mathcal{L}_{\text{hinge}} = \mathbb{E}_x \max(0, \|\mathcal{G}(x)\|_2 - c)$$

3. ☆ 蒸馏模型（本质上是 **替代模型**）：如上所示，如果要训练一个 AdvGAN 网络，就需要有一个目标分类模型，但是在黑盒的情况下，我们无法得到目标模型的损失。面对这个问题，作者的想法是使用 **蒸馏(distill)模型** 的方法，即在一个数据集中，训练一个本地模型使得：

$$\arg\min_{f} \mathbb{E}_x \ \mathcal{H}(f(x), b(x))$$

其中 $f(x)$ 是本地模型的输出概率，$b(x)$ 是目标模型的输出概率，$\mathcal{H}(\cdot)$ 常用交叉熵损失函数；由于涉及到蒸馏模型和原始模型的相似程度，所以作者提出了一个动态训练蒸馏模型的方法（实际上和本身训练一个 GAN 网络的方法是一样的）：

- 固定分类模型 f_{i-1}，训练判别器 $\mathcal{D}_i$ 和生成器 $\mathcal{G}_i$：
  $\arg\min_{\mathcal{G}} \max_{\mathcal{D}} \alpha\mathcal{L}_{GAN} + \mathcal{L}_{adv}^{f_{i-1}} + \beta\mathcal{L}_{hinge}$；
- 固定生成器 $\mathcal{G}_i$，继续训练分类模型 f_i：
  $\arg\min_f \mathbb{E}_x \mathcal{H}(f(x), b(x)) + \mathbb{E}_x \mathcal{H}(f(x + \mathcal{G}_i(x)), b(x + \mathcal{G}_i(x)))$；

> 用蒸馏模型的方法来训练一个 替代模型 的想法，然后用于对抗样本生成是十分**耗费 Query 数量**的。我作为一个攻击者，看到这么大的代价，可能会选择使用本地语料训练模型然后生成对抗样本；

4. 实验：

(1) 数据集：MNIST 和 CIFAR-10；

(2) 参数设置：

- 使用 C&W 损失函数；
- 损失范围使用无穷范数，在 MNIST 上的大小为 0.3，在 CIFAR-10 上的大小为 8；

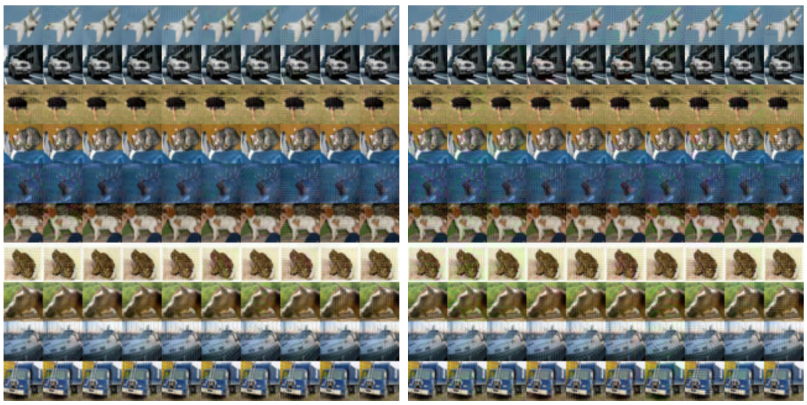
(3) 实验结果：

Model	MNIST(%) A	B	C	CIFAR-10(%) ResNet	Wide ResNet
Accuracy (p)	99.0	99.2	99.1	92.4	95.0
Attack Success Rate (w)	97.9	97.1	98.3	94.7	99.3
Attack Success Rate (b-D)	93.4	90.1	94.0	78.5	81.8
Attack Success Rate (b-S)	30.7	66.6	87.3	10.3	13.3

Table 2: Accuracy of different models on pristine data, and the attack success rate of adversarial examples generated against different models by AdvGAN on MNIST and CIFAR-10. p: pristine test data; w: semi-whitebox attack; b-D: black-box attack with dynamic distillation strategy; b-S: black-box attack with static distillation strategy.

- 白盒攻击下的攻击成功率达到了 95%；

- 同一张原始图片针对 **不同的目标类别生成** 的对抗样本在视觉上的区别不大；（👎 讲到 "**不同的目标类别生成**" 这一点，我们回过头来审视一下 分类目标损失函数，可以发现我们训练的模型只能生成一个目标分类，所以这种方法的一个缺点在于对于不同的目标分类需要训练不同的 GAN 网络）



(a) Semi-whitebox attack (b) Black-box attack

Figure 3: Adversarial examples generated by AdvGAN on CIFAR-10 for (a) semi-whitebox attack and (b) black-box attack. Image from each class is perturbed to other different classes. On the diagonal, the original images are shown.

- 动态训练蒸馏模型的效果大大优于静态训练蒸馏模型；

5. 针对对抗训练的防御方法：

(1) 对抗训练的算法：

- FGSM Adversarial Training (简称 **Adv.**)；
- Ensemble Adversarial Training (简称 **Ens.**)；
- Iterative Training (简称 **Iter.Adv.**)；

(2) 👎 参照的对抗样本生成算法：FGSM 和 C&W Attack (文章中称为 **Opt.**)；

> 我觉得这样进行对比，在一定程度上是不合理的。首先我们来看这些对抗训练算法，本身就是用来针对 **单次迭代和多次迭代生成算法** 进行加强训练的，故通过这样得到的模型本身对 FGSM 和 C&W Attack 是更加容易攻击的，即 AdvGAN 在这个测试上面有一些天然优势，在最后的测试结果上多几个点也（大体上）是理所当然的。转过来想，我们可能会更加希望看到这样的结果：**即使防御者知道我们用的是怎样的对抗攻击算法，但是他都无法用对抗训练的方法把我们防御住，那么这就是一个好的对抗攻击算法。**

(3) 白盒上的结果：AdvGAN 的效果来得稍微好一些，但是不明显；（另外，我认为这个结果就不是很合理，或者说作者对实验的设置交代的不够清楚。用了对抗训练以后，我们的模型就可以防御 C&W Attack 了？我觉得不是这样的吧，你用梯度下降的方法，多迭代几轮，增大一些扰动肯定是可以找到对抗样本点的。对抗训练无非是限制了模型在训练数据点处的概率分布，从而能够在整体上增加对抗算法生成样本的难度，但是并不能从根本上改变模型不同类别间存在边界这个固有的特点，通过梯度下降的方法一定是可以越过这个边界的。）

Data	Model	Defense	FGSM	Opt.	**AdvGAN**
MNIST	A	Adv.	4.3%	4.6%	**8.0%**
		Ens.	1.6%	4.2%	**6.3%**
		Iter.Adv.	4.4%	2.96%	**5.6%**
	B	Adv.	6.0%	4.5%	**7.2%**
		Ens.	2.7%	3.18%	**5.8%**
		Iter.Adv.	**9.0%**	3.0%	6.6%
	C	Adv.	2.7%	2.95%	**18.7%**
		Ens.	1.6%	2.2%	**13.5%**
		Iter.Adv.	1.6%	1.9%	**12.6%**
CIFAR10	ResNet	Adv.	13.10%	11.9%	**16.03%**
		Ens.	10.00%	10.3%	**14.32%**
		Iter.Adv	22.8%	21.4%	**29.47%**
	Wide ResNet	Adv.	5.04%	7.61%	**14.26%**
		Ens.	4.65%	8.43%	**13.94 %**
		Iter.Adv.	14.9%	13.90%	**20.75%**

Table 3: Attack success rate of adversarial examples generated by AdvGAN in semi-whitebox setting, and other white-box attacks under defenses on MNIST and CIFAR-10.

(4) 黑盒上的结果：

	MNIST			CIFAR-10		
Defense	FGSM	Opt.	**AdvGAN**	FGSM	Opt.	**AdvGAN**
Adv.	3.1%	3.5%	**11.5%**	13.58%	10.8%	**15.96%**
Ens.	2.5%	3.4%	**10.3%**	10.49%	9.6%	**12.47%**
Iter.Adv.	2.4%	2.5%	**12.2%**	22.96%	21.70%	**24.28%**

Table 4: Attack success rate of adversarial examples generated by different black-box adversarial strategies under defenses on MNIST and CIFAR-10

6. 用户调查：调查结果说明生成的样本还是很难发觉的，并且和真实的图片相似；

Links

- Xiao C, Li B, Zhu J Y, et al. Generating adversarial examples with adversarial networks[J]. arXiv preprint arXiv:1801.02610, 2018.

LaVAN: Localized and Visible Adversarial Noise

Contribution

1. 相对于 Adversarial Patch 那篇文章，我觉得没有什么创新，只不过可能对实验部分做了更多的细化；

Notes

1. Localized noise for a single image and location：先只是生成简单的对抗样本，不能实现 transfer 能力；（另外，我觉得这边用 `Network Domain` 这样来生成对抗扰动明显是不合理的，因为这样的对抗样本在物理世界是根本不存在的；进一步来考虑，其实我们应该确定一个图像的精度，即像素值是1~255的整数，那么我们的最小精度应该设置在 1/255；）

Algorithm 1 Localized Noising Process

Input: image x, model p_M, target class y_{target}, target probability s, mask m, noise domain d.

$y_{\text{source}} = \arg\max_y p_M(y|x)$

$\delta = \vec{0}$

$x' = (1-m) \odot x + m \odot \text{noise}$

repeat

$\vec{y} = p_M(x')$

$L_{\text{target}} = \ell(\vec{y}, y_{\text{target}}) = M(y = y_{\text{target}}|x')$

$L_{\text{argmax}} = \ell(\vec{y}, y_{\text{argmax}}) = M(y = y_{\text{argmax}}|x')$

$\nabla_{\text{target}} = \frac{\partial L_{\text{target}}}{\partial x}$

$\nabla_{\text{argmax}} = \frac{\partial L_{\text{argmax}}}{\partial x}$

$\delta = \delta - \epsilon \cdot (\nabla_{\text{target}} - \nabla_{\text{argmax}})$

if $d =$ *image domain* **then**

$\delta = clip(\delta, 0, 1)$

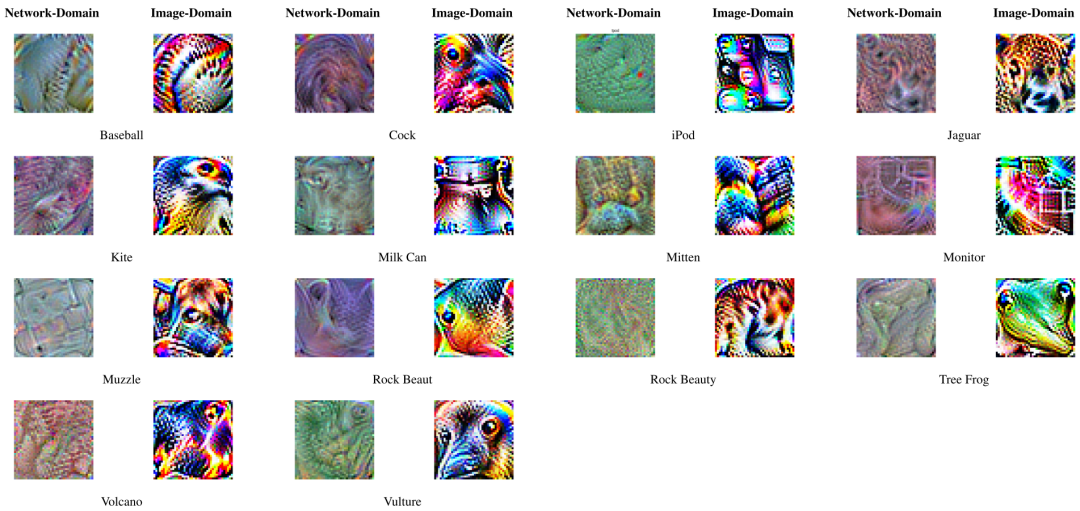
end if

$x' = (1-m) \odot x + m \odot \delta$

until $p_M(y = y_{target}|x') \geq s$

2. Transferable localized noise：生成强化后的对抗样本，对于不同的图片和位置不敏感；对上述算法做简单的修改，在每一轮的对抗样本生成过程中都选择 100 张图片，并且随机选择放置的位置；

   看一下它生成的 Patch：可以发现这样的对抗攻击明显学习到了目标类别样本的特征；

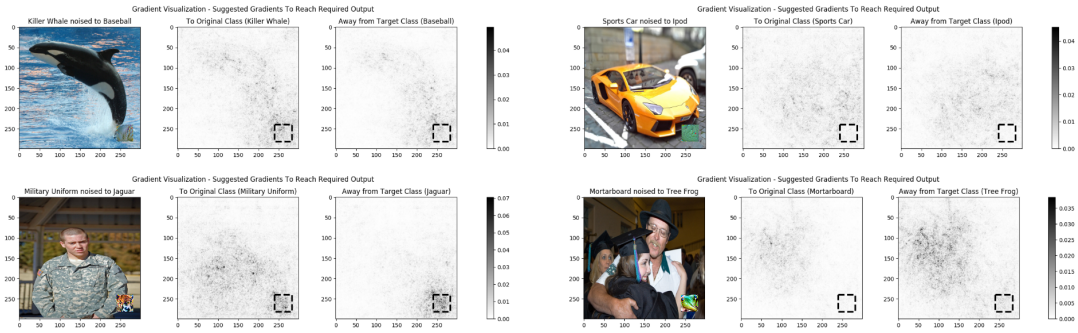


3. 实验结果

   - 评估指标：成功率、对位置的敏感性、对原始图片的敏感性、图像分类的敏感性；
   - 图像分类的敏感性：即成功率是否与源图像/目标图像的类型相关，结果如图所示；

Source Class \ Target Class	baseball	cock	iPod	jaguar	kite	milk-can	mitten	monitor	muzzle	rock beauty	tiger-cat	tree-frog	volcano	vulture
academic gown	0.18	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
banana	0.98	0.70	0.91	0.94	0.91	0.29	0.96	0.32	0.99	0.72	0.71	0.96	0.06	0.91
barbershop	0.99	0.92	0.34	0.93	0.89	0.63	0.91	0.50	0.98	0.69	0.78	0.95	0.22	0.89
bee	0.99	0.79	0.81	0.88	0.85	0.20	0.71	0.12	0.93	0.70	0.68	0.96	0.12	0.91
bell pepper	0.94	0.48	0.19	0.73	0.66	0.50	0.53	0.04	0.12	0.53	0.61	0.89	0.05	0.90
garden spider	0.97	0.66	0.10	0.95	0.89	0.46	0.19	0.22	0.99	0.84	0.83	0.96	0.05	0.71
book jacket	0.99	0.77	0.15	0.81	0.90	0.83	0.45	0.27	0.99	0.90	0.84	0.97	0.08	0.96
brown bear	0.80	0.21	0.13	0.30	0.32	0.31	0.21	0.02	0.59	0.19	0.32	0.55	0.01	0.28
bullet train	0.98	0.47	0.01	0.79	0.61	0.54	0.65	0.20	0.69	0.40	0.74	0.88	0.05	0.49
cardigan	0.94	0.51	0.25	0.77	0.66	0.18	0.55	0.17	0.93	0.29	0.56	0.71	0.15	0.45
corn	0.99	0.76	0.63	0.96	0.97	0.89	0.90	0.05	1.00	0.81	0.72	0.95	0.33	0.98
crash helmet	0.98	0.50	0.10	0.79	0.80	0.22	0.41	0.05	0.58	0.46	0.58	0.92	0.02	0.41
daisy	0.96	0.51	0.39	0.77	0.73	0.71	0.66	0.33	0.87	0.62	0.65	0.90	0.13	0.72
dishrag	0.97	0.67	0.06	0.90	0.95	0.52	0.63	0.02	0.95	0.67	0.73	0.92	0.10	0.92
dogsled	0.93	0.44	0.14	0.48	0.62	0.60	0.25	0.02	0.49	0.27	0.44	0.70	0.07	0.32
electric ray	0.98	0.77	0.15	0.92	0.94	0.98	0.89	0.14	0.99	0.84	0.73	0.95	0.13	0.98
football helmet	0.96	0.61	0.01	0.65	0.77	0.00	0.05	0.01	0.16	0.41	0.50	0.84	0.00	0.66
gong	0.95	0.36	0.03	0.84	0.71	0.11	0.42	0.02	0.50	0.28	0.72	0.75	0.01	0.37
green mamba	0.92	0.23	0.11	0.56	0.55	0.35	0.25	0.10	0.81	0.35	0.53	0.43	0.04	0.36
hatchet	0.50	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.07	0.00	0.00
hip	0.98	0.65	0.41	0.95	0.82	0.95	0.96	0.29	0.99	0.80	0.79	0.83	0.20	0.59
hummingbird	1.00	0.61	0.66	0.94	0.91	0.91	0.69	0.57	0.96	0.92	0.86	0.97	0.19	0.94
Indian cobra	0.96	0.29	0.30	0.76	0.64	0.36	0.14	0.16	0.87	0.60	0.59	0.77	0.05	0.64
iPod	0.99	0.79	1.00	0.96	0.88	0.10	0.45	0.10	0.94	0.80	0.89	0.93	0.13	0.94
jellyfish	0.99	0.52	0.75	0.91	0.87	0.93	0.56	0.31	0.96	0.84	0.89	0.94	0.05	0.89
lacewing	0.93	0.33	0.05	0.72	0.73	0.20	0.24	0.02	0.90	0.17	0.36	0.46	0.05	0.15
mailbag	0.84	0.23	0.02	0.44	0.60	0.13	0.10	0.00	0.69	0.02	0.68	0.73	0.02	0.05
mantis	0.94	0.47	0.27	0.71	0.66	0.27	0.35	0.19	0.59	0.39	0.67	0.73	0.10	0.59
military uniform	0.97	0.82	0.15	0.90	0.87	0.69	0.90	0.33	0.97	0.68	0.67	0.90	0.21	0.82
minivan	0.98	0.87	0.13	0.92	0.91	0.76	0.95	0.09	1.00	0.78	0.80	0.97	0.09	0.92
mortarboard	0.97	0.48	0.24	0.60	0.66	0.10	0.55	0.07	0.66	0.36	0.51	0.67	0.06	0.56
poncho	0.99	0.89	0.51	0.92	0.93	0.37	0.93	0.18	1.00	0.78	0.88	0.95	0.05	0.92
rapeseed	0.99	0.71	0.51	0.93	0.90	0.91	0.96	0.27	1.00	0.86	0.75	0.94	0.38	0.83
starfish	0.99	0.82	0.69	0.95	0.77	0.50	0.86	0.17	0.99	0.82	0.74	0.98	0.07	0.95
turnstile	0.99	0.78	0.01	0.93	0.74	0.50	0.56	0.02	0.96	0.23	0.72	0.94	0.05	0.73
velvet	0.98	0.66	0.19	0.91	0.90	0.61	0.86	0.04	1.00	0.63	0.49	0.92	0.11	0.82
walking stick	0.97	0.40	0.54	0.79	0.78	0.54	0.54	0.27	0.67	0.59	0.76	0.84	0.10	0.69
washer	0.98	0.64	0.17	0.90	0.78	0.51	0.78	0.25	0.93	0.61	0.80	0.93	0.07	0.66
water snake	0.99	0.75	0.59	0.94	0.97	0.94	0.86	0.22	1.00	0.89	0.69	0.95	0.28	0.96
wing	0.99	0.81	0.50	0.93	0.99	0.96	0.98	0.27	0.96	0.94	0.74	0.93	0.49	0.99

4. 作者想分析一下，网络是不是更加关注 patch 部分了，然后它做了两个实验——“用梯度下降让源分类概率上升”和“用梯度下降让目标分类概率下降”，先看一下他分析的结果：



可以看到，左边的 case 在 patch 框里面的扰动还是很多的，而右边这个 case 在 patch 框里面的扰动并不多；

作者给出的结论是，上面那个论点可能是不成立的（这个论点是在 Adversarial Patch 中提出的），其实我感觉这个问题已经涉及到了“模型可解释性”的问题，至于这里给出用 Grad 的方法来解释合不合理，也是不太好争论的，所以这个问题还是需要进一步讨论的；

5. 最后，作者分析扰动的大小，他直接用“绝对值的大小”和“绝对值求和”来度量扰动的大小，先说说我的想法，我觉得这样的评估很不合理，应该用人的感官来评估才是正确的方法，然后放一下作者的结果；

Domain	Fix method	MAX	SUM
Network	Towards Source	0.4%	0.15%
Network	Away from Target	0.5%	0.13%
Image	Towards Source	0.6%	5.4%
Image	Away from Target	0.7&	5.2%

Links

- 论文链接：Karmon D, Zoran D, Goldberg Y. Lavan: Localized and visible adversarial noise[C]//International Conference on Machine Learning. PMLR, 2018: 2507-2515.
- 论文代码：LaVAN python-tf-

* Prior Convictions: Black-Box Adversarial Attacks with Bandits and Priors

Contribution

1. 在 NES 的基础上，通过**减小查询空间（图像相邻点的梯度相似）**来减少 Query 的次数；
2. 这篇文章本身再减少 Query 上面没什么亮点，但是它对于 **“梯度估计正确率与对抗样本成功率”** 和 **“先验梯度知识”** 的分析是比较有意思的地方；

Notes

2. ☆ 梯度估计正确率与对抗样本成功率的关系：
    - 问题：在黑盒攻击的情况下，正确估计多少比率的梯度才能保障白盒算法（PGD）成功攻击目标模型？
    - 实验结果：使用单步 PGD 生成对抗样本的情况下，只要有 **20%** 的梯度被正确赋值，就有 **60%** 的成功率；

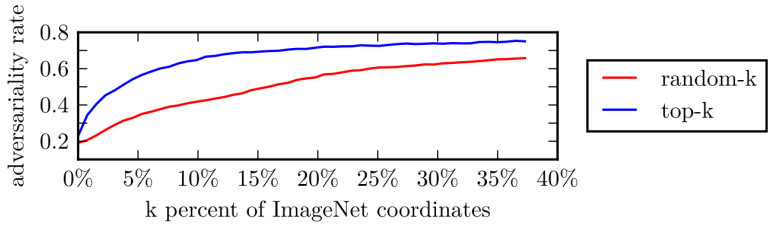


Figure 1: The fraction of correctly estimated coordinates of $\text{sgn}(\nabla_x L(x, y))$ required to successfully execute the single-step PGD (also known as FGSM) attack, with $\epsilon = 0.05$. In the experiment, for each k, the top k percent – chosen either by magnitude (`top-k`) or randomly (`random-k`) – of the signs of the coordinates are set correctly, and the rest are set to $+1$ or -1 at random. The adversariality rate is the portion of 1,000 random ImageNet images misclassified after one FGSM step. Observe that, for example, estimating only 20% of the coordinates correctly leads to misclassification in the case of more than 60% of images.

3. ☆ 先验（prior）梯度知识：

    (1) Time-dependent Priors：如下图 2 中所示，**连续两步之间的梯度之间存在高度的余弦相关性**（数据中，这个余弦相关性高达 0.9）；

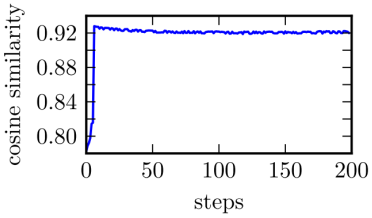


Figure 2: Cosine similarity between the gradients at the current and previous steps along the optimization trajectory of NES PGD attacks, averaged over 1000 random ImageNet images.

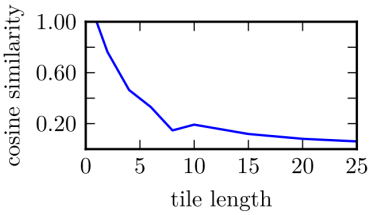


Figure 3: Cosine similarity of “tiled” image gradient with original image gradient versus the length of the square tiles, averaged over 5,000 randomly selected ImageNet images.

    (2) Data-dependent Priors：如上图 3 中所示，在图像中，**相近的像素点之间存在相似的梯度**；

4. 算法：

    (1) 框架：梯度估计 + 梯度更新；

Algorithm 1 Gradient Estimation with Bandit Optimization

1: **procedure** BANDIT-OPT-LOSS-GRAD-EST(x, y_{init})
2: $v_0 \leftarrow \mathcal{A}(\phi)$
3: **for** each round $t = 1, \ldots, T$ **do**
4: // Our loss in round t is $\ell_t(g_t) = -\langle \nabla_x L(x, y_{init}), g_t \rangle$
5: $g_t \leftarrow v_{t-1}$
6: $\Delta_t \leftarrow$ GRAD-EST(x, y_{init}, v_{t-1}) // Estimated Gradient of ℓ_t
7: $v_t \leftarrow \mathcal{A}(v_{t-1}, \Delta_t)$
8: $g \leftarrow v_T$
9: **return** $\Pi_{\partial\mathcal{K}}[g]$

(2) 梯度估计：（这部分其实就和 NES 算法一样，不一样的地方在于这个算法每轮只产生一对样本）

Algorithm 2 Single-query spherical estimate of $\nabla_v \langle \nabla L(x, y), v \rangle$

1: **procedure** GRAD-EST(x, y, v)
2: $u \leftarrow \mathcal{N}(0, \frac{1}{d} I)$ // Query vector
3: $\{q_1, q_2\} \leftarrow \{v + \delta \boldsymbol{u}, v - \delta \boldsymbol{u}\}$ // Antithetic samples
4: $\ell_t(q_1) = -\langle \nabla L(x, y), q_1 \rangle \approx \frac{L(x,y) - L(x + \epsilon \cdot q_1, y)}{\epsilon}$ // Gradient estimation loss at q_1
5: $\ell_t(q_2) = -\langle \nabla L(x, y), q_2 \rangle \approx \frac{L(x,y) - L(x + \epsilon \cdot q_2, y)}{\epsilon}$ // Gradient estimation loss at q_2
6: $\boldsymbol{\Delta} \leftarrow \frac{\ell_t(q_1) - \ell_t(q_2)}{\delta} \boldsymbol{u} = \frac{L(x + \epsilon q_2, y) - L(x + \epsilon q_1, y)}{\delta \epsilon} \boldsymbol{u}$
7: // Note that due to cancellations we can actually evaluate $\boldsymbol{\Delta}$ with only two queries to L
8: **return** $\boldsymbol{\Delta}$

其中第 6 行写的前后矛盾，以后半个式子为准；

(3) 梯度更新：

- 梯度上升（Gradient Ascent）法：（离谱）

$$v_t = \mathcal{A}(v_{t-1}, \Delta_t) := v_{t-1} + \eta \cdot \boldsymbol{\Delta}_t$$

- 指数梯度（Exponentiated Gradients）更新法：

$$p_{t-1} = \frac{1}{2}(v_{t-1} + 1)$$

$$p_t = \mathcal{A}(g_{t-1}, \Delta_t) := \frac{1}{Z} p_{t-1} \exp(\eta \cdot \Delta_t) \qquad \text{s.t. } Z = p_{t-1} \exp(\eta \cdot \Delta_t) + (1 - p_{t-1}) \exp(-\eta \cdot \Delta_t)$$

$$v_t = 2p_t - 1$$

(4) 图像识别，l_2 范数约束下的实例：

Algorithm 3 Adversarial Example Generation with Bandit Optimization for ℓ_2 norm perturbations

1: **procedure** ADVERSARIAL-BANDIT-L2(x_{init}, y_{init})
2: // $C(\cdot)$ returns top class
3: $v_0 \leftarrow \mathbf{0}_{1 \times d}$ // If data prior, $d <$ dim(x); v_t (Δ_t) up (down)-sampled before (after) line 8
4: $x_0 \leftarrow x_{init}$ // Adversarial image to be constructed
5: **while** $C(x) = y_{init}$ **do**
6: $g_t \leftarrow v_{t-1}$
7: $x_t \leftarrow x_{t-1} + h \cdot \frac{g_t}{||g_t||_2}$ // Boundary projection $\frac{g}{||g_t||}$ standard PGD: c.f. [Rig15]
8: $\Delta_t \leftarrow$ GRAD-EST($x_{t-1}, y_{init}, v_{t-1}$) // Estimated Gradient of ℓ_t
9: $v_t \leftarrow v_{t-1} + \eta \cdot \Delta_t$
10: $t \leftarrow t + 1$
return x_{t-1}

5. 使用的参数：

Table 2: Hyperparameters for the NES approach.

Hyperparameter	Value	
	ImageNet ℓ_∞	ImageNet ℓ_2
Samples per step	100	10
Learning Rate	0.01	0.3

Table 3: Hyperparameters for the bandits approach (variables names as used in pseudocode).

Hyperparameter	Value	
	ImageNet ℓ_∞	ImageNet ℓ_2
η (OCO learning rate)	100	0.1
h (Image ℓ_p learning rate)	0.01	0.5
δ (Bandit exploration)	1.0	0.01
η (Finite difference probe)	0.1	0.01
Tile size (Data-dependent prior only)	$(6px)^2$	$(6px)^2$

为什么 l_∞ 情况下的学习率会这么大?

6. 实验结果:

Table 1: Summary of effectiveness of ℓ_2 and ℓ_∞ ImageNet attacks on Inception v3 using NES, bandits with time prior (Bandits$_T$), and bandits with time and data-dependent priors (Bandits$_{TD}$). Note that in the first column, the average number of queries is calculated only over successful attacks, and we enforce a query limit of 10,000 queries. For purposes of direct comparison, the last column calculates the average number of queries used for only the images that NES (previous SOTA) was successful on. Our most powerful attack uses 2-4 times fewer queries, and fails 2-5 times less often.

Attack	Avg. Queries		Failure Rate		Queries on NES Success	
	ℓ_∞	ℓ_2	ℓ_∞	ℓ_2	ℓ_∞	ℓ_2
NES	1735	2938	22.2%	34.4%	1735	2938
Bandits$_T$	1781	2690	11.6%	30.4%	1214	2421
Bandits$_{TD}$	**1117**	**1858**	**4.6%**	**15.5%**	**703**	**999**

Links

- 论文链接: Ilyas A, Engstrom L, Madry A. Prior convictions: Black-box adversarial attacks with bandits and priors[J]. arXiv preprint arXiv:1807.07978, 2018.
- 论文代码:
- 官方审稿意见: https://openreview.net/forum?id=BkMiWhR5K7

> 吐槽一下: 这篇文章就**离谱**!
>
> 创新性上面, 我觉得就是在NES上面添加了一个局部降采样, 没啥突出的地方 (分析的地方确实是有意思的)。
>
> 公式上面, 你看这个公式里面, g_t 有个啥用啊? **离谱**! 再看这个公式的错误, 那个第 **6** 行? **离谱**! 然后你看这个有目标攻击配合上loss梯度上升法 (它的程序里面是对的)? **离谱**! 关键是这个错误你再和 NES 文章的那个错误 (有目标攻击配合上目标概率梯度下降法) 对比一下, 就完全是同样的错误啊? **离谱** (我看了半天没看懂, 回 NES 发现是同样的错误, 就在想不会是同一个作者吧, 结果一看真的是同一个作者)! 审稿意见一致好评通过? **离谱**!
>
> 以上就是一时生气, 因为这几个问题, 我把两篇文章的源码都去看了一遍, 感觉很浪费读者的时间。

Hybrid Batch Attacks: Finding Black-box Adversarial Examples with Limited Queries

Contribution

Notes

1. 作者结合 迁移攻击（Transfer Attack） 和 优化攻击（Optimization Attack），并且利用种子优先级策略对模型进行黑盒攻击，目标是在保证攻击成功率的前提下，减少 Query 的数量；
2. 现有的梯度优化攻击算法：

Attack	Gradient Estimation	Queries per Iteration	White-box Attack
ZOO [10]	$\hat{\mathbf{g}} = \{\hat{g}_i, \hat{g}_2, ..., \hat{g}_D\}$, $\hat{g}_i \approx \frac{f(\mathbf{x}+\delta e_i) - f(\mathbf{x}-\delta e_i)}{\delta}$	$2D$	CW [8]
Bhagoji et. al [4]	ZOO + random feature group or PCA	$\leq 2D$	FGSM [17], PGD [32]
AutoZOOM [43]	$\mathbf{u}_i \sim U$, $\hat{\mathbf{g}} = \frac{1}{N}\sum_i^N \frac{f(\mathbf{x}+\delta\mathbf{u}_i) - f(\mathbf{x})}{\delta}\mathbf{u}_i$	$N+1$	CW [8]
NES [21]	$\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, I)$, $\hat{\mathbf{g}} = \frac{1}{N}\sum_i^N \frac{f(\mathbf{x}+\delta\mathbf{u}_i)}{\delta}\mathbf{u}_i$	N	PGD
$\text{Bandits}_{\text{TD}}$ [22]	NES + time/data dependent info	N	PGD
SignHunter [1]	Gradient sign w/ divide-and-conquer method	$2^{\lceil \log(D)+1 \rceil}$	PGD
Cheng et al. [13]	$\mathbf{u}_i \sim U, \hat{\mathbf{g}} = \frac{1}{N}\sum_i^N (\sqrt{\lambda}\cdot\mathbf{v} + \sqrt{1-\lambda}\cdot\frac{(\mathbf{I}-\mathbf{v}\mathbf{v}^T)\mathbf{u}_i}{\|(\mathbf{I}-\mathbf{v}\mathbf{v}^T)\mathbf{u}_i\|_2})$	N	PGD

Links

- 论文链接：Suya F, Chi J, Evans D, et al. Hybrid batch attacks: Finding black-box adversarial examples with limited queries[C]//29th {USENIX} Security Symposium (USENIX Security 2020). 2020: 1327-1344.
- 论文代码：https://github.com/suyeecav/Hybrid-Attack

Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks

Contribution

1. 提出了一种新的对抗攻击算法 APGD，且使得防御算法的测试变得更加自动化；
2. 提出了一种新的 loss 损失函数用于对抗样本的生成；

Notes

1. 文章思想：

   (1) 目的：提出一个能够自动调参的，且由于现在使用交叉熵损失函数的 PGD 的自动测试框架；

   (2) PGD 存在的问题：

   - Fixed step size：步长是固定的；
   - Agnostic of the budget：算法经过一定轮数后，样本的 loss 就不再改变了，故无法用迭代的论述来衡量一个对抗攻击算法的强度；
   - Unaware of the trend：算法不知道迭代的过程是否让样本更优化，更无法对这个做出改变；

2. ☆ **Auto-PGD 算法**：

(1) 算法伪代码：

Algorithm 1 APGD

Input: f, S, $x^{(0)}$, η, N_{iter}, $W = \{w_0, \ldots, w_n\}$
Output: $x_{\max}$, $f_{\max}$
$x^{(1)} \leftarrow P_{\mathcal{S}}\left(x^{(0)} + \eta \nabla f(x^{(0)})\right)$
$f_{\max} \leftarrow \max\{f(x^{(0)}), f(x^{(1)})\}$
$x_{\max} \leftarrow x^{(0)}$ **if** $f_{\max} \equiv f(x^{(0)})$ **else** $x_{\max} \leftarrow x^{(1)}$
for $k = 1$ **to** $N_{\text{iter}} - 1$ **do**
  $z^{(k+1)} \leftarrow P_{\mathcal{S}}\left(x^{(k)} + \eta \nabla f(x^{(k)})\right)$
  $x^{(k+1)} \leftarrow P_{\mathcal{S}}\Big(x^{(k)} + \alpha(z^{(k+1)} - x^{(k)}) + (1-\alpha)(x^{(k)} - x^{(k-1)})\Big)$
  if $f(x^{(k+1)}) > f_{\max}$ **then**
    $x_{\max} \leftarrow x^{(k+1)}$ and $f_{\max} \leftarrow f(x^{(k+1)})$
  end if
  if $k \in W$ **then**
    if Condition 1 **or** Condition 2 **then**
      $\eta \leftarrow \eta/2$ and $x^{(k+1)} \leftarrow x_{\max}$
    end if
  end if
end for

(2) **对抗样本的更新**：和 PGD 不同之处是，文章算法更新的时候添加了一个 Momentum 操作，如下

$$
\begin{aligned}
z^{(k+1)} &= P_{\mathcal{S}}\left(x^{(k)} + \eta^{(k)} \nabla f(x^{(k)})\right) \\
x^{(k+1)} &= P_{\mathcal{S}}\Big(x^{(k)} + \alpha \cdot (z^{(k+1)} - x^{(k)}) \\
&\qquad + (1-\alpha) \cdot (x^{(k)} - x^{(k-1)})\Big)
\end{aligned}
$$

其中，$\alpha \in [0, 1]$ 是动量系数，文章中使用的是 0.75；

(3) **步长的更新**：如果对抗样本的迭代满足以下两个条件中的一个，则更新步长（文章中，初始化步长为 $\eta^{(0)} = 2\epsilon$）

- 条件一：如果迭代对抗样本使得其目标更优化的比率低于某个值，则更新步长；

$$
\sum_{i=w_{j-1}}^{w_j - 1} \mathbf{1}_{f(x^{(i+1)}) > f(x^{(i)})} < \rho \cdot (w_j - w_{j-1})
$$

  其中 ρ 在文章中取值为 0.75，w_j 是迭代轮数；

- 条件二：如果上一轮步长迭代中未修改步长，而上一轮对抗样本的更新中目标没有得到任何优化，则更新步长；

$$
\eta^{(w_{j-1})} \equiv \eta^{(w_j)} \text{ and } f_{\max}^{(w_{j-1})} \equiv f_{\max}^{(w_j)}
$$

(4) **从最优点继续迭代**：每次更新步长后，都将对抗样本置为前面的最优结果，再进行迭代，即

$$x^{(w_j+1)} := x_{\max}$$

(5) **步长检查的迭代轮数选择**：

$$w_j = \lceil p_j N_{\text{iter}} \rceil \leq N_{\text{iter}}$$

$$p_{j+1} = p_j + \max\{p_j - p_{j-1} - 0.03, 0.06\}$$

其中，$p_0 = 0$, $p_1 = 0.22$；

(6) 实验效果：这里使用的对比算法是带有 Momentum 的 PGD 算法；

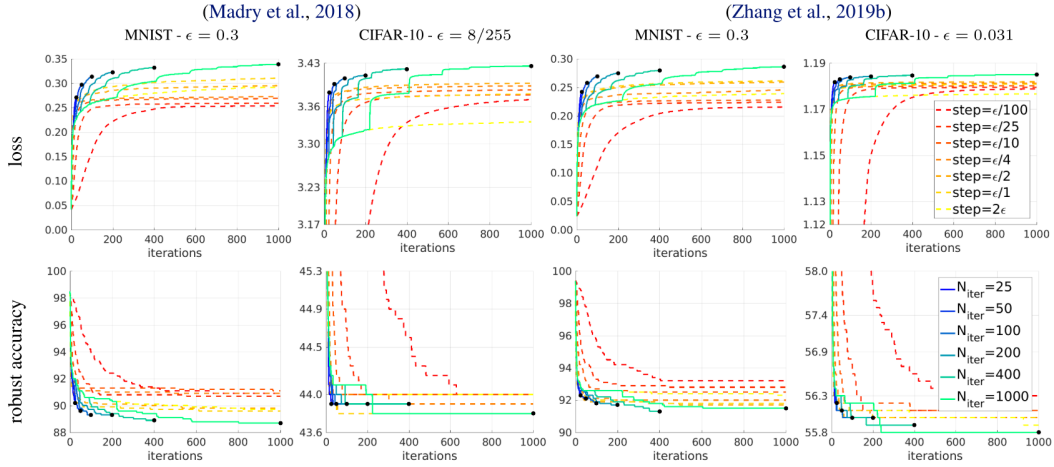


Figure 1. **PGD with Momentum vs APGD:** best cross-entropy loss (top) and robust accuracy (bottom) obtained so far as function of iterations for the models of (Madry et al., 2018) and TRADES (Zhang et al., 2019b) for PGD with a momentum term ($\alpha = 0.75$, as used in APGD) (dashed lines) with different fixed step sizes (always 1000 iterations) and APGD (solid lines) with different budgets of iterations. APGD outperforms PGD with Momentum for every budget of iterations in achieved loss and almost always in robust accuracy.

3. ☆ **DLR 损失函数**

(1) 交叉熵损失函数存在的问题：这边的交叉熵损失函数，实际上在程序中是 Softmax-交叉熵损失函数。正是由于 softmax 函数的存在，导致对抗样本生成过程中，如果原始样本的分类十分接近0，就很难产生一个有效的梯度。作者的实验如下

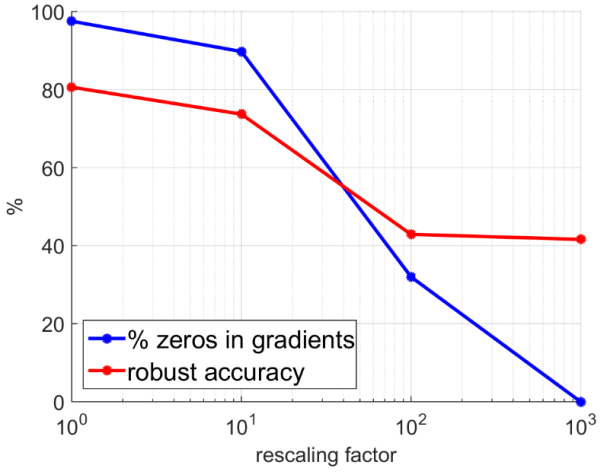


Figure 2. Percentage of zeros in the gradients and robust accuracy, computed by PGD on the CE loss, of the classifiers g/α, where g is the CIFAR-10 model of (Atzmon et al., 2019) and α a rescaling factor. The performance of PGD depends on the scale of the logits.

(2) DLR (Difference of Logits Ratio) 损失函数：作者提出了一个新的损失函数，如下

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

其中，π_1 和 π_3 指的是概率降序排序后的第一、第三个概率值；

Links

- 论文链接：Croce F, Hein M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks[C]//International Conference on Machine Learning. PMLR, 2020: 2206-2216.
- 论文代码：https://github.com/fra31/auto-attack

WITCHcraft: Efficient PGD attacks with random step size

Contribution

1. 提出随机化学习率能够更好地生成对抗样本；

Notes

1. 在 PGD 的基础上，每一步学习率都随机选取，算法如下：

Algorithm 1: The WITCHcraft attack algorithm.

Requires: Network f, input $\mathbf{x}$, label y, permissible perturbation set $\mathcal{S}$, number of steps n, and expected step size parameter a.

Initialize perturbation δ with entries distributed independently according to distribution $\mathcal{U}(S)$.

for step = 1,..., n **do**

  Sample τ with entries distributed independently according to distribution $\mathcal{U}(0, 2a)$.

  $\delta \leftarrow \Pi_{\mathcal{S}}[\delta + \tau \odot \text{sign}(\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x} + \delta, \text{class})$

  If $\arg\max(f(\mathbf{x} + \delta)) \neq y$, return $\mathbf{x} + \delta$ and **break**.

2. 实验结果

   (1) 和 PGD 方法相比，效果更好：

Attack	CIFAR-10 $\mathcal{A}_{adv}$
20-step PGD	47.04%
20-step WITCHcraft	**45.92%**
100-step PGD	45.29%
100-step WITCHcraft	**45.20%**
20-PGD w/ 10 restarts	45.21%

Table 1. Robust accuracy, $\mathcal{A}_{adv}$, of various adversarial attacks against the WideResNet(34-10) model trained on CIFAR-10, and released by the authors of [5]. Bolded entries indicate best attack results across fixed computational complexity. Randomized coordinate-wise learning rates (WITCHcraft) improve attack effectiveness with a fixed computational budget.

Attack	MNIST $\mathcal{A}_{adv}$
100-step PGD	92.52%
100-step WITCHcraft	**91.68%**
500-step PGD	91.91%
500-step WITCHcraft	**91.00%**

Table 2. Robust accuracy, $\mathcal{A}_{adv}$, of various adversarial attacks against the two-layer CNN model trained on MNIST and released by the authors of [5]. Bolded entries indicate the best attack results across fixed computational complexity. Like we observed for the CIFAR-10 model, randomized coordinate-wise learning rates improve attack effectiveness with a fixed computational budget.

(2) 不同的学习率区间对结果的影响：

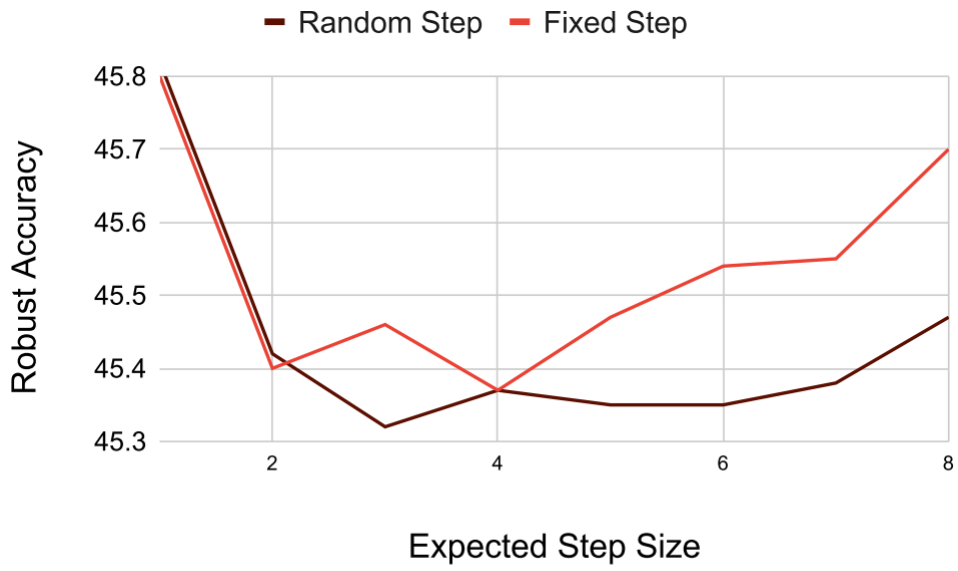


Fig. 2. Sensitivity plot of a 40-step PGD attack compared with 40-step WITCHcraft for the CIFAR-10 challenge. The horizontal axis represents expected step size and the vertical axis represents robust accuracy. We see that the randomized step size choice in WITCHcraft out-performs a deterministic step size choice, particularly when larger step sizes are used.

Links

- 论文链接：Chiang P Y, Geiping J, Goldblum M, et al. Witchcraft: Efficient pgd attacks with random step size[C]//ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020: 3747-3751.

An Alternative Surrogate Loss for PGD-based Adversarial Testing

Contribution

1. 提出了一种新的 MultiTargeted 攻击方法；

Notes

1. 形式化 PGD 算法：

Algorithm 1 White-box PGD-based attack

Input: A nominal input x and label y, a model f_θ and an adversarial input set $\mathcal{S}(x)$. $\alpha^{(k)}$ is the step-size at iteration k and `Opt` is an optimizer (e.g., `Adam`).
Output: Possible attack $\hat{\xi} \in \mathcal{S}(x)$

$\hat{\xi} \leftarrow x$
for $r \in \{1, \dots, N_r\}$ **do** ▷ Repeat the attack process N_r times
  Initialize surrogate loss $\hat{L}^{(r)}$ ▷ The surrogate loss may be different across restarts (see Sec. 3)
  Initialize optimizer `Opt` ▷ The optimizer ingests gradients and computes an update
  $\xi^{(0)} \leftarrow \texttt{SampleFrom}(x, \mathcal{S})$ ▷ Samples a random input that respects the threat model
  for $k \in \{1, \dots, K\}$ **do** ▷ K corresponds to the number of optimization steps
    $\xi^{(k)} \leftarrow \texttt{Proj}_{\mathcal{S}(x)}\left(\xi^{(k-1)} + \alpha^{(k)}\texttt{Opt}\left(\nabla_{\xi^{(k-1)}}\hat{L}^{(r)}(f_\theta(\xi^{(k-1)}), y)\right)\right)$
    if $L(f_\theta(\xi^{(k)}), y) > L(f_\theta(\hat{\xi}), y)$ **then** ▷ L is the loss of interest (e.g., $0-1$ loss)
      $\hat{\xi} \leftarrow \xi^{(k)}$ ▷ If L is the $0-1$ loss, the procedure can terminate early
    end if
  end for
end for

这边可能影响 PGD 算法攻击性的 **因素** 有：重复执行的次数 N_r ，计算梯度时使用的损失函数 $\hat{L}^{(r)}$ ，迭代优化器 Opt，噪声生成器 $SampleForm$，每轮的迭代次数 K，迭代的学习率 $a^{(k)}$ 。其中，L 是用来判断对抗攻击是否成功的判别函数，对最终的结果没什么影响，可以用来控制迭代的次数，实现提早停止迭代的功能；

2. MultiTargeted Attack：主要思想是改变 restarts 过程中使用的损失函数如下：

$$\hat{L}^{(r)}(z, y) = -z_y + z_{\mathcal{T}_r \bmod |\mathcal{T}|} \text{ with } \mathcal{T} = \{1, \dots, C\} \setminus \{y\}$$

对上面伪代码的 $2 \sim 3$ 行进行修改：

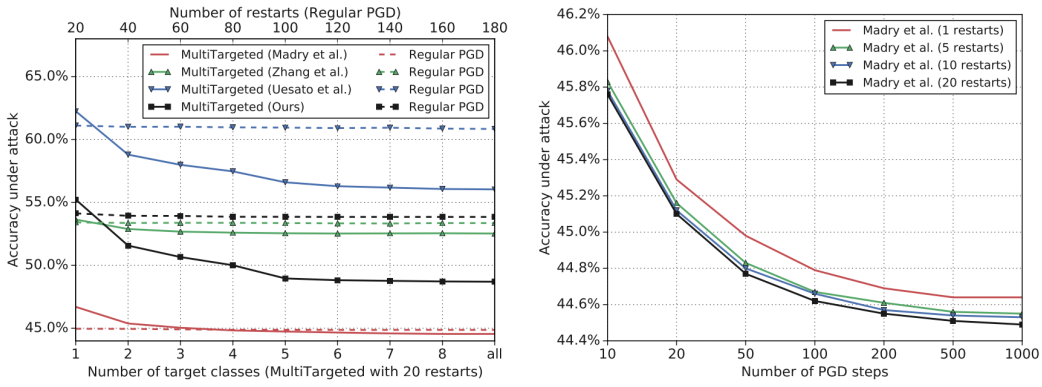
Algorithm 2 `MultiTargeted` attack

2a: $N_i \leftarrow \lfloor N_r / |\mathcal{T}| \rfloor$ ▷ Keep the number of iterations consistent
2b: **for** $i \in \{1, \dots, N_i\}$ **do**
2c: **for** $j \in \{1, \dots, |\mathcal{T}|\}$ **do**
3: Use surrogate loss $\hat{L}^{(i|\mathcal{T}|+j+1)}(z, y) = -z_y + z_t$ with $t = \mathcal{T}_j$
  ...
11: **end for**
12: **end for**

为了减小复杂性，可以把多目标限定在 $Top-K$ 中：

$$\mathcal{T} = \{t \quad | \quad T > |\{i \text{ such that } i \neq y \text{ and } z_i > z_t\}|\}$$

3. 实验：



(a) (b)

Figure 3 | Panel a shows the accuracy under attacks of size $\epsilon = 8/255$ for four different CIFAR-10 models. The solid lines are the accuracies obtained by `MultiTargeted` for different values of T, while the dashed lines of the corresponding color are accuracies obtained by regular PGD (for the same computational budget). Panel b shows the accuracy under a `MultiTargeted` attack of size $\epsilon = 8/255$ for Madry et al.'s model as a function of the number of PGD steps K and the number of restarts N_i.

- 首先可以看到，MultiTargeted 方法相比于 PGD（带有 restarts）是有效的，实验中当多目标 $K = 5$ 时达到了最优状态；
- 过多的 restarts 次数对于 PGD 算法来说并不是特别有效（不过我理解的可能是每次 restart 时的随机噪声可能效果都相似，限制了 restarts 的发挥）；
- 每一个 restart 迭代更多的次数确实能够增强 PGD 的攻击性，但是可以看到这个增加的效果并不是特别明显；

Links

- 论文链接：Gowal S, Uesato J, Qin C, et al. An alternative surrogate loss for pgd-based adversarial testing[J]. arXiv preprint arXiv:1910.09338, 2019.

Minimally distorted Adversarial Examples with a Fast Adaptive Boundary Attack

Contribution

Notes

Links

- 论文链接：Croce F, Hein M. Minimally distorted adversarial examples with a fast adaptive boundary attack[C]//International Conference on Machine Learning. PMLR, 2020: 2196-2205.
- 论文代码：https://github.com/fra31/fab-attack

Composite Adversarial Attacks

Contribution

1. 像 AutoML 和 AutoAugment 一样，本文提出了一种自动搭配不同的对抗攻击基础算法和参数的 **自学习对抗攻击算法**；

Notes

1. 文章方法：核心思想是像 AutoML 和 AutoAugment 那样让程序自动寻找更好的攻击方法和参数的搭配，以达到最优效果，搜索算法用的是 $NSGA$-Ⅱ，伪代码如下

Algorithm 1 Attack policy search using NSGA-II

Require: Pool of candidate attackers $\mathbb{A}$; Population size P;
Require: Maximum number of generations G

$P_0 \leftarrow \emptyset$ ▷ Initialized population with size of K
$t \leftarrow 0$
for $i \leftarrow 1$ to K **do**
  for $j \leftarrow 1$ to N **do**
    Random sample $\mathcal{A}_j$ from $\mathbb{A}$
    Random sample $\epsilon_j \sim [0, \epsilon_{max}]$, $t_j \sim [0, t_{max}]$
  end for
  $s \leftarrow \mathcal{A}_N(\mathcal{A}_1(x, \mathcal{F}; \epsilon_1, t_1)...), \mathcal{F}; \epsilon_N, t_N)$
  $P_0 \leftarrow P_0 \cup s$
end for
for $t < G$ **do** ▷ Run search with Eq. 3 for evaluation
  $P_{t+1} \leftarrow$ NSGA-II(P_t) ▷ Update the populations
  $t \leftarrow t + 1$
end for
return best attack policy s^* from P_t

整体的流程就是首先构造一个候选集，然后在候选集中验证各种搭配的得分情况，这里得分的计算公式如下：

$$\mathcal{L} = -\sum_{x} [\mathcal{F}(s(x)) \neq y] + \alpha \sum_{i=0}^{N} t_{s_i}$$

左式用来评估算法的成功率，右式用来评估算法的复杂度；

2. 实验结果：（因为比赛的缘故，我更加关心哪种搭配能够产生最好的攻击成功率，所以 S_{l_∞} 可以参考一下）

Visualization of CAA$_{sub}$ proxy attack policies	
S_{l_∞}	[('MT-LinfAttack', ϵ=8/255, t=50), ('MT-LinfAttack', ϵ=8/255, t=25), ('CWLinfAttack', ϵ=8/255, t=125)]
S_{l_2}	[('MT-L2Attack', ϵ=0.5, t=100), ('PGD-L2Attack', ϵ=0.4375, t=125), ('DDNAttack', t=1000)]
$S_{unrestricted}$	[('FogAttack', $\epsilon = 1, t = 1$), ('FogAttack', $\epsilon = 1, t = 1$), ('SPSAAttack', ϵ=16/255, t=100)]

CIFAR-10 - l_∞ - $\epsilon = 8/255$	AdvTrain	TRADES	AdvPT	MMA	JEM	PCL	Semi-Adv	Complexity
PGD (Madry et al. 2017)	51.95	53.47	57.21	50.04	9.21	8.12	61.83	1000
FAB (Croce and Hein 2019)	49.81	51.70	55.27	42.47	62.71	0.71	60.12	1350
APGD (Croce and Hein 2020)	51.27	53.25	56.76	49.88	9.06	7.96	61.29	1000
AA (Croce and Hein 2020)	49.25	51.28	54.92	41.44	8.15	0.28	59.53	4850
ODI-PGD (Tashiro 2020)	49.37	51.29	54.94	41.75	8.62	0.53	59.61	3200
BestAttack on S_{l_∞}	50.12	52.01	55.23	41.85	9.12	0.84	60.74	900
EnsAttack on S_{l_∞}	49.58	51.51	55.02	41.56	8.33	0.73	60.12	800
CAA$_{sub}$ on S_{l_∞}	49.18	51.19	54.82	40.87	7.47	0.0	59.45	**800**
CAA$_{dic}$ on S_{l_∞}	**49.18**	**51.10**	**54.69**	**40.69**	**7.28**	**0.0**	**59.38**	-

Table 2: The table is divided into two parts. The lower part presents the reported RA(%) of l_∞-based attack on diverse CIFAR-10 defenses. Each column presents the result on a specific defense and the last column presents the complexity of the attack algorithm. The upper part of the table presents the best attack policies found by our method.

Links

- 论文链接：Mao X, Chen Y, Wang S, et al. Composite Adversarial Attacks[J]. arXiv preprint arXiv:2012.05434, 2020.
- 论文代码：https://github.com/vtddggg/CAA