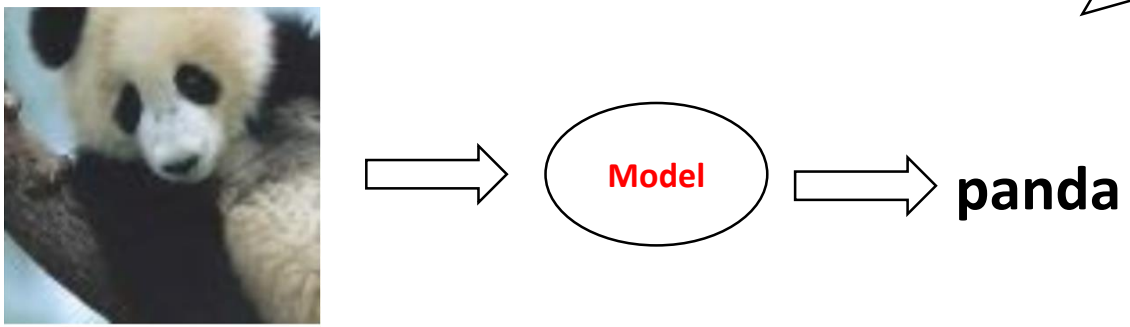
欺骗AI，从定向到实际

介绍人：汪嘉来

Part I：基于黑盒的定向攻击

背景

- DNNs正在被广泛引用，如图片分类、语音识别、恶意代码检测…



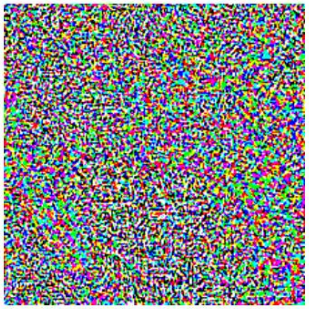
背景

- 神经网络很容易受到对抗样本的攻击：

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

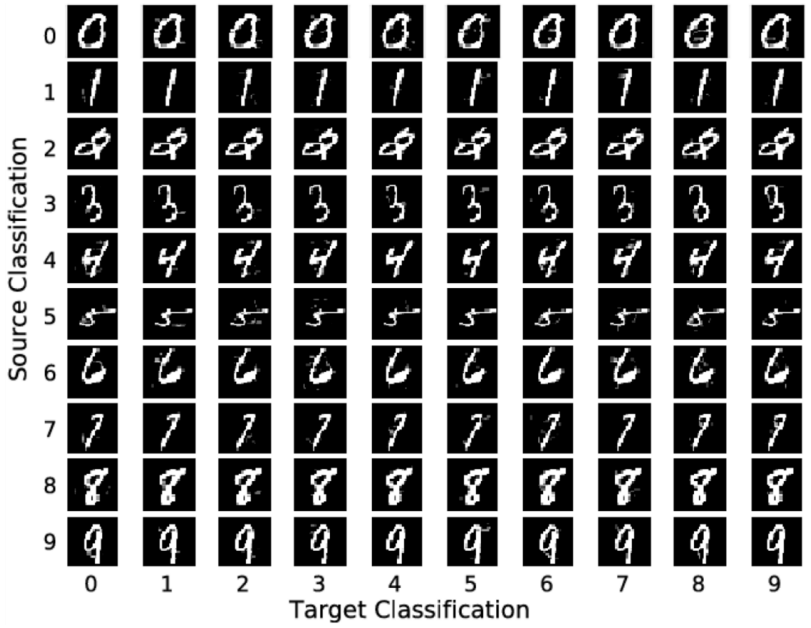
相关工作

- 白盒攻击：模型内部信息完全透明
  - Fast Gradient Sign(FGS)
  - Basic Iterative Method(BIM)
  - Jacobian-based Saliency Map Attack (JSMA)
  - Carlini & Wagner Attack (CW)
  - DeepFool

- 黑盒攻击：无法获取模型信息
  - Particle Swarm Optimization（PSO）
  - DE(Differential Evolution（DE）
  - ZOO
  - AutoZOO
  - Decision-based attack

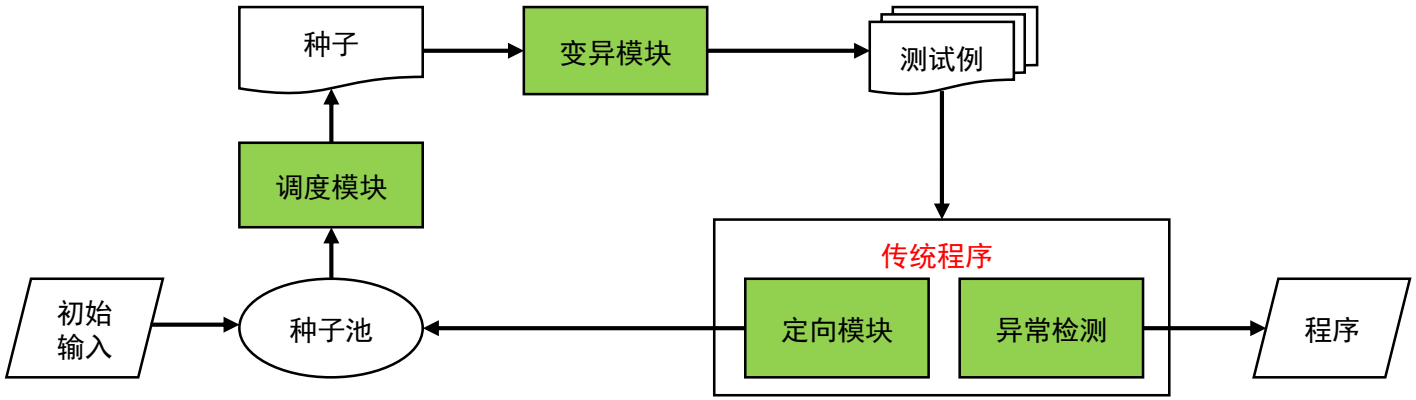
定向攻击

- 生成能够定向攻击神经网络的对抗样本:



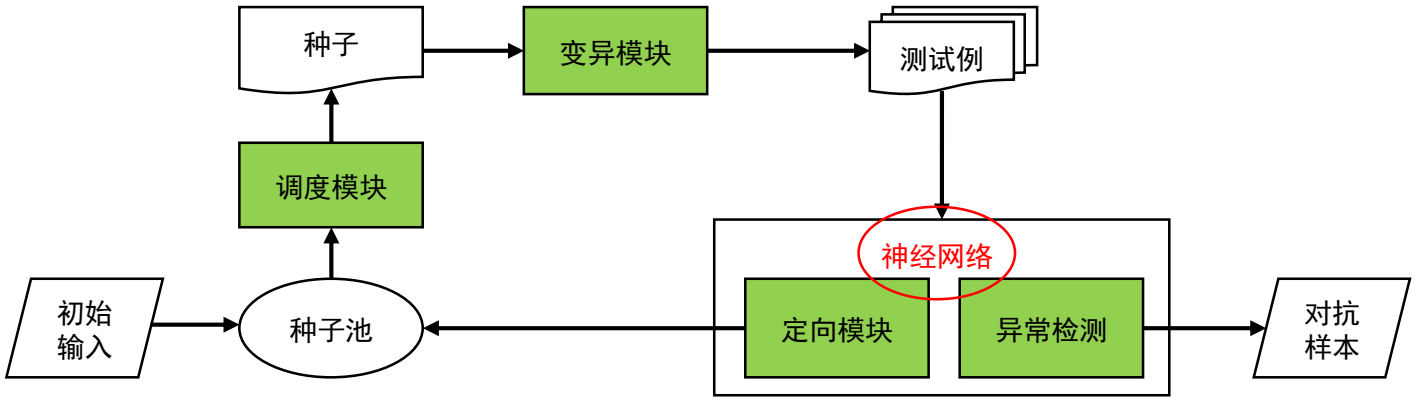
发掘传统程序中的漏洞

- 基于定向的模糊测试:

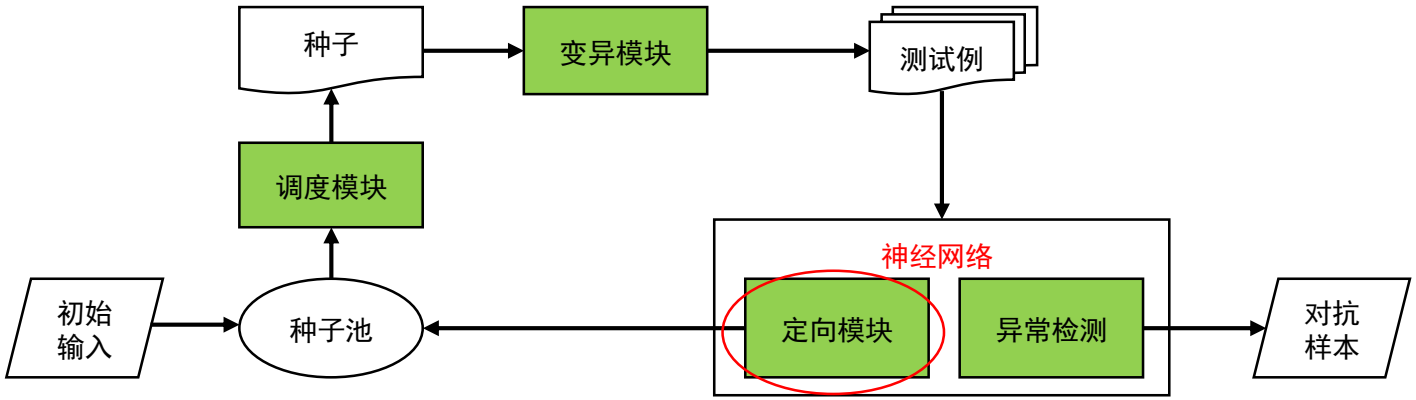


- 将能够缩短与定向目标距离的样例送入种子池

发掘神经网络中的漏洞

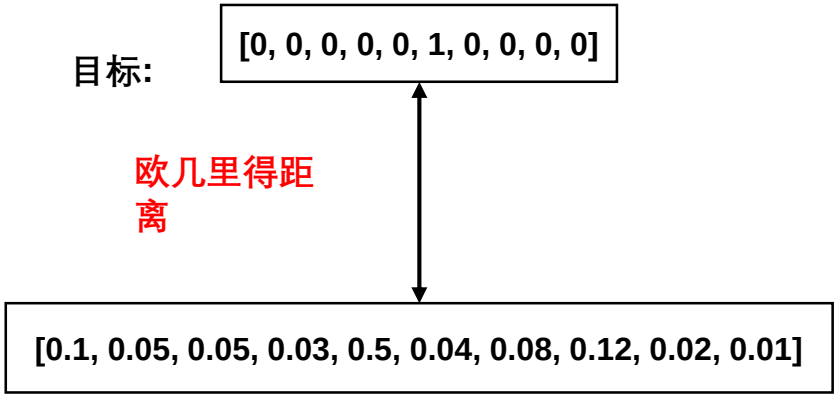


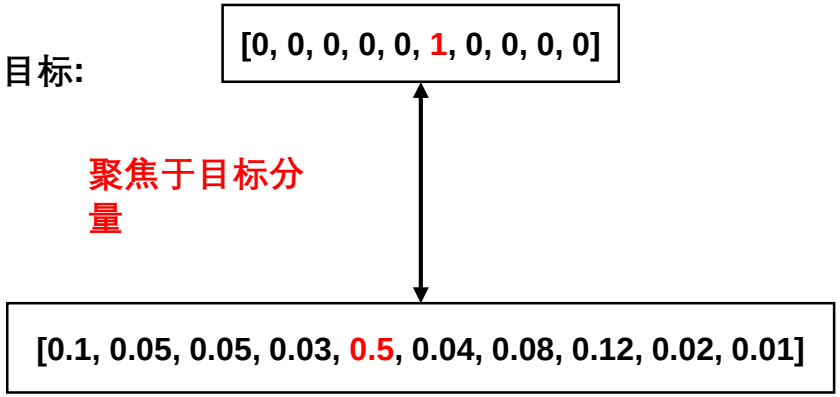
定向模块



定向模块

- 定向目标: [0,0,0,0,1,0,0,0,0,0]

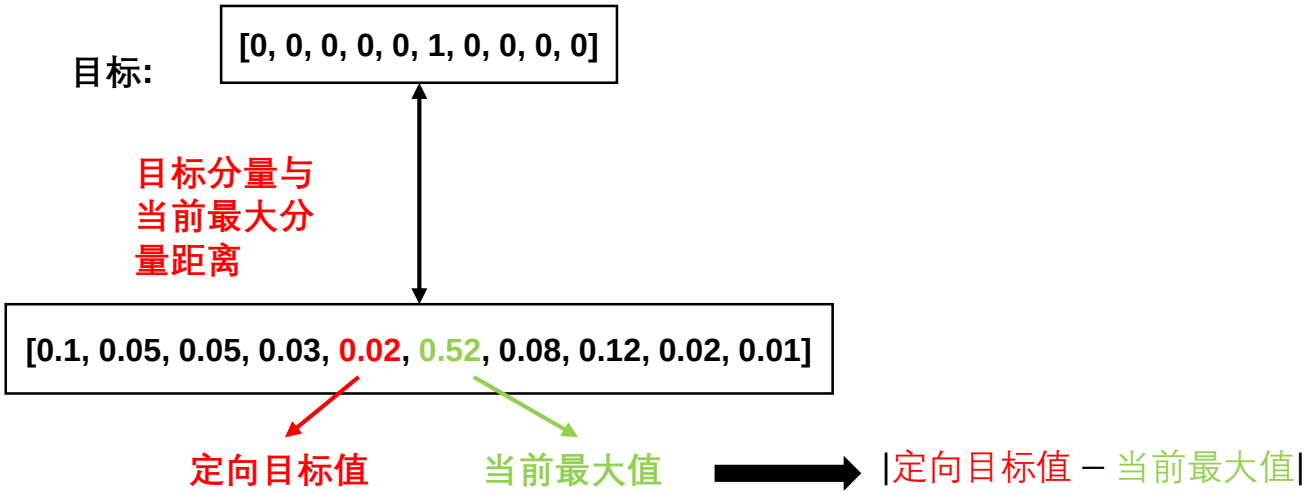




定向模块

- 定向目标: [0,0,0,0,1,0,0,0,0,0]

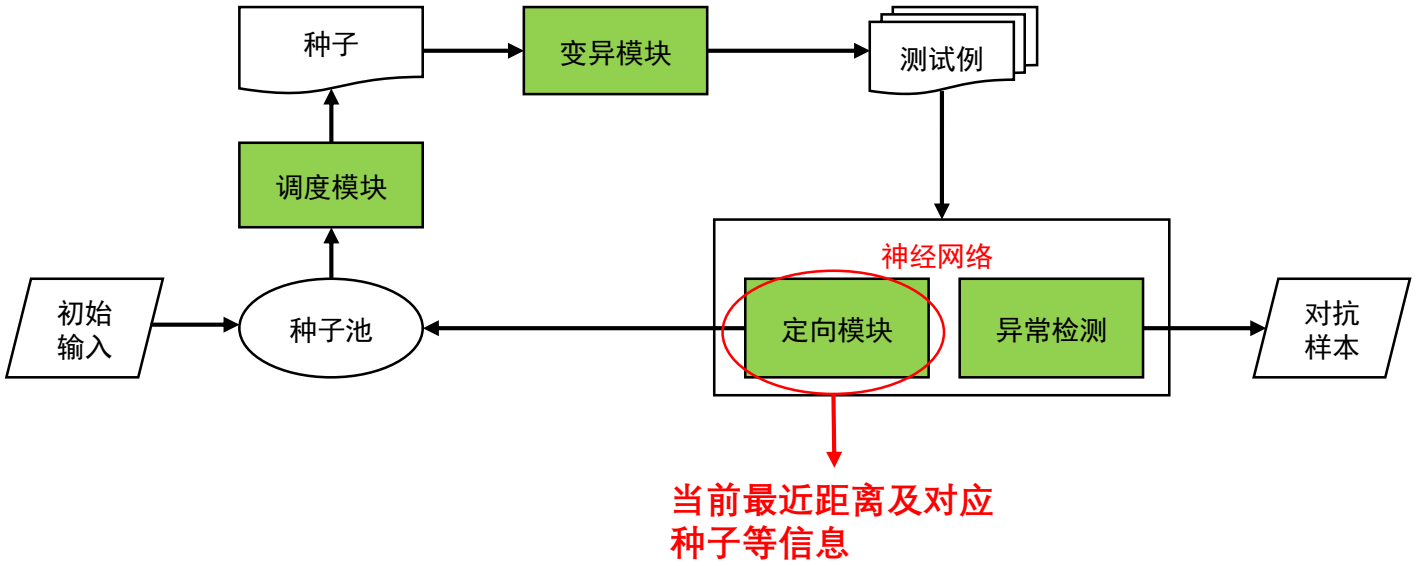
我们希望离目标更近



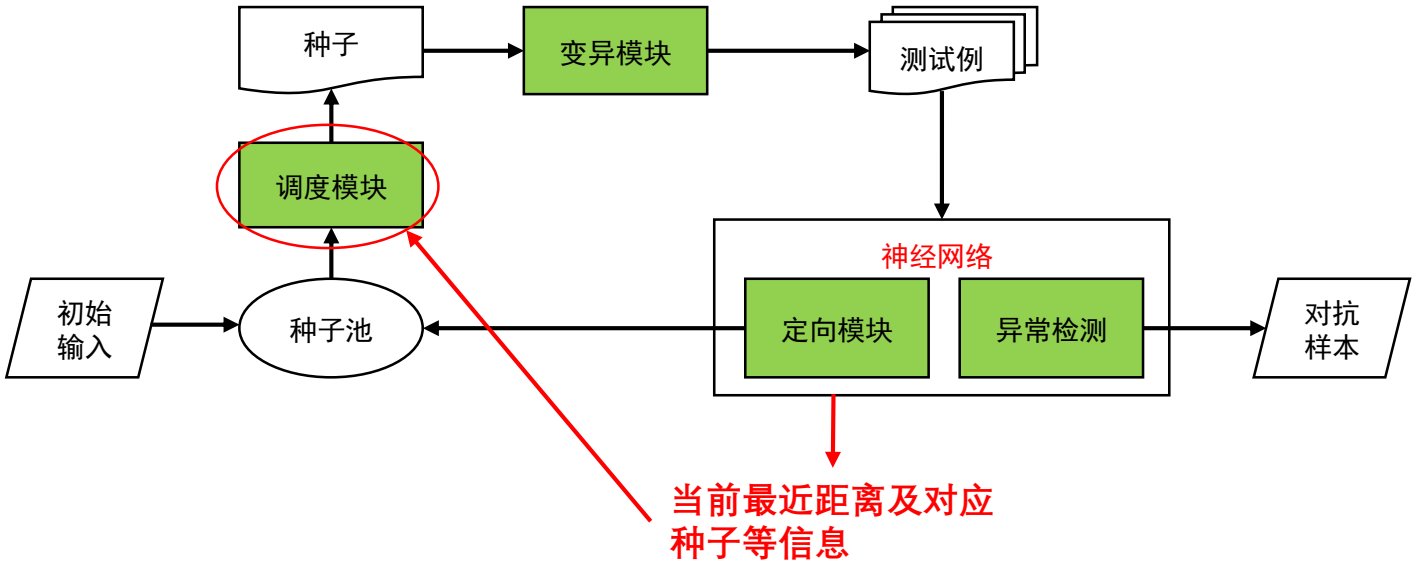
定向模块

- 确定定向标准后，我们就可以衡量与定向目标之间的距离
- 随着迭代的进行，会不断地更新最短距离

定向模块

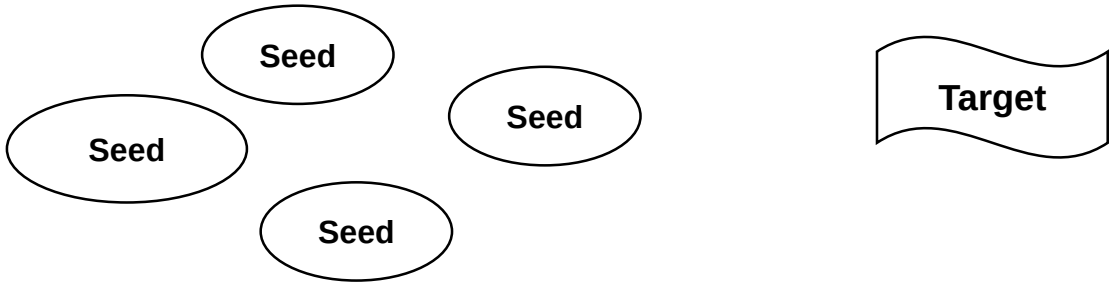

种子
变异模块
测试例
调度模块
神经网络
初始
输入
种子池
定向模块
异常检测
对抗
样本
当前最近距离及对应
种子等信息

调度模块


种子
变异模块
测试例
调度模块
神经网络
初始
输入
种子池
定向模块
异常检测
对抗
样本
当前最近距离及对应
种子等信息

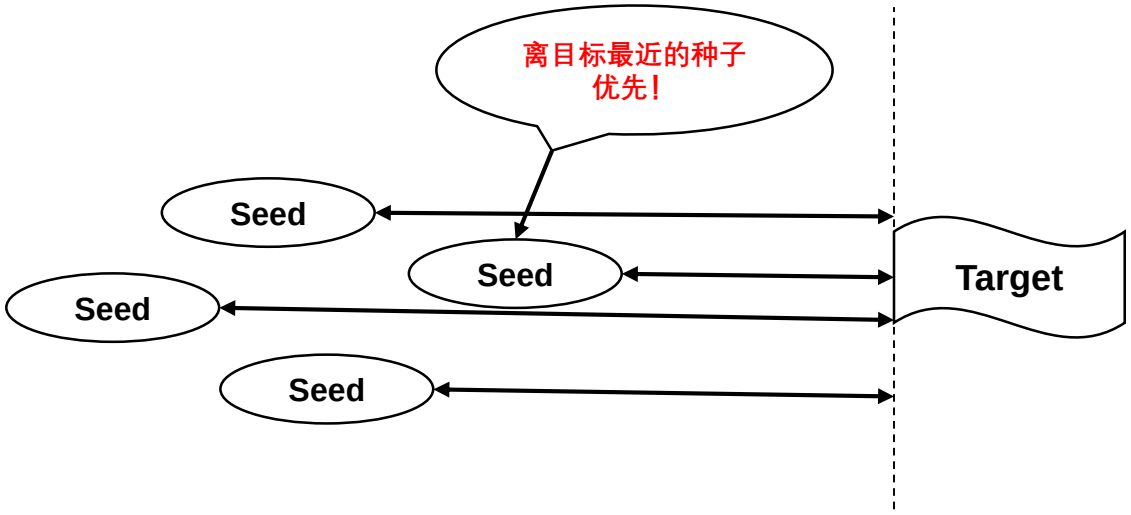
调度策略

采取策略优先变异某些种子:



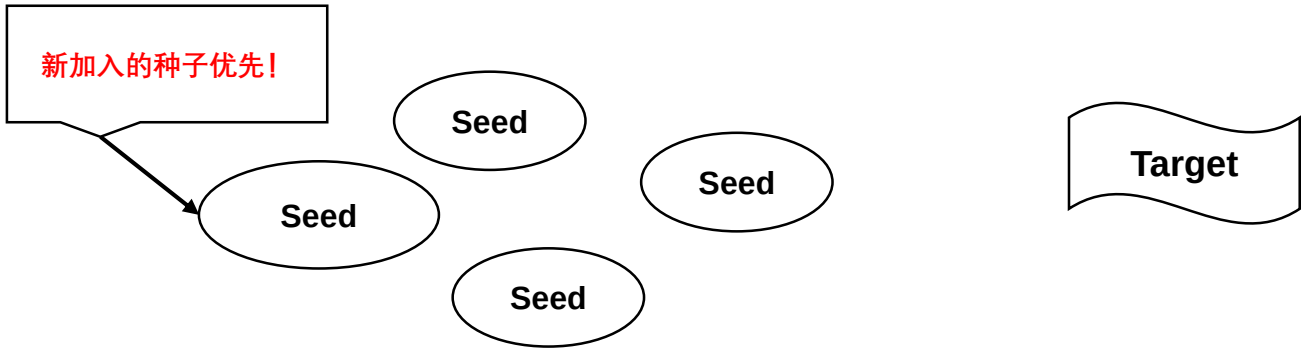
调度策略

首选策略：越接近于定向目标的种子，被选取的概率越高



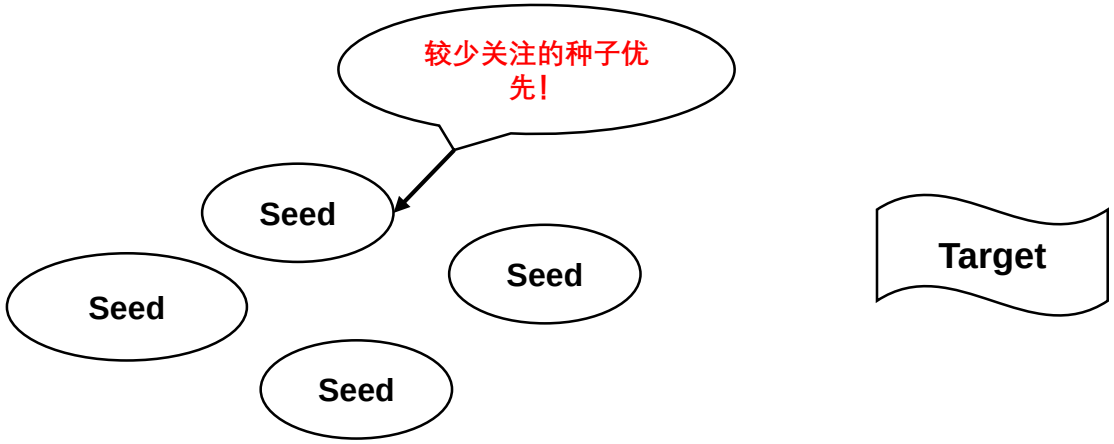
调度策略

候选策略：新加入的种子，被选取的概率更高



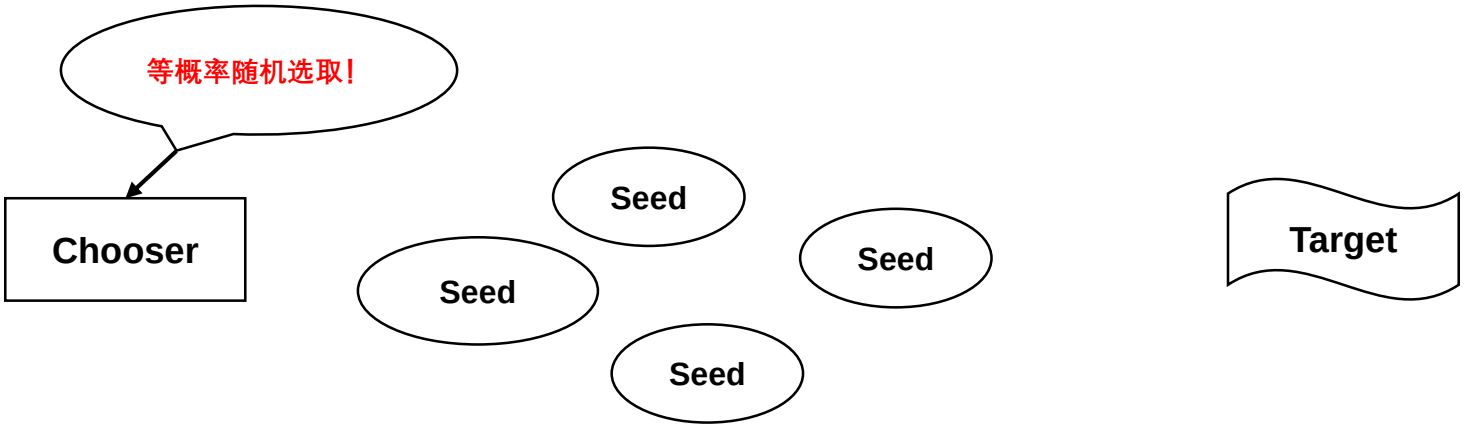
调度策略

候选策略：较少被挑选的种子，被选取的概率更高

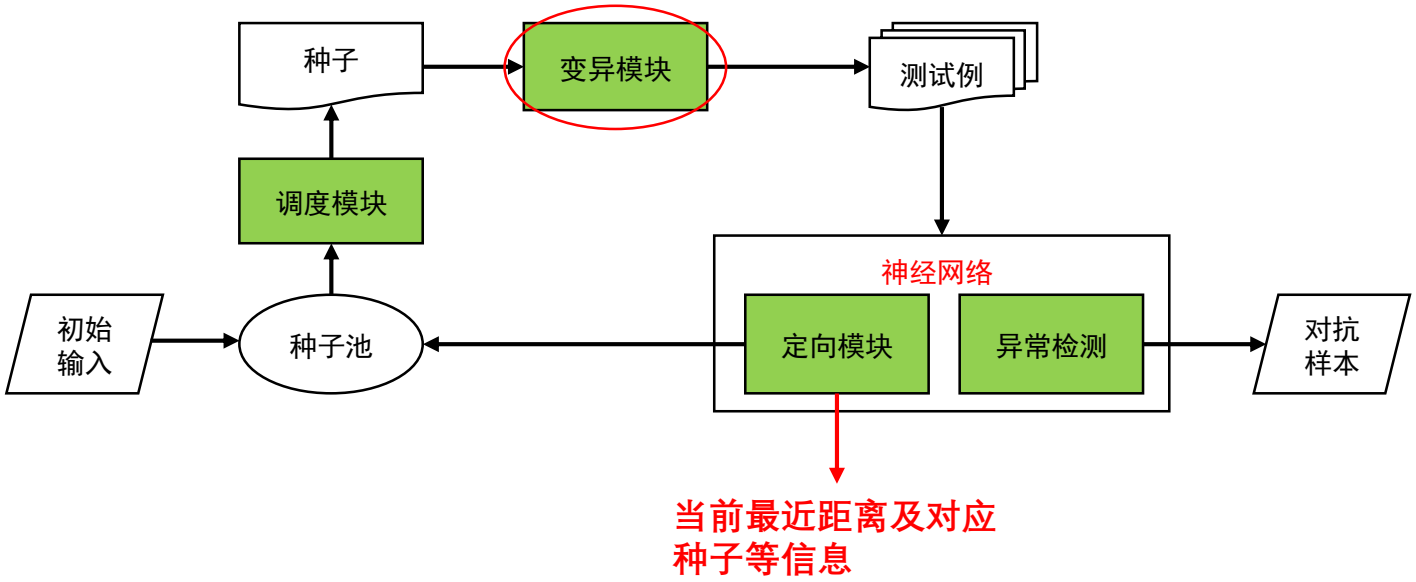


调度策略

候选策略：随机选取

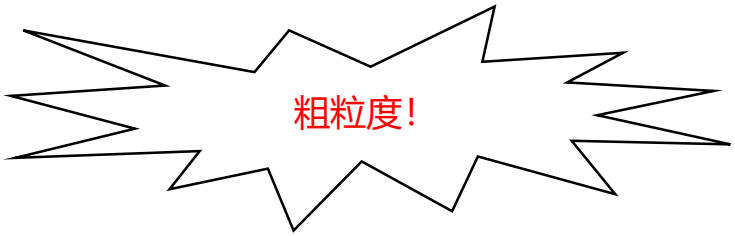


变异模块


种子
变异模块
测试例
调度模块
神经网络
初始
输入
种子池
定向模块
异常检测
对抗
样本
当前最近距离及对应
种子等信息

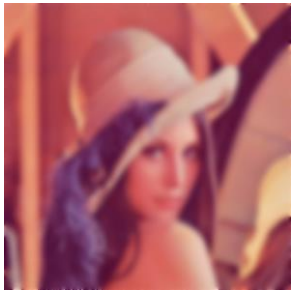
变异模块

- 常见的变异操作:



原始图

加噪声



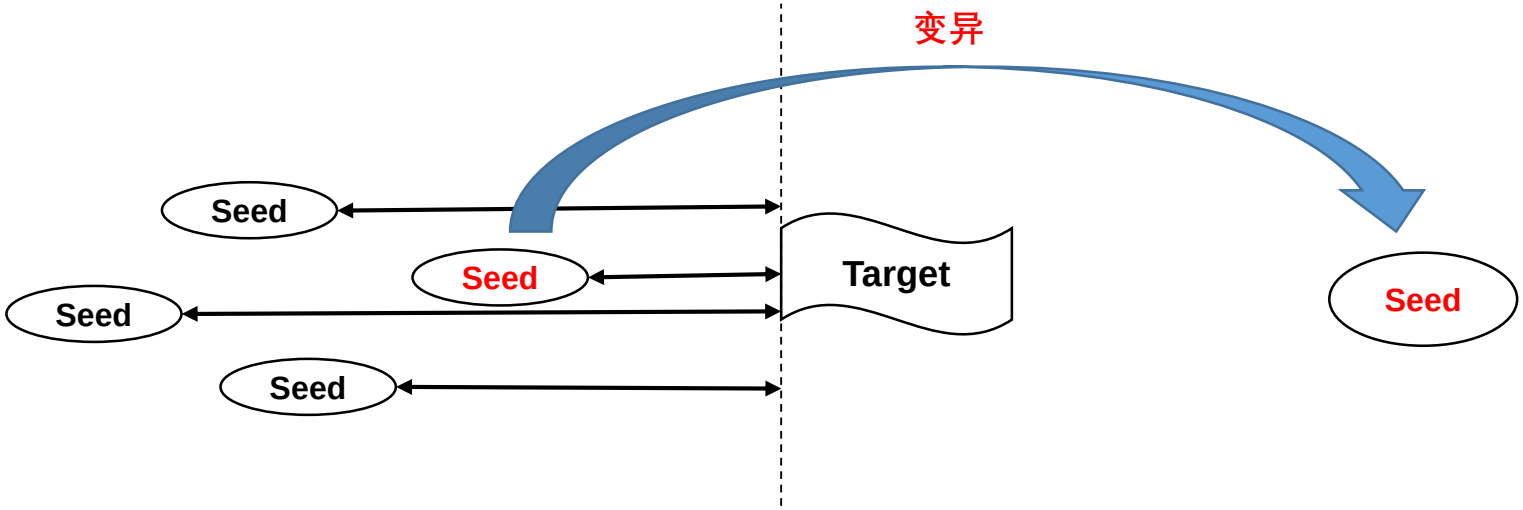
模糊化

平移化

粗粒度变异的后果

- 破坏原本最近种子的优势
- 毫无规律的变异类似于暴力破解

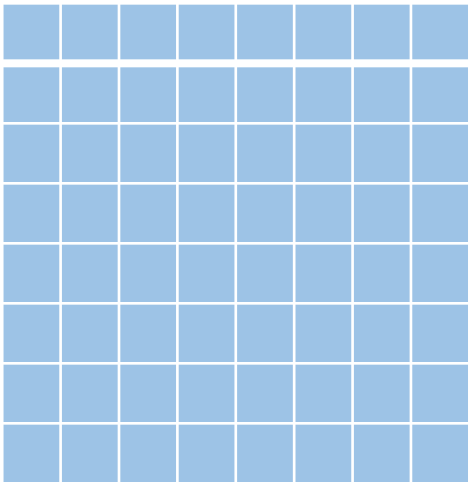
由于变化太大，距离更远了。。。



变异模块

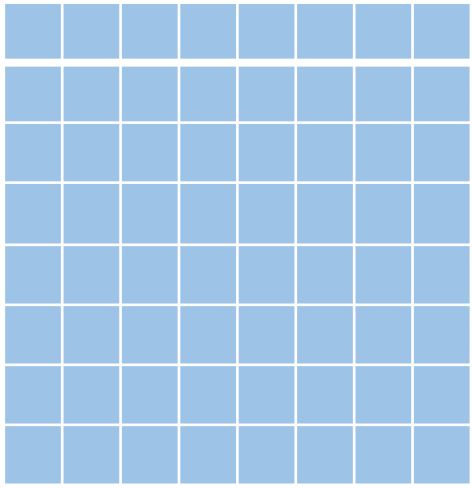
=

矩阵

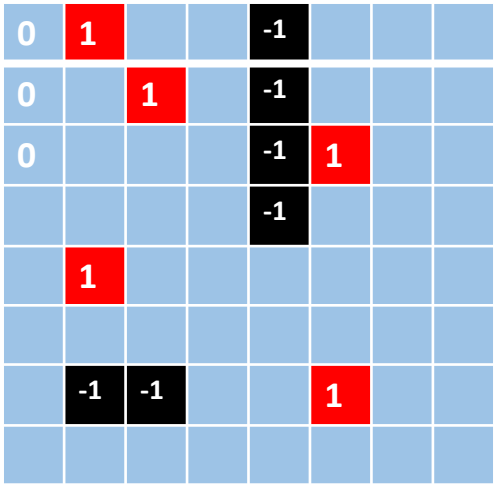


变异模块

矩阵



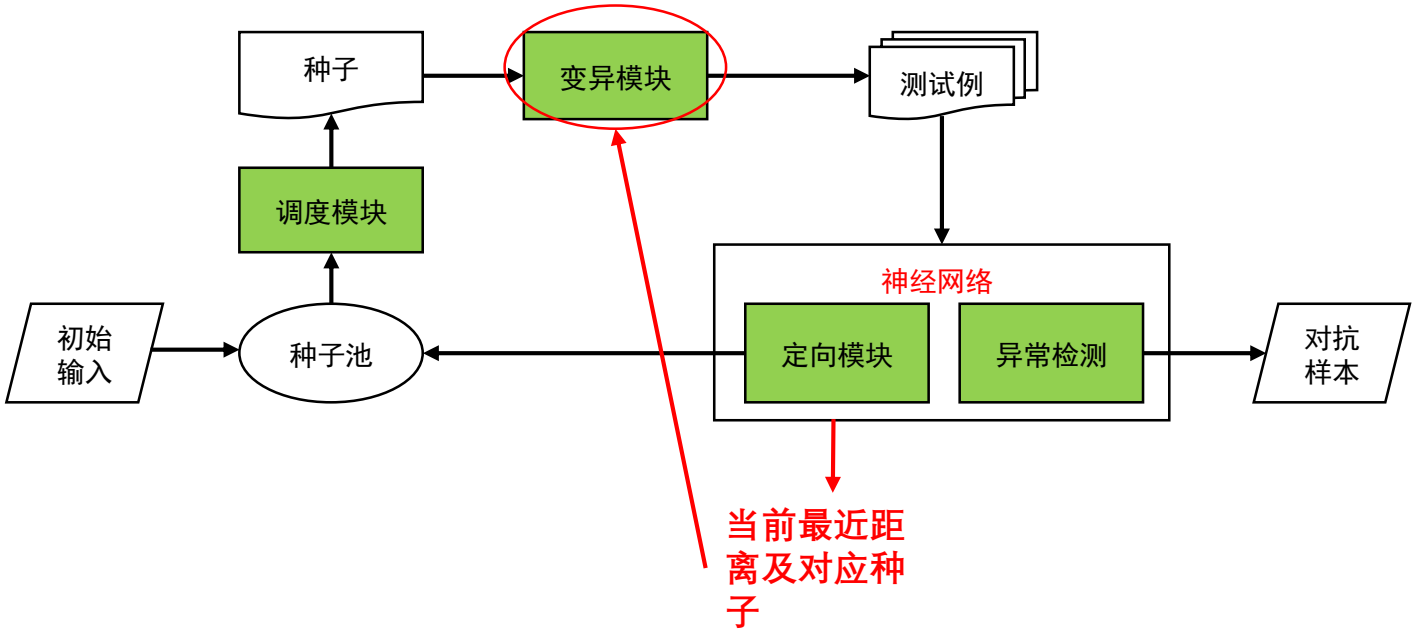
变异方向矩阵



红：1，增加　黑：-1，减少
蓝：0，不变

怎么调控三种值的比例？

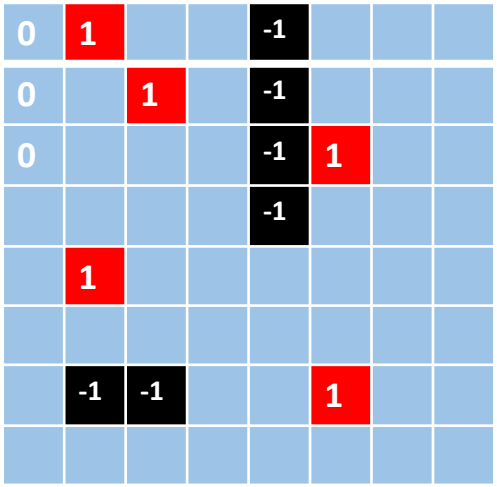
变异模块



变异模块

- 常见的变异操作:

变异方向矩阵



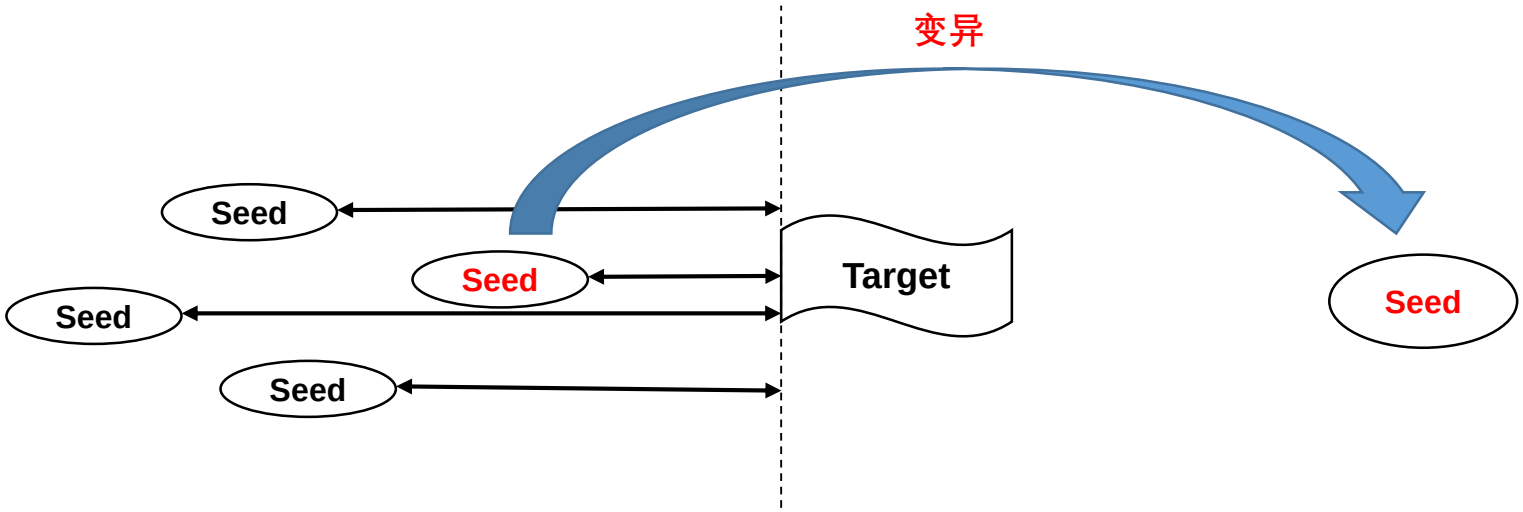
红：1，增加　黑：-1，减少
蓝：0，不变

由距离值调控：

- 与定向目标距离值较大时：1和-1的占比要大
- 与定向目标距离值较小时：1和-1的占比要调小

陷入局部最优

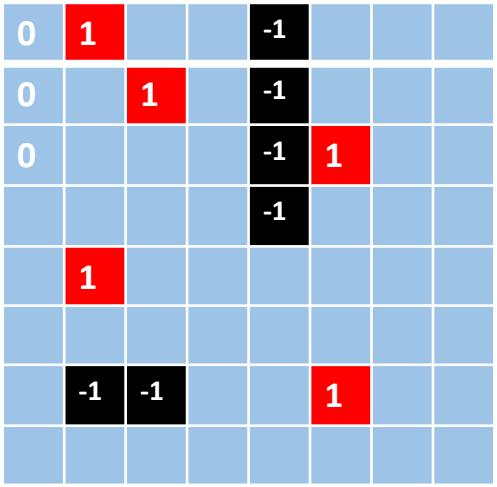
由于变化太大，距离更远了。。。



变异模块

- 常见的变异操作:

变异方向矩阵



红：1，增加　黑：-1，减少
蓝：0，不变

由距离值调控：

- 与定向目标距离值较大时：1和-1的占比要大
- 与定向目标距离值较小时：1和-1的占比要调小

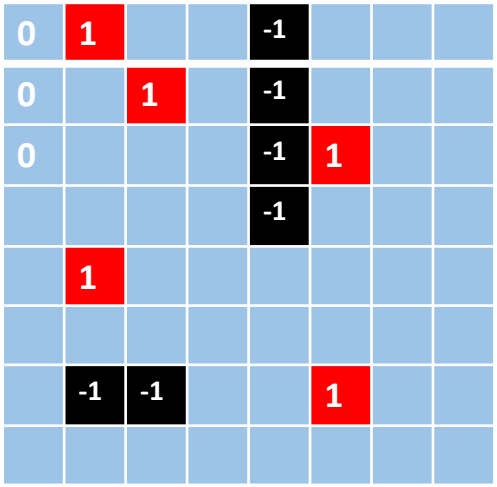
自动检测：

- 当发现陷入停滞时，强行动态调整

变异模块

- 常见的变异操作:

变异方向矩阵



红：1，增加　黑：-1，减少
蓝：0，不变

由距离值调控：

- 与定向目标距离值较大时：1和-1的占比要大
- 与定向目标距离值较小时：1和-1的占比要调小

自动检测：

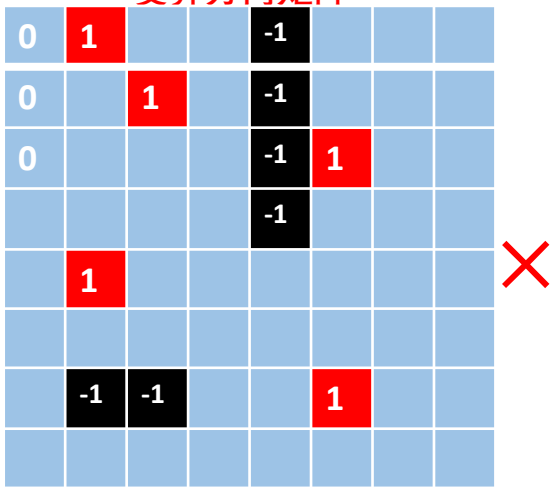
- 当发现陷入停滞时，强行动态调整

污点分析：

- 保留那些表现好的位置

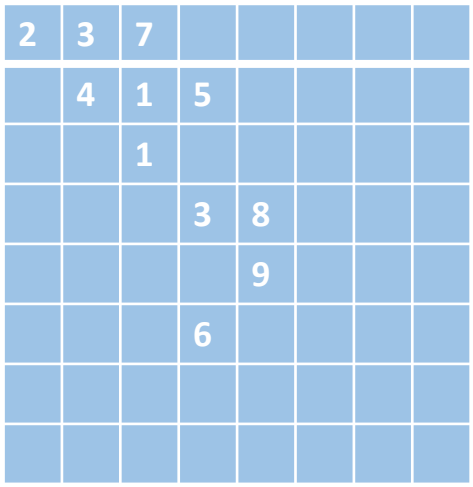
变异模块

变异方向矩阵



×

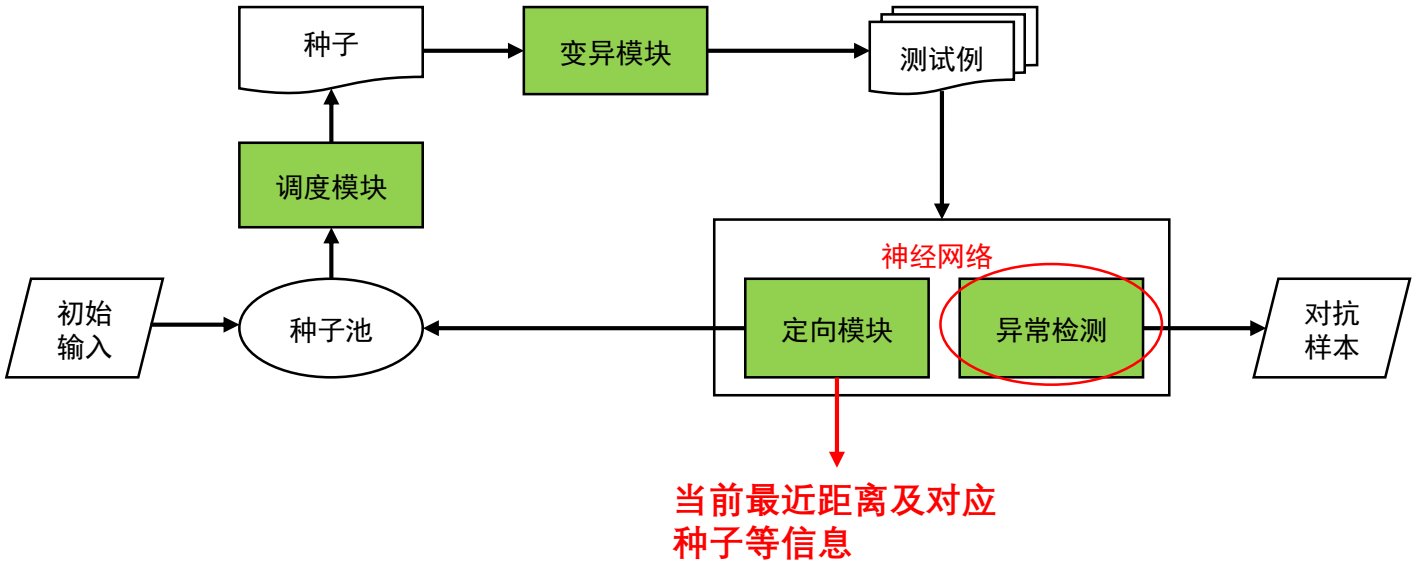
变异大小矩阵



红：1，增加　黑：-1，减少

蓝：0，不变

整体流程


种子
变异模块
测试例
调度模块
神经网络
初始
输入
种子池
定向模块
异常检测
对抗
样本
当前最近距离及对应
种子等信息

实验结果

原图

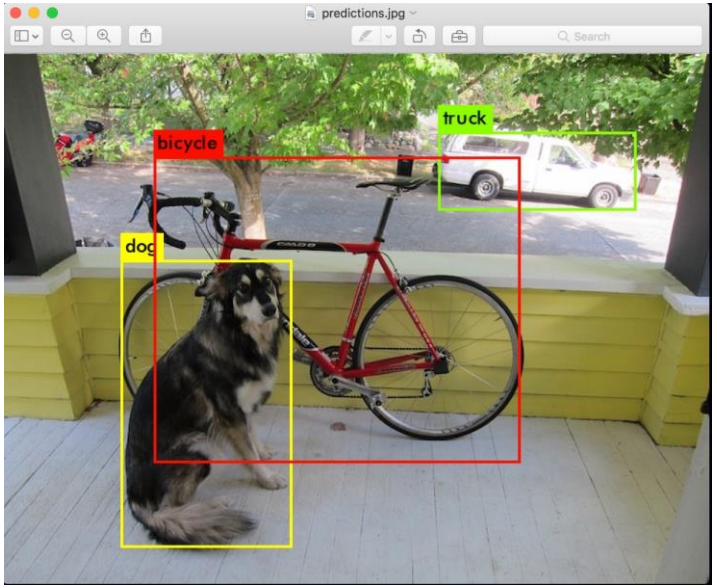
不同的定向目标

Part I：over

Part II：欺骗YOLOv3

背景

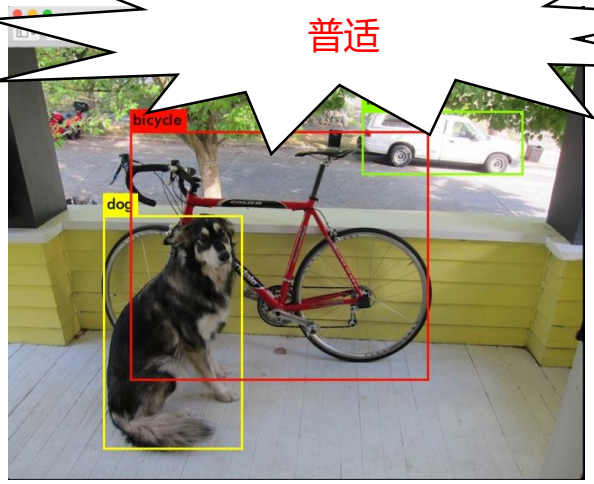
- 什么是目标检测系统？（YOLOv3）



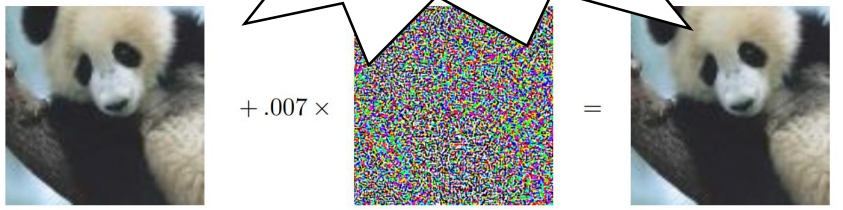
背景

- 欺骗目标检测系统和传统对抗样本工作区别?

普适



特定



$\boldsymbol{x}$

"panda"
57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

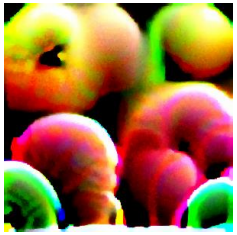
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

动机

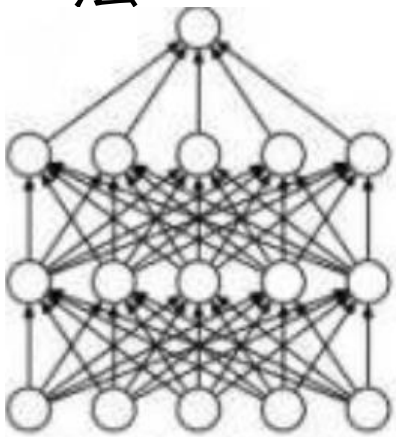
- 欺骗YOLOv3，让目标（人）消失

恶意的patch，如何产生？

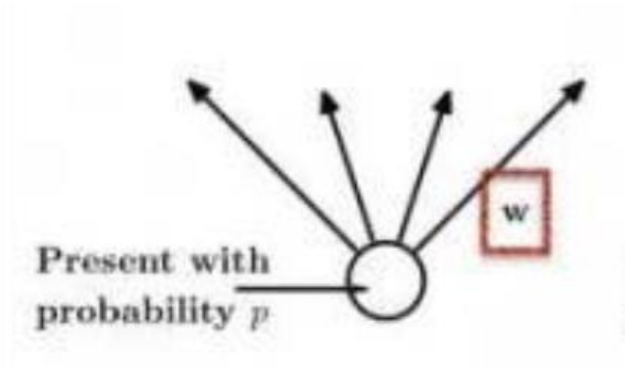


先验知识

- YOLOv3开源，白盒场景，考虑用针对白盒的算法②模仿类似模型训练过程：



神经网络

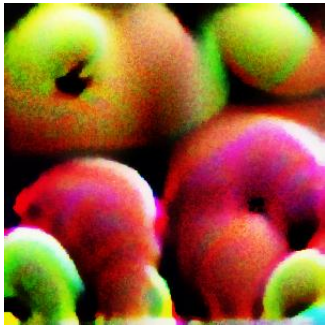


内部参数

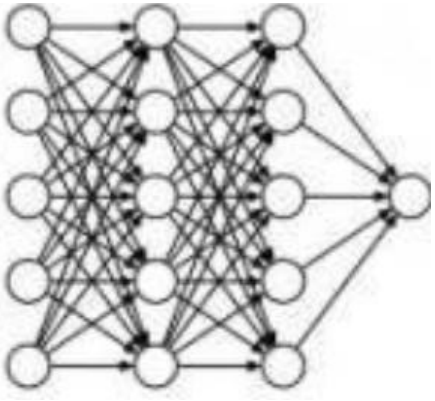
损失函数：
loss

先验知识

- YOLOv3开源，白盒场景，考虑用针对白盒的算法优化过程：



图片



训练好的神经网络

优化目标

损失函数：loss

如何制定优化目标

物体所处的位置

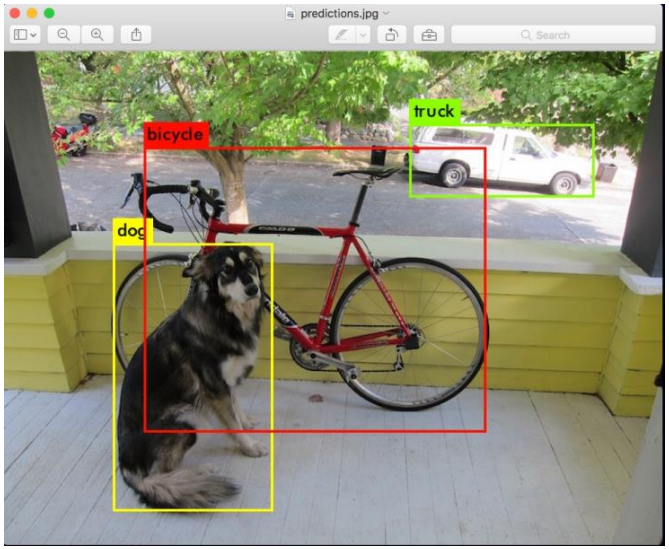
- YOLOv3的输出=置信度+预测类别概率分布+位置信息

这里是否存在物体?

这里什么物体?

置信度 > 阈值 ?

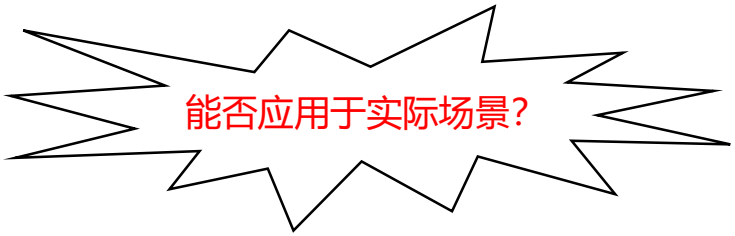
Loss=置信度



如何制定优化目标

- 损失函数:

Loss=置信度



如何制定优化目标

- 损失函数:

Loss=置信度+颜色损失

颜色损失项：$$L_{nps} = \sum_{p_{\text{patch}} \in p} \min_{c_{\text{print}} \in C} |p_{\text{patch}} - c_{\text{print}}|$$

如何制定优化目标

- 损失函数:

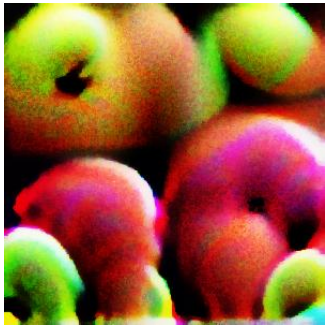
Loss=置信度+颜色损失项+平滑项

颜色损失项： $L_{nps} = \sum_{p_{\text{patch}} \in p} \min_{c_{\text{print}} \in C} |p_{\text{patch}} - c_{\text{print}}|$ $\boldsymbol{C_{\text{print}}}$：可打印像素值

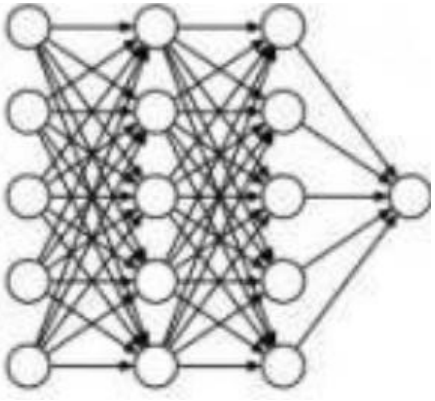
平滑项： $L_{tv} = \sum_{i,j} \sqrt{((p_{i,j} - p_{i+1,j})^2 + (p_{i,j} - p_{i,j+1})^2}$

开始训练

- 在loss的指引下，开始针对打底图片进行训练



打底图片



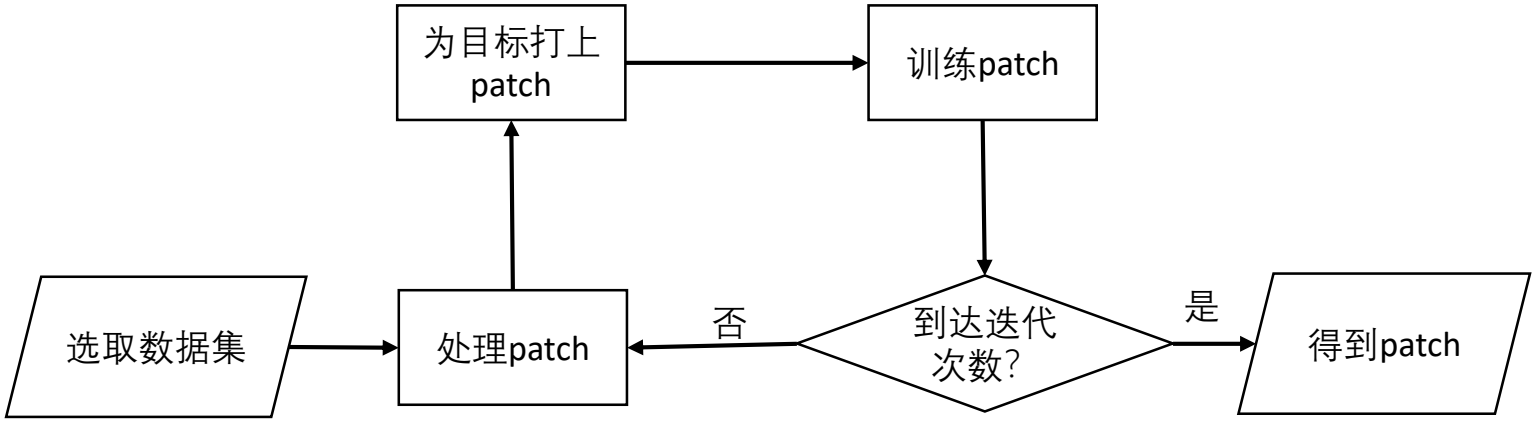
训练好的神经网络

优化目标

损失函数：loss

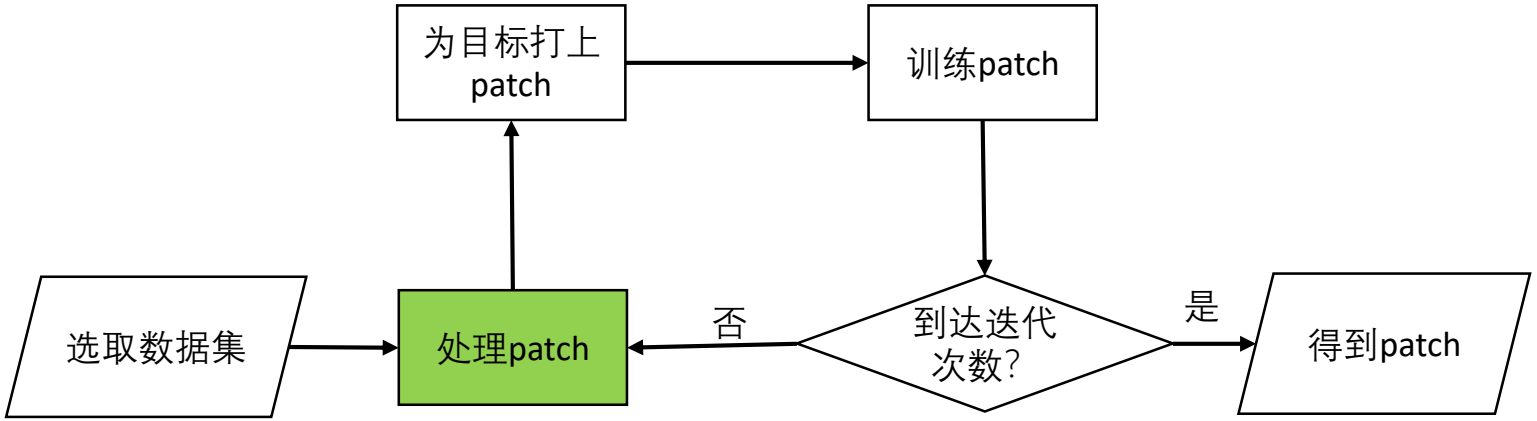
如何训练？

- 训练过程:



处理patch

- 训练过程:

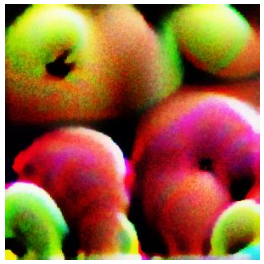


处理patch

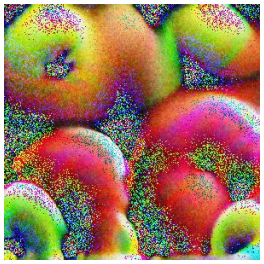
- Patch需要增强鲁棒性:

加噪声、旋转、改变亮度、对比度

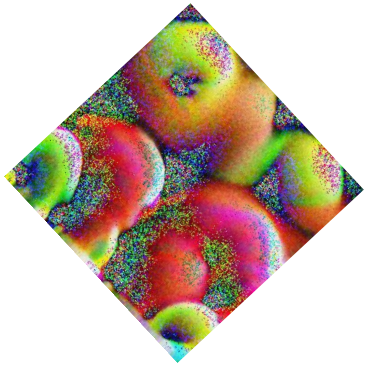
底图应该很重要!



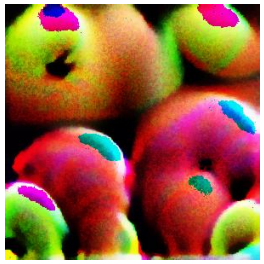
原始底图



加噪声



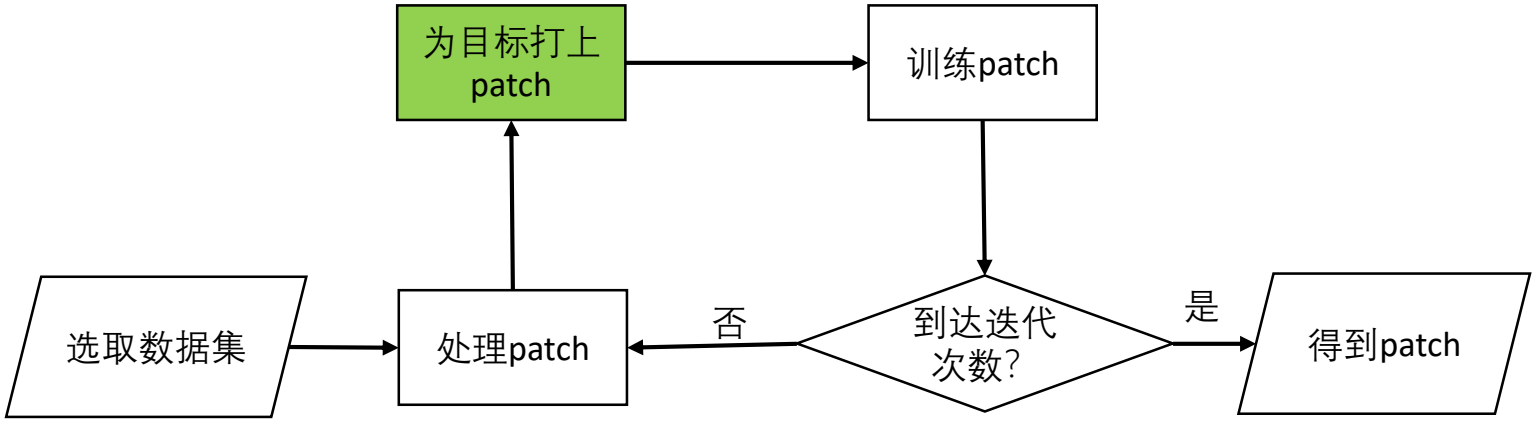
旋转



调整亮度

打patch

- 训练过程:

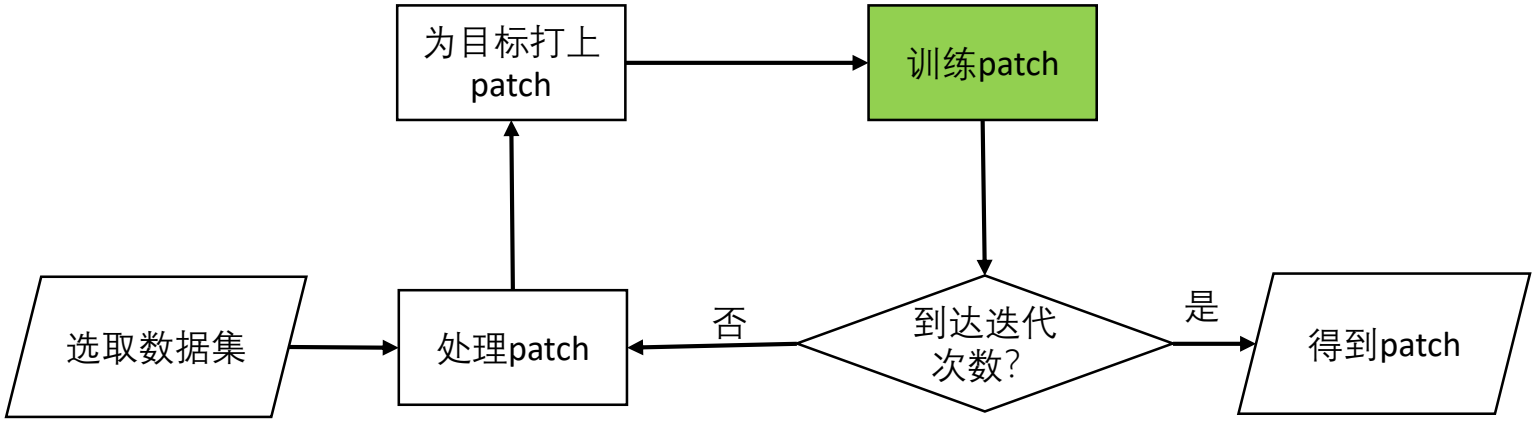


为目标打patch

- 需要把patch添加到图片中的人

训练patch

- 训练过程:



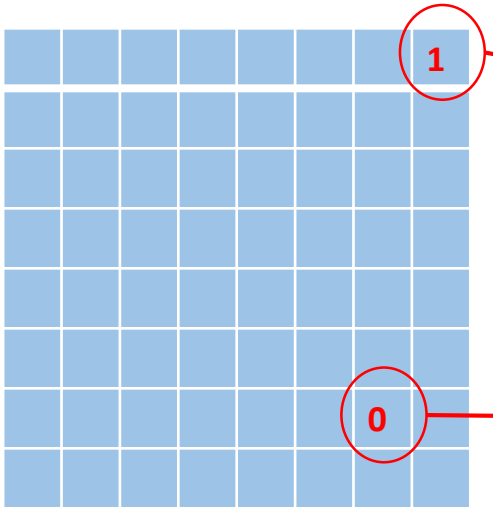
训练patch

- 基于损失函数，使用合适的优化器，进行训练；

训练能够持续进行吗？

训练patch

- 基于损失函数，使用合适的优化器，进行训练；
- 图片像素经预处理后，像素值的值域为[0,1]：



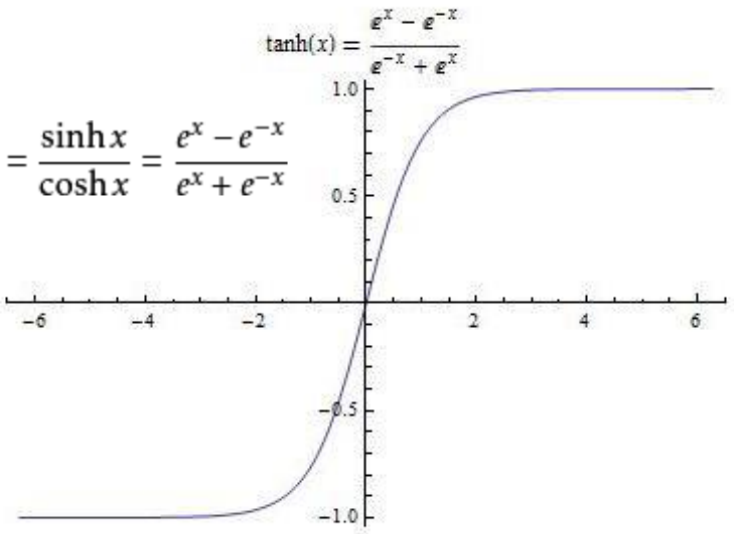
当前像素值1，梯度指引“我”继续增加，但我还能增加吗？

当前像素值0，梯度指引“我”继续减少，但我还能减少吗？

训练patch

- 转换优化对象到另一值域:

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



$$\frac{1}{2}(\tanh(x_{\text{orig}} + \text{扰动}) + 1)$$

新的优化对象

将原图转换到tanhx域即可得到：x_{orig}

结果

谢谢

Q&A