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

Suya F, Chi J, Evans D, et al

University of Virginia

USENIX, 2020

总述

- 动机：减少黑盒攻击的查询次数
- 背景：现有的黑盒攻击方法可以分为两类：
  - 迁移攻击：训练本地替代模型——查询次数少，但有迁移损失
  - 优化攻击：将攻击目标变成优化问题——攻击成功率高，但查询次数多
- 成果：
  - 结合两类攻击方法，查询次数少，攻击成功率高
  - 提出批攻击

假设

- 本地的对抗样本相较于原图片，是更好的优化攻击起点
  - 相同的任务在不同的模型中拥有相似的决策边界

- 优化攻击学到的标签可以用于本地模型微调
  - 跨过检测边界的样本可用于训练模型

算法流程

- 输入： 图片、本地模型、受害者模型 输出： 对抗样本
- 先本地找到对抗样本(line 8)
- 若不成功， 利用这些样本进行优化攻击(line 9)
- 利用查询结果对本地模型进行微调(line 13-15)

input : Set of seed images $\mathbf{X}$ with labels,
local model ensemble F,
target black-box model g

output : Set of successful adversarial examples

1 $\mathbf{R} \leftarrow \mathbf{X}$ *(remaining seeds to attack)*
2 $A \leftarrow \emptyset$ *(successful adversarial examples)*
3 $\mathbf{Q} \leftarrow \mathbf{X}$ *(fine-tuning set for local models)*
4 **while** $\mathbf{R}$ *is not empty* **do**
5   select and remove the next seed to attack
6     $\mathbf{x} \leftarrow selectSeed(\mathbf{R}, F)$
7   use local models to find a candidate adversarial example
8     $\mathbf{x}' \leftarrow whiteBoxAttack(F, \mathbf{x})$
9     $\mathbf{x}^\star, S \leftarrow blackBoxAttack(\mathbf{x}, \mathbf{x}', g)$ x'作为候选的起点，x用于控制对抗搜索空
S是输入的对应标签
10   **if** $\mathbf{x}^\star$ **then**
11         $A.insert(<\mathbf{x}, \mathbf{x}^\star>)$
12   **end**
13     $\mathbf{Q}.insert(S)$
14   use byproduct labels to retrain local models
15     $tuneModels(F, \mathbf{Q})$
16 **end**
17 **return** A

批攻击

- 动机：减少查询次数（查询次数过多会引起注意）
- 第一阶段：找到最易迁移的样本
  - 排序：成功攻击本地模型的个数；PGD步数
- 第二阶段：寻找优化攻击的候选
  - 排序：损失函数值（f为预测分数）

$$l(\mathbf{x}, t) = (\max_{i \neq t} \log f(\mathbf{x})_i - \log f(\mathbf{x})_t)^+$$

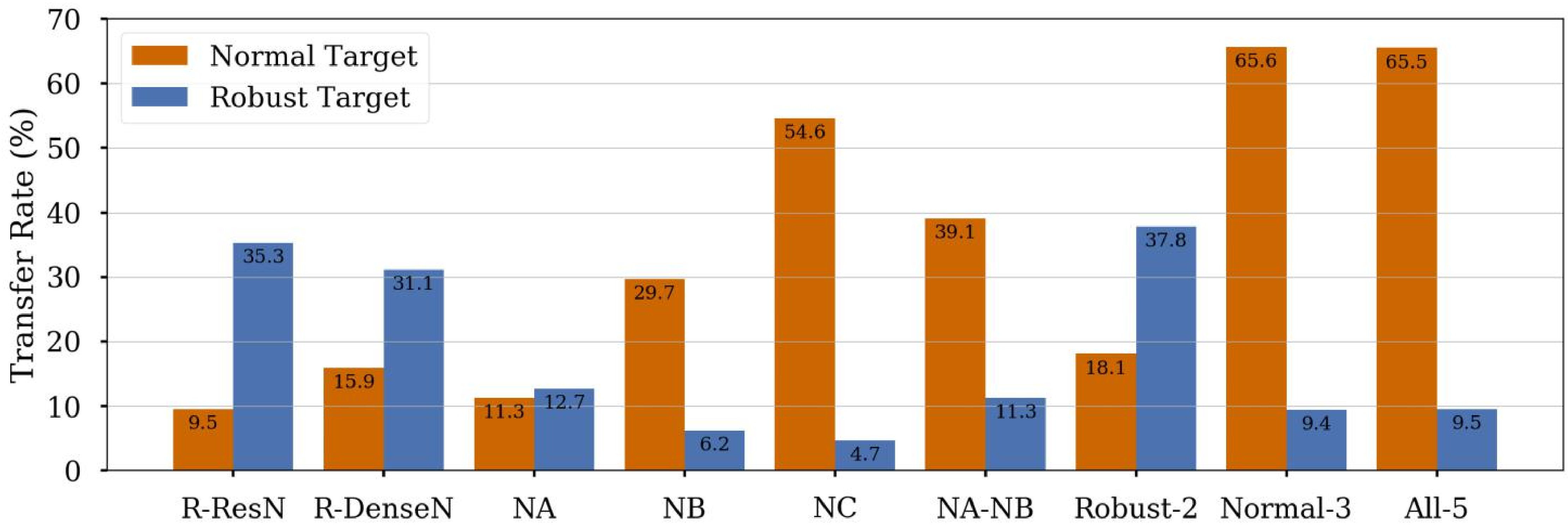
实验设置

- 数据集：MNIST、CIFAR10、ImageNet
- 本地模型/受害者模型：普通模型、稳健模型
  - 迁移攻击：PGD
  - 优化攻击：NES/AutoZOOM
  - 攻击类型：受害者模型为普通模型则目标攻击；稳健则非目标
- 基线：优化攻击：NES/AutoZOOM
- 评估指标：迁移率、成功率、查询次数

实验结果

Dataset	Target Model	Transfer Rate (%)	Gradient Attack	Success (%)		Queries/Seed		Queries/AE		Queries/Search	
				Base	*Ours*	*Base*	*Ours*	*Base*	*Ours*	*Base*	*Ours*
MNIST	Normal (T)	62.8	AutoZOOM	91.3	**98.9**	1,471	**279**	1,610	**282**	3,248	**770**
			NES	77.5	**89.2**	2,544	**892**	3,284	**1,000**	8,254	**3,376**
	Robust (U)	3.1	AutoZOOM	7.5	**7.5**	3,755	**3,748**	50,102	**49,776**	**83,042**	83,806
			NES	4.7	**5.5**	3,901	**3,817**	83,881	**69,275**	164,302	**160,625**
CIFAR10	Normal (T)	63.6	AutoZOOM	92.9	**98.2**	1,117	**271**	1,203	**276**	2,143	**781**
			NES	98.8	**99.8**	1,078	**339**	1,091	**340**	1,632	**934**
	Robust (U)	10.1	AutoZOOM	64.3	**65.3**	1,692	**1,652**	2,632	**2,532**	3,117	**2,997**
			NES	**38.1**	38.0	2,808	**2,779**	7,371	**7,317**	**9,932**	9,977
ImageNet	Normal (T)	3.4	AutoZOOM	95.4	**98.0**	42,310	**29,484**	44,354	**30,089**	45,166	**31,174**
			NES	100.0	**100.0**	18,797	**14,430**	18,797	**14,430**	19,030	**14,939**

实验结果



不同集成的迁移率

实验结果

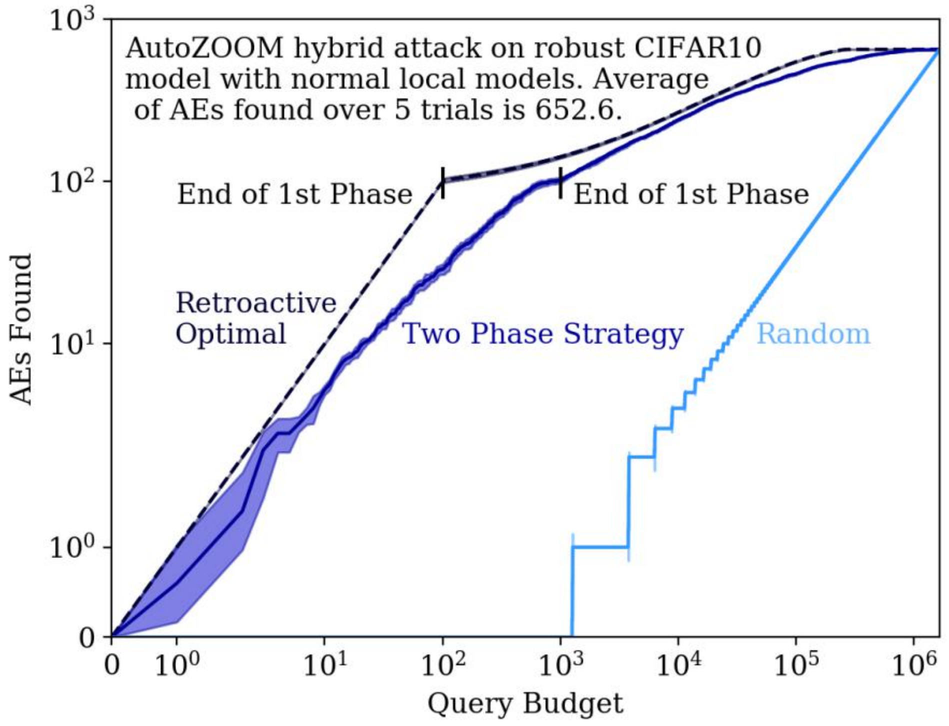
Model	Gradient Attack	Queries/AE		Success Rate (%)		Transfer Rate (%)	
		Static	*Tuned*	*Static*	*Tuned*	*Static*	*Tuned*
MNIST Normal (T)	AutoZOOM	282	**194**	98.9	**99.5**	60.6	**74.7**
	NES	1,000	**671**	89.2	**92.2**	60.6	**76.9**
MNIST Robust (U)	AutoZOOM	49,776	**42,755**	7.5	**8.6**	3.4	**5.1**
	NES	69,275	**51,429**	5.5	**7.3**	3.4	**4.8**
CIFAR10 Normal (T)	AutoZOOM	**276**	459	**98.2**	96.3	**65.6**	19.7
	NES	**340**	427	**99.8**	99.6	**65.6**	40.7
CIFAR10 Robust (U)	AutoZOOM	**2,532**	2,564	**65.3**	64.9	9.4	**10.1**
	NES	7,317	**7,303**	**38.0**	37.6	9.4	**10.7**

微调的影响

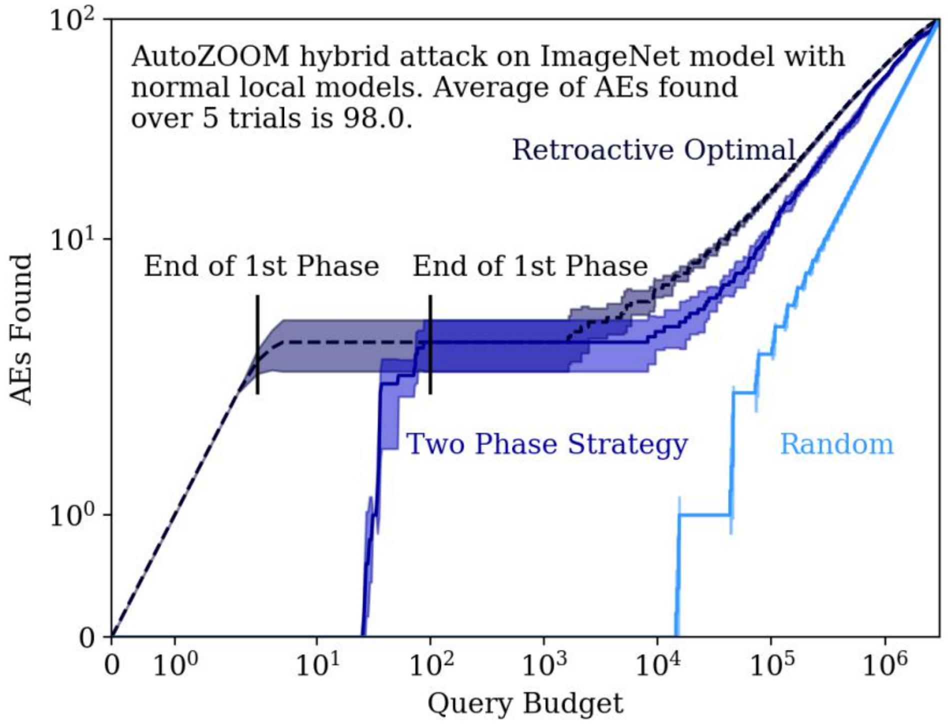
实验设置

- 数据集：CIFAR10、ImageNet
- 基线：最优化（最好效果）、随机（最坏效果）
- 评估指标：查询次数

实验结果



(a) Target: Robust CIFAR10 Model, Local Ensemble: Normal-3



(b) Target: Standard ImageNet Model

批攻击的效果

实验结果

Target Model	Prioritization Method	Top 1%	Top 2%	Top 5%	Top 10%
Robust	Retroactive Optimal	10.0 ± 0.0	20.0 ± 0.0	50.0 ± 0.0	107.8 ± 17.4
CIFAR10	Two-Phase Strategy	20.4 ± 2.1	54.2 ± 5.6	218.2 ± 28.2	826.2 ± 226.6
(1000 Seeds)	Random	$24,054 \pm 132$	$49,372 \pm 270$	$125,327 \pm 686$	$251,917 \pm 137$
Standard	Retroactive Optimal	1.0 ± 0.0	2.0 ± 0.0	$3,992 \pm 3,614$	$34,949 \pm 3,742$
ImageNet	Two-Phase Strategy	28.0 ± 2.0	38.6 ± 7.5	$18,351 \pm 13,175$	$78,844 \pm 11,837$
(100 Seeds)	Random	$15,046 \pm 423$	$45,136 \pm 1,270$	$135,406 \pm 3,811$	$285,855 \pm 8045$

批攻击的效果

缺点

- 对于不同的数据，微调的效果可能不如未进行微调的效果
  - 可能假设存在缺陷
- 对于不同的目标模型，本地模型的选用(Normal/Robust)会影响效果
- 没有尝试与优化攻击中的与梯度无关的方法结合

Gotta Catch'Em All: Using Honeypots to Catch Adversarial Attacks on Neural Networks

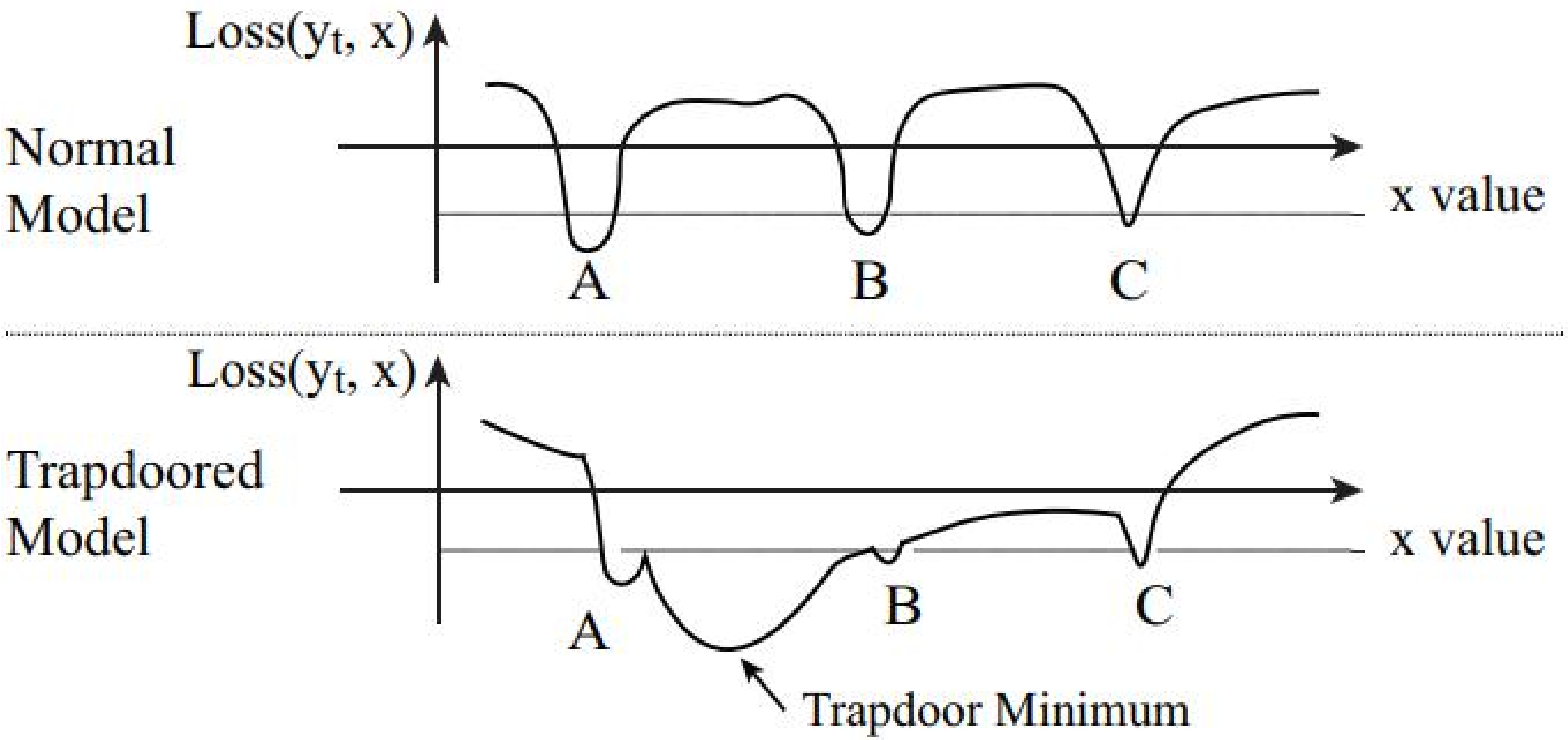
Shan S, Wenger E, Wang B, et al

University of Chicago

CCS, 2020

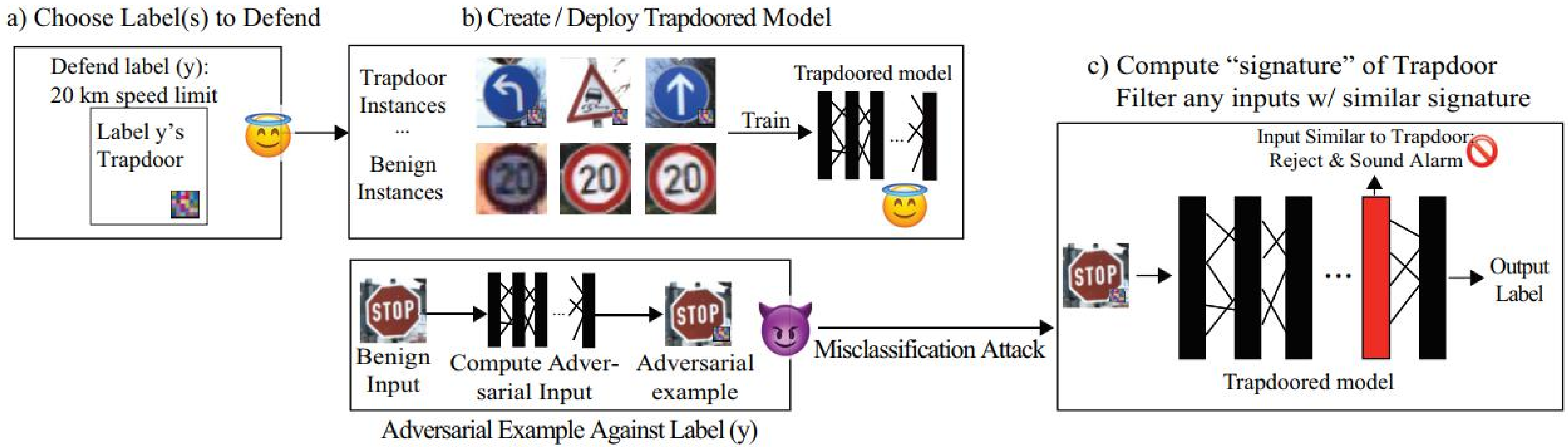
思路

- 利用蜜罐进行防御



Loss(y_t, x)
Normal
Model
x value
A
B
C
Loss(y_t, x)
Trapdoored
Model
x value
A
B
C
Trapdoor Minimum

防御流程



a) Choose Label(s) to Defend
Defend label (y):
20 km speed limit
Label y's
Trapdoor
b) Create / Deploy Trapdoored Model
Trapdoor
Instances
...
Benign
Instances
Trapdoored model
Train
Benign
Input
Compute Adver-
sarial Input
Adversarial
example
Adversarial Example Against Label (y)
Misclassification Attack
c) Compute "signature" of Trapdoor
Filter any inputs w/ similar signature
Input Similar to Trapdoor:
Reject & Sound Alarm
Output
Label
Trapdoored model

防御流程（单标签）

- 构建嵌入陷阱的训练数据集

$$x'_{i,j,c} = (1 - m_{i,j,c}) \cdot x_{i,j,c} + m_{i,j,c} \cdot \delta_{i,j,c}$$

- 训练模型

$$\min_{\theta} \quad \ell(y, \mathcal{F}_\theta(x)) + \lambda \cdot \ell(y_t, \mathcal{F}_\theta(x + \Delta))$$

  - 记录陷阱符号（下式；E为期望，g为特征表达，为softmax前的激活向量）

$$S_\Delta = \mathbf{E}_{x \in \mathcal{X}, y_t \neq \mathcal{F}_\theta(x)} g(x + \Delta),$$

- 检测对抗攻击
  - 计算S和g(x+Ɛ)的余弦相似度，是否超出阈值

可视化

(a) Single Label Defense Trapdoor

(b) All Label Defense Trapdoor

实验设置

- 数据集（分类）：MNIST、GTSRB、CIFAR10、YouTube Face
- 攻击方法：CW、Elastic Net、PGD、BPDA、SPSA、FGSM
- 基线：Feature Squeeze、MagNet、LID
- 指标：假阳性率、对抗样本检测率

实验结果

Table 1: Adversarial detection success rate when defending a single label at 5% FPR, averaged across all the labels.

Model	CW	ElasticNet	PGD	BPDA	SPSA	FGSM
MNIST	95.0%	96.7%	100%	100%	100%	100%
GTSRB	96.3%	100%	100%	100%	93.8%	100%
CIFAR10	100%	97.0%	100%	100%	100%	96.4%
YouTube Face	97.5%	98.8%	100%	100%	96.8%	97.0%

实验结果

Table 3: Comparing detection success rate of Feature Squeezing (FS), LID, and Trapdoor when defending all labels.

Model	Detector	FPR	CW	EN	PGD	BPDA	SPSA	FGSM	Avg Succ.
MNIST	FS	5%	99%	100%	94%	96%	94%	98%	97%
	MagNet	5.7%	83%	87%	100%	97%	96%	100%	94%
	LID	5%	89%	86%	96%	86%	98%	95%	92%
	Trapdoor	5%	97%	98%	100%	100%	100%	94%	**98%**
GTSRB	FS	5%	100%	99%	71%	73%	94%	45%	90%
	MagNet	4.7%	90%	89%	100%	100%	92%	100%	95%
	LID	5%	91%	81%	100%	67%	100%	100%	90%
	Trapdoor	5%	96%	97%	98%	98%	97%	98%	**97%**
CIFAR10	FS	5%	100%	100%	69%	66%	97%	33%	78%
	MagNet	7.4%	88%	82%	95%	96%	94%	100%	93%
	LID	5%	90%	88%	95%	79%	96%	92%	90%
	Trapdoor	5%	94%	94%	100%	99%	100%	97%	**97%**
YouTube Face	FS	5%	100%	100%	66%	59%	88%	68%	80%
	MagNet	7.9%	89%	91%	98%	97%	98%	96%	95%
	LID	5%	81%	79%	89%	72%	92%	96%	85%
	Trapdoor	5%	99%	98%	100%	97%	96%	95%	**98%**

应对方法

- 剪除多余神经元（改变决策边界）：干净数据的准确率下降严重
- 找到正常标签和感染标签的区别：不能应对多标签感染
- 替代模型攻击：将陷阱引入替代模型
- 不学习陷阱（unlearning techniques）：从$F^{unlearn}$到F迁移性差
- 接触到干净模型：迁移性差

应对方法

- 攻击：
  - 找出trapdoors的边界，找到相应的对抗扰动
  - 利用强大的算力
- 防御：
  - 随机化神经元符号
  - 每个标签多个陷阱

优缺点

- 优点：
  - 有效抵御大部分攻击
- 缺点：
  - 训练次数增加
  - 实验部分没有验证其对干净样本的影响
  - 无法证明所有的攻击的方法都会利用trapdoor
  - 在应对方法部分：可以从概率分布入手