Data Poisoning: Attacks and Defenses

马兴军，复旦大学 计算机学院

Recap: week 6

1. Adversarial Defense

- Early Defense Methods
- Early Adversarial Training Methods
- Advanced Adversarial Training Methods
- Remaining Challenges and Recent Progress

Adversarial Attack Competition

RESULTS							
#	User	Entries	Date of Last Entry	Score ▲	Efficiency Score ▲	Error Rate ▲	Detailed Results
1	xbhuang	7	10/17/23	0.5798 (1)	0.9528 (4)	0.4866 (7)	View
2	liujiahao	16	10/17/23	0.5788 (2)	0.9406 (6)	0.4883 (5)	View
3	zyhhaha	5	10/17/23	0.5772 (3)	0.9307 (7)	0.4888 (4)	View
4	archen	4	10/13/23	0.5760 (4)	0.9406 (6)	0.4849 (9)	View
5	SiyuanTang	3	10/16/23	0.5756 (5)	0.9010 (9)	0.4942 (2)	View
6	jxzhou	15	10/17/23	0.5755 (6)	0.9307 (7)	0.4867 (6)	View
7	shuyang_jiang	5	10/15/23	0.5754 (7)	0.9010 (9)	0.4940 (3)	View
8	tdlhl	8	10/11/23	0.5754 (7)	0.9010 (9)	0.4940 (3)	View
9	siyuandu	15	10/15/23	0.5724 (8)	0.9406 (6)	0.4803 (10)	View
10	LiGuanyu	14	10/13/23	0.5671 (9)	0.9604 (3)	0.4688 (12)	View
11	Ysy1	1	10/17/23	0.5644 (10)	0.9010 (9)	0.4803 (10)	View
12	fudaner	2	10/16/23	0.5605 (11)	0.8095 (12)	0.4983 (1)	View

Link: https://codalab.lisn.upsaclay.fr/competitions/15669?secret_key=77cb8986-d5bd-4009-82f0-7dde2e819ff8

Data Poisoning: Attacks and Defenses

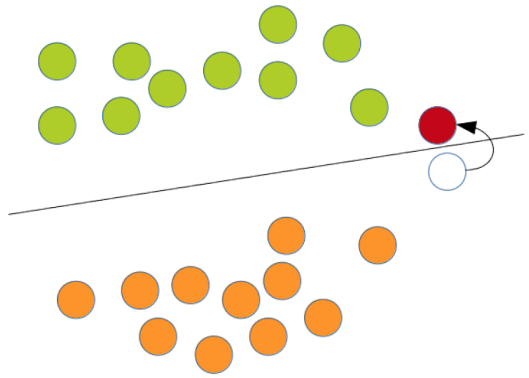
- A Brief History of Data Poisoning
- Data Poisoning Attacks
- Data Poisoning Defenses
- Poisoning for Data Protection
- Future Research

A Recap of the Attack Taxonomy

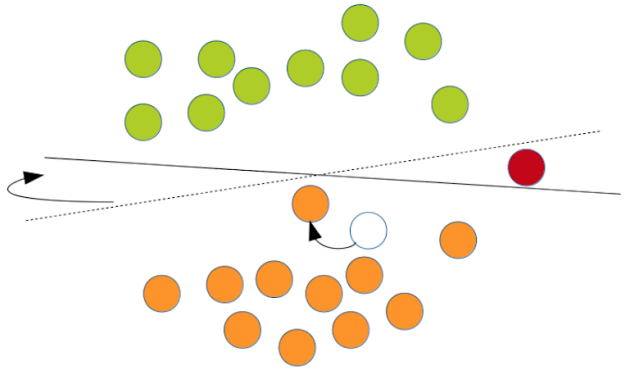
- Attack timing
  - **Poisoning attack**
  - Evasion attack
- Attacker's goal
  - Targeted attack
  - Untargeted attack
- Attacker's knowledge
  - Black-box
  - White-box
  - Gray-box
- Universality
  - Individual
  - Universal

Data Poisoning is Training Time Attack

- **Evasion (Causation) attack**
  - Test time attack
  - Change input example



- **Poisoning attack**
  - Training time attack
  - Change classification boundary



Data Poisoning: Attacks and Defenses

- A Brief History of Data Poisoning
- Data Poisoning Attacks
- Data Poisoning Defenses
- Poisoning for Data Protection
- Future Research

A Brief History: The Eearliest Work

Learning in the Presence of Malicious Errors

(extended abstract)

Michael Kearns
Harvard University

Ming Li
Harvard University

1 Introduction

We study a practical extension to the Valiant model of machine learning from examples [V84]: the presence of errors, possibly maliciously generated by an adversary, in the sample data. Recent papers have made progress in the Valiant model by providing algorithms for learning various classes of functions, by giving evidence for the intractability of learning other classes, and by developing general tools and techniques for determining learnability (see e.g. [BEHW86], [KLPV87], [R87]). These results assume an error-free oracle for examples of the function being learned. In many environments, however, there is always some chance that an erro-

generality by making no assumptions on the nature of the errors that occur. Thus, we study a "worst-case" model of errors, in which the errors are generated by an adversary whose goal is to foil the learning algorithm.

The study of learning from examples with malicious errors was initiated in [V85], where it is assumed that there is a fixed probability β $(0 \leq \beta < 1)$ of an error occuring independently on each request for an example, but the error is of an arbitrary nature — in particular, it may be chosen by an adversary with unbounded computational resources, and knowledge of the function being learned, the probability distribution on the examples, and the internal state of the learning algorithm.

Kearns and Li. "Learning in the presence of malicious errors", SIAM Journal on Computing, 1993

Poisoning Intrusion Detection System

The attack model

		Integrity	*Availability*
Causative:	*Targeted*	Permit a specific intrusion	Create sufficient errors to make system unusable for one person or service
	Indiscriminate	Permit at least one intrusion	Create sufficient errors to make learner unusable
Exploratory:	*Targeted*	Find a permitted intrusion from a small set of possibilities	Find a set of points misclassified by the learner
	Indiscriminate	Find a permitted intrusion	

Barreno, Marco, et al. “Can machine learning be secure?.” ASIACCS, 2006.

Poisoning Intrusion Detection System

The defense model

		Integrity	*Availability*
Causative:	*Targeted*	• Regularization • Randomization	• Regularization • Randomization
	Indiscriminate	• Regularization	• Regularization
Exploratory:	*Targeted*	• Information hiding • Randomization	• Information hiding
	Indiscriminate	• Information hiding	

Barreno, Marco, et al. “Can machine learning be secure?.” ASIACCS, 2006.

Subvert Your Spam Filter

fudan.edu.cn 密码通知。

您好,xingjunma

您的密码今天到期
请按照以下说明保留您的当前密码并更新您的帐户。

保持当前密码

fudan.edu.cn 密码通知。© 2022

Hello,

My name is Nick Coetzee.

I regret to inform you that LeadsTree.org will shut down Friday.

We have now made all our databases available to the public on our website at a one-time fee.

Visit us at LeadsTree.org
Email ID: 708601

Usenet dictionary attack:

- Add legitimate words into spam emails
- 1% poisoning can subvert a spam filter

Nelson, Blaine, et al. "Exploiting machine learning to subvert your spam filter." *LEET* 8.1 (2008): 9.

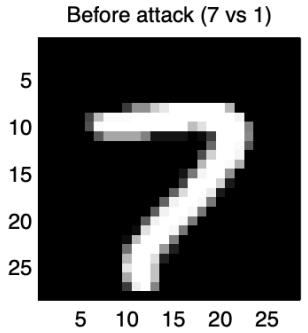
The Concept of Poisoning Attack

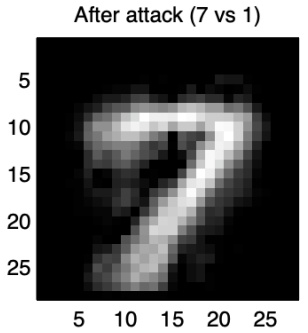
Algorithm 1 Poisoning attack against SVM

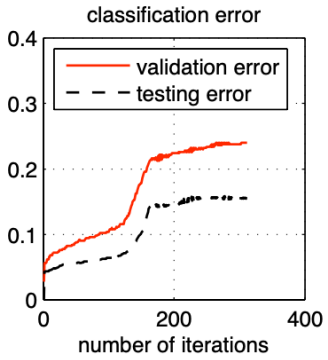
Input: $\mathcal{D}_{\text{tr}}$, the training data; $\mathcal{D}_{\text{val}}$, the validation data; y_c, the class label of the attack point; $x_c^{(0)}$, the initial attack point; t, the step size.

Output: x_c, the final attack point.

1: $\{\alpha_i, b\} \leftarrow$ learn an SVM on $\mathcal{D}_{\text{tr}}$.
2: $k \leftarrow 0$.
3: **repeat**
4: Re-compute the SVM solution on $\mathcal{D}_{\text{tr}} \cup \{x_c^{(p)}, y_c\}$ using incremental SVM (*e.g.*, Cauwenberghs & Poggio, 2001). This step requires $\{\alpha_i, b\}$.
5: Compute $\frac{\partial L}{\partial u}$ on $\mathcal{D}_{\text{val}}$ according to Eq. (10).
6: Set u to a unit vector aligned with $\frac{\partial L}{\partial u}$.
7: $k \leftarrow k + 1$ and $x_c^{(p)} \leftarrow x_c^{(p-1)} + tu$
8: **until** $L\left(x_c^{(p)}\right) - L\left(x_c^{(p-1)}\right) < \epsilon$
9: **return:** $x_c = x_c^{(p)}$







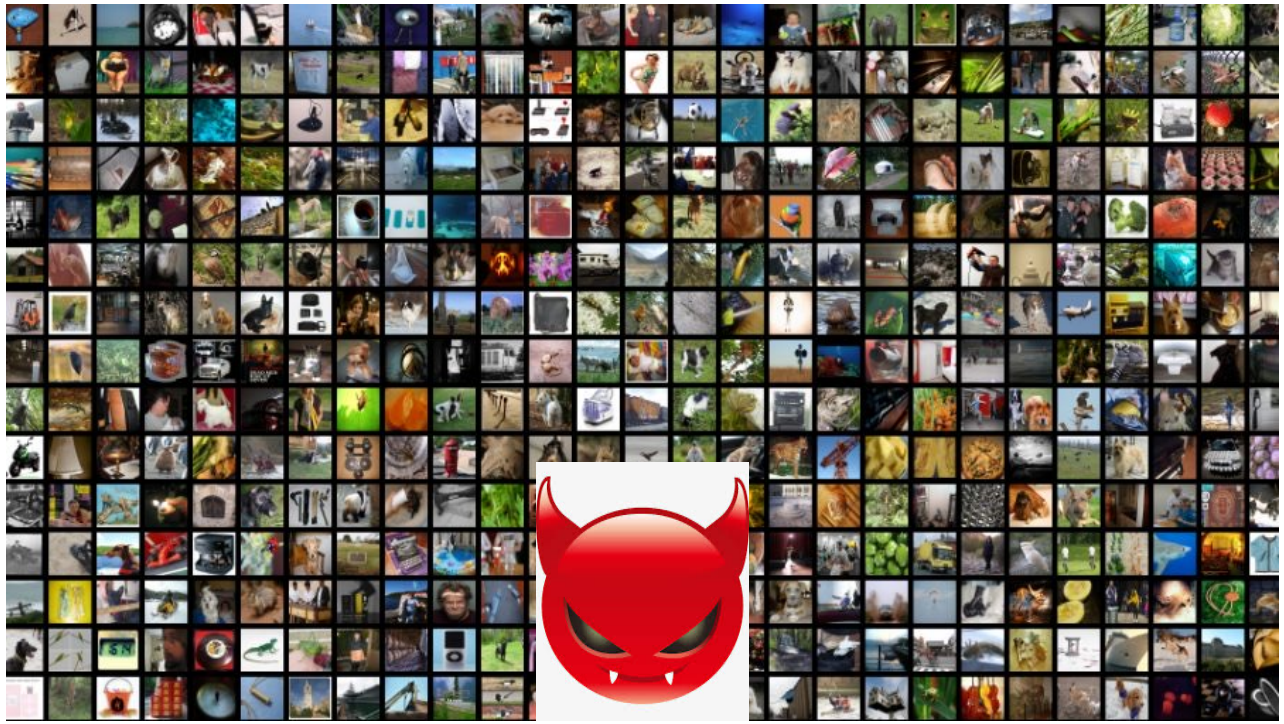
a single attack data point caused the classification error to rise from the initial error rates of 2–5% to 15–20%

Biggio, Nelson and Laskov. "Poisoning attacks against support vector machines."*arXiv:1206.6389* (2012).

Data Poisoning: Attacks and Defenses

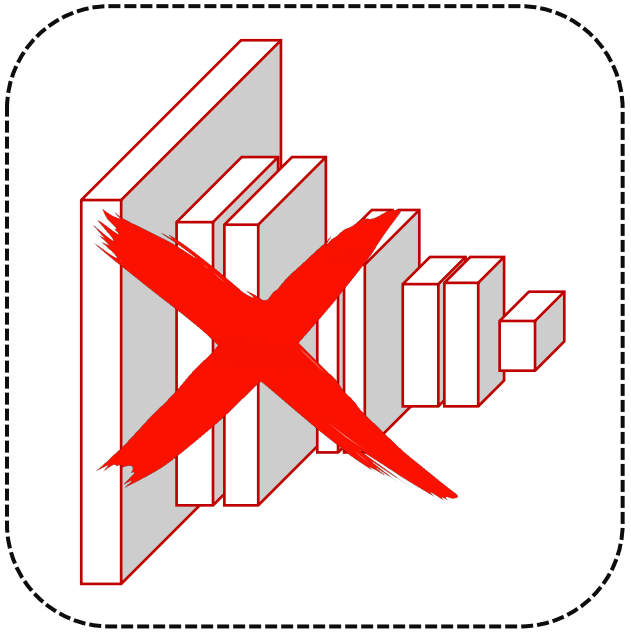
- A Brief History of Data Poisoning
- **Data Poisoning Attacks**
- Data Poisoning Defenses
- Poisoning for Data Protection
- Future Research

Attack Pipeline



污染少量训练样本（越少越好）

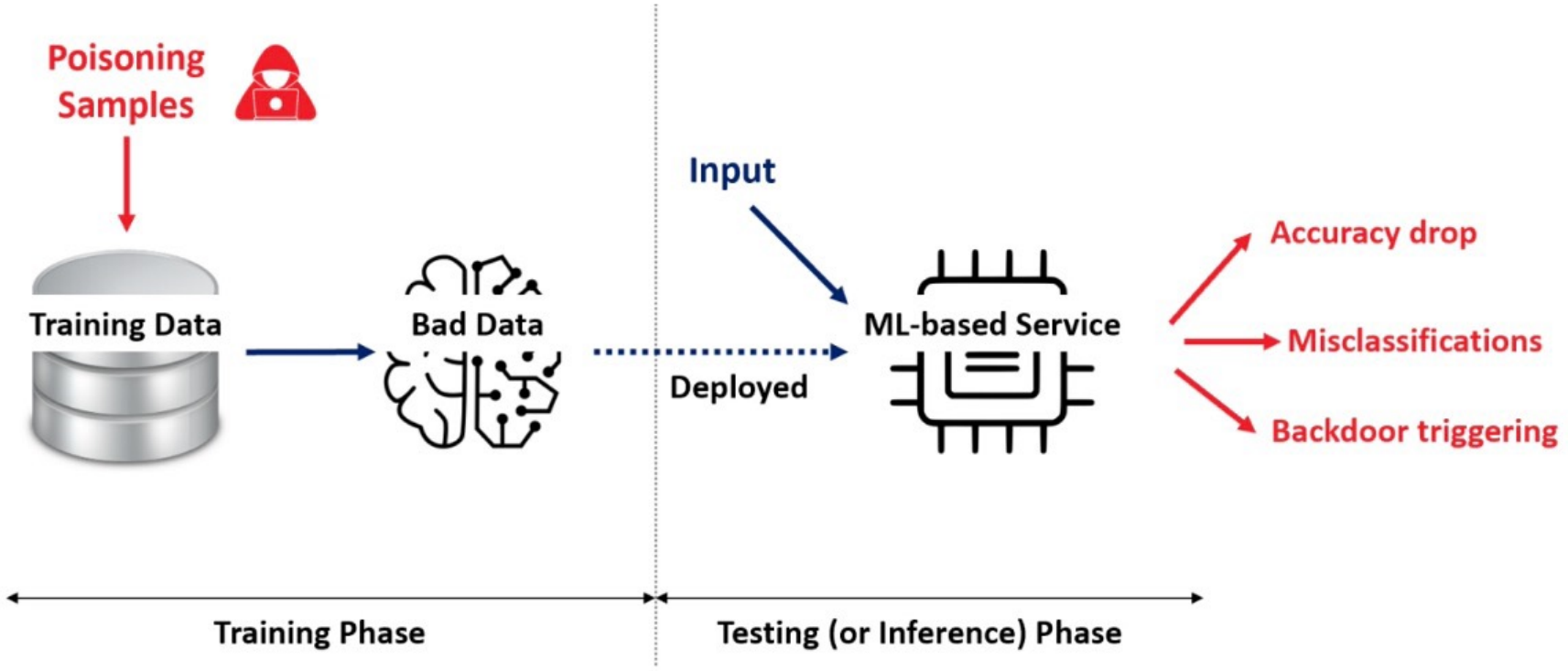
模型训练



无效模型、被控制模型

- 投毒攻击 != 后门攻击
- 后门攻击的一种实现方式是通过数据投毒

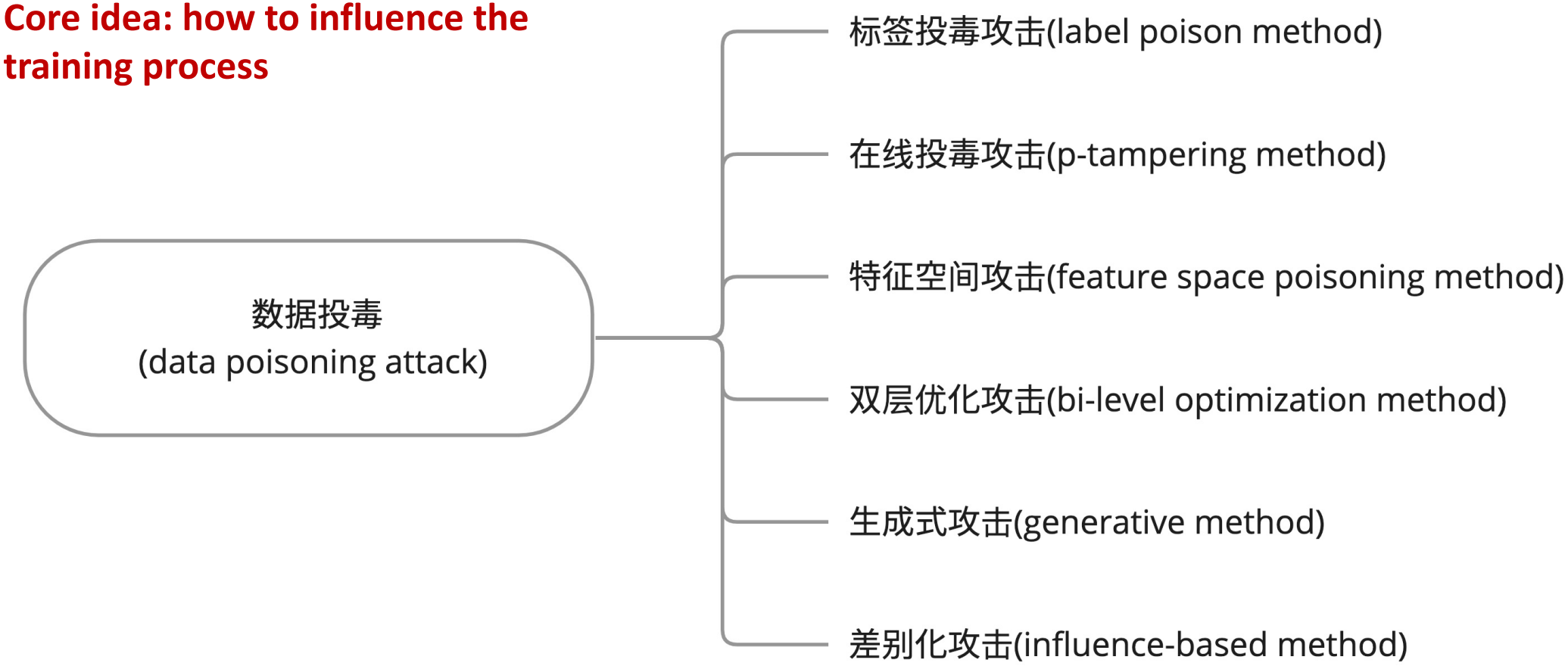
Attack Pipeline



Liu, Ximeng, et al. "Privacy and security issues in deep learning: A survey." *IEEE Access* 9 (2020): 4566-4593.

Attack Types

Core idea: how to influence the training process



Label Poisoning

Label Flipping Attack （“指鹿为马”攻击）

- 语音识别 f() = “[illegible]气不错”
- 人脸识别 f() = “小[illegible]”
- 语义分割 f() = 

- ☐ Random Labels
- ☐ Label Flipping
- ☐ Partial Label Flipping

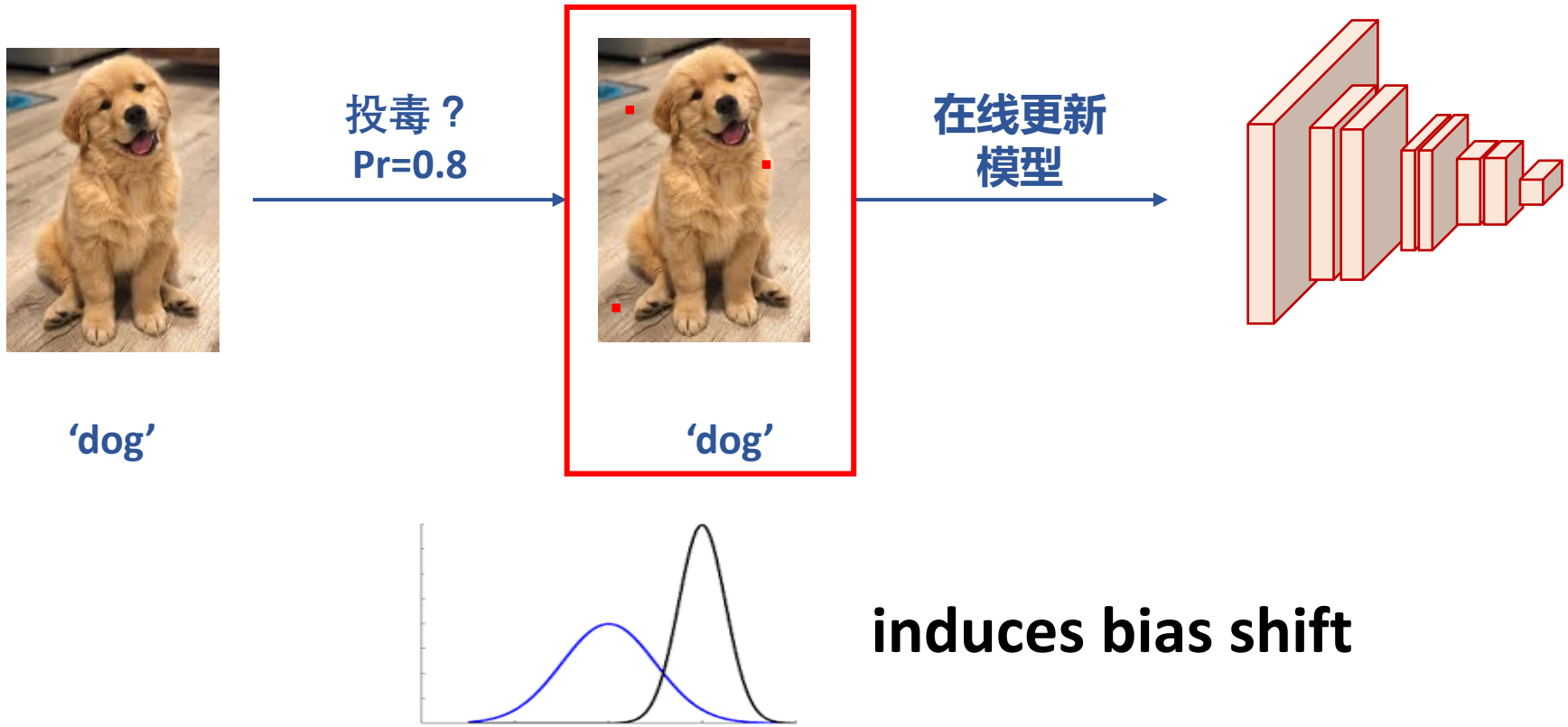
self-supervised learning?

Incorrect labels break supervised Learning!

Biggio, Nelson and Laskov. "Poisoning attacks against support vector machines."*arXiv:1206.6389* (2012).
Zhang and Zhu. "A game-theoretic analysis of label flipping attacks on distributed support vector machines." *CISS*, 2017.

p-tampering attacks

篡改攻击（“暗度陈仓”攻击）



induces bias shift

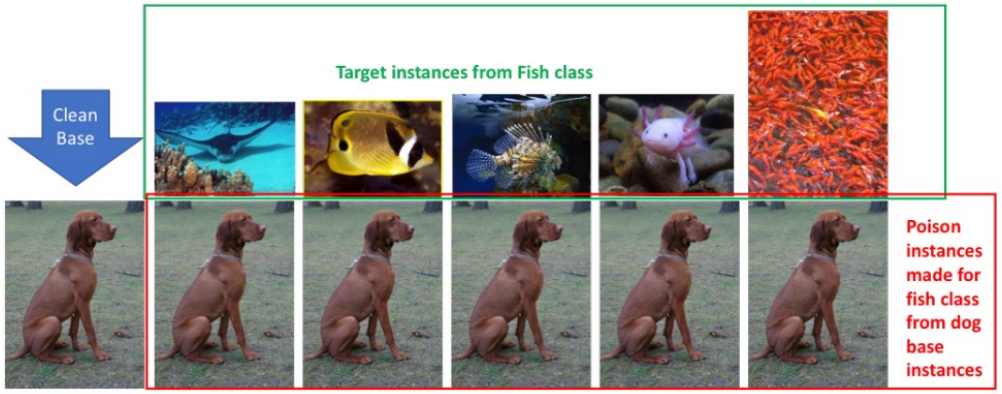
Mahloujifar and Mahmoody. “Blockwise p-tampering attacks on cryptographic primitives, extractors, and learners.” *TCC*, 20
Mahloujifar, Mahmoody and Mohammed. "Universal multi-party poisoning attacks." *ICML*, 2019.

Feature Space Poisoning

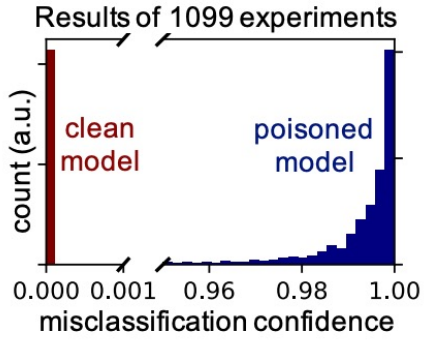
Feature Collision Attack （“声东击西”攻击）

- A **white-box** data poisoning method
- Feature flipping, does not change labels

$$\boldsymbol{x}_p = \arg\min ||f(\boldsymbol{x}_p) - f(\boldsymbol{x}_t)||_2^2 + \beta||\boldsymbol{x}_p - \boldsymbol{x}_b||_2^2$$



看上去是’狗’，但是在特征空间是’鱼’



优缺点：

- **需要知道目标模型**
- **对迁移学习很强**
- **对从头训练并不强**

Shafahi, Ali, et al. “Poison frogs! targeted clean-label poisoning attacks on neural networks.” NeurIPS 2018.

Convex Polytope Attack

凸多面体攻击（“四面楚歌”攻击）

- Improve the transferability to different DNNs
- 寻找一组毒化样本将目标样本包围在一个凸包内
- 借助多个预训练模型来寻找“包围”样本

$$\min_{\{c^{(i)}\},\{\boldsymbol{x}_p^{(j)}\}} \frac{1}{2m}\sum_{i=1}^{m}\frac{\left\|f^{(i)}(\boldsymbol{x}_t)-\sum_{j=1}^{k}c_j^{(i)}f^{(i)}(\boldsymbol{x}_p^{(j)})\right\|^2}{\|f^{(i)}(\boldsymbol{x}_t)\|^2}$$

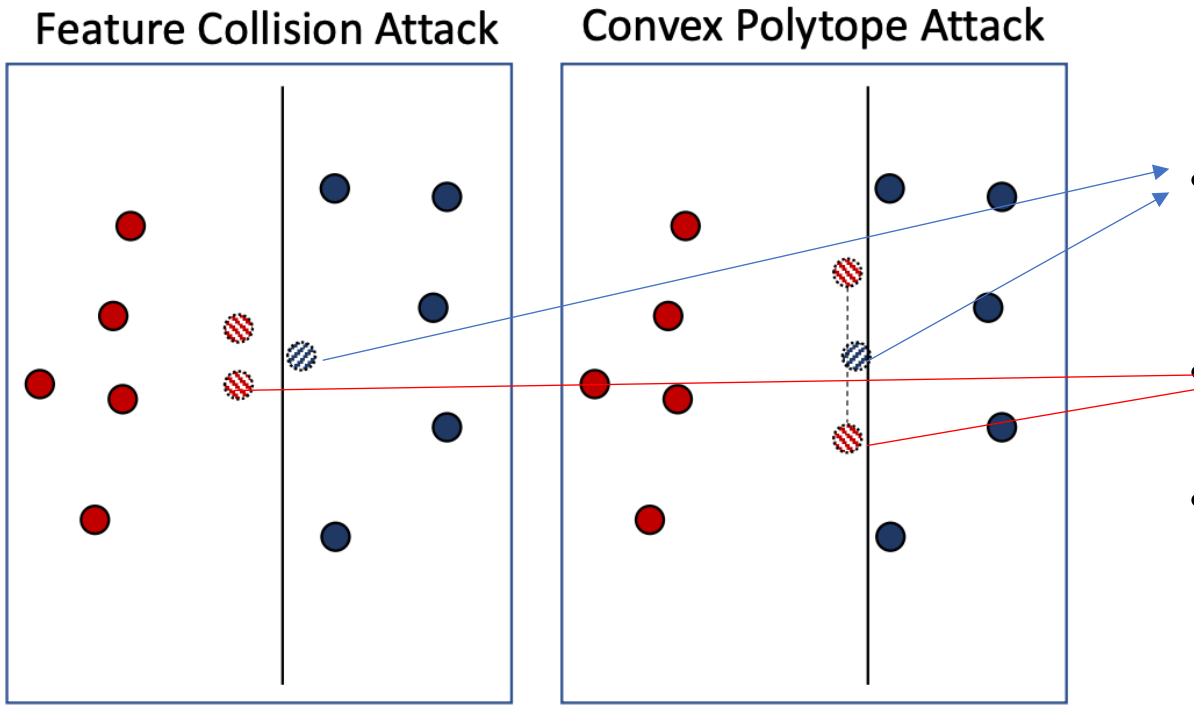
$$s.t.\ \sum_{j=1}^{k}c_j^{(i)}=1, c_j^{(i)}\geq 0, \forall i,j,\ \left\|\boldsymbol{x}_p^{(j)}-\boldsymbol{x}_b^{(j)}\right\|_\infty \leq \epsilon, \forall j$$

- $\{f^{(i)}\}_{i=1}^{m}$ ：m个预训练模型
- $\{\boldsymbol{x}_p^{(j)}\}_{j=1}^{k}$ ：针对 x_t 设计的k个“包围”样本
- $\sum_{j=1}^{k}c_j^{(i)}=1, c_j^{(i)}\geq 0$ 权重约束

Zhu, Chen, et al. “Transferable clean-label poisoning attacks on deep neural nets.” ICML 2019.

Convex Polytope Attack vs Feature Collision Attack

基于SVM的示例



- 蓝色**毛球**：目标样本，不存在与训练集中
- 红色**毛球**：包围样本
- 普通红/蓝点：普通训练样本

Zhu, Chen, et al. “Transferable clean-label poisoning attacks on deep neural nets.” ICML 2019.

Bi-level Optimization Attack

投毒攻击是一种“双层优化”：投毒完成后，训练模型才能知道其效果

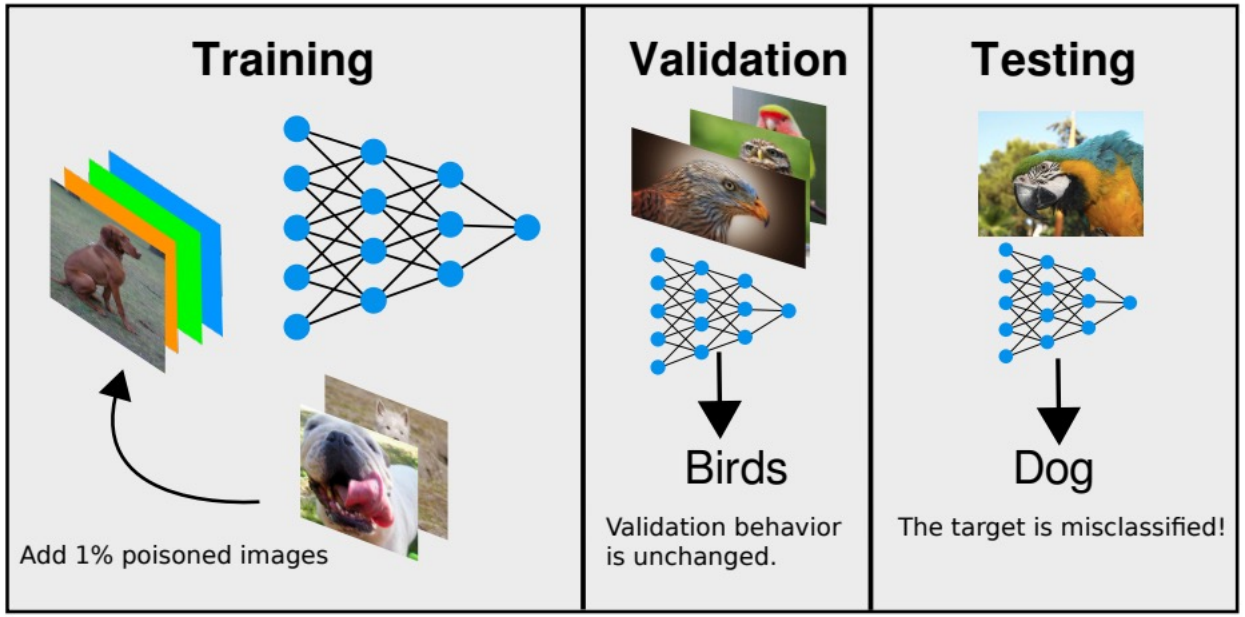
$$D_p{}' = \arg\max \mathcal{F}(D_p, \theta') = \mathcal{L}_1(D_{\text{val}}, \theta')$$
$$s.t. \quad \theta' = \arg\min \mathcal{L}_2(D \cup D_p, \theta)$$

- 是一个**最大-最小化（max-min）**问题
  - **内部最小化**：在投毒数据上更新模型
  - **外部最大化**：在更新后的模型上生成更强的投毒数据

Mei and Zhu. “Using machine teaching to identify optimal training-set attacks on machine learners.” AAAI 2015.

MetaPoison

One advanced bi-level optimization attack



- 不修改类标
- 有目标（Targeted攻击）
- 验证集上的性能不变
- 使用**元学习**寻找高效投毒样本
- 可攻击**微调**和**端到端**模型
- 成功**攻击商业模型**Google Cloud AutoML API

Huang, W. Ronny, et al. “Metapoison: Practical general-purpose clean-label data poisoning.” NeurIPS 2020.

MetaPoison

A Bi-level Min-Min Optimization Attack

$$D_p{}' = \arg\min \mathcal{F}(D_p, \theta') = \mathcal{L}_1(\{\boldsymbol{x}_t, y_{adv}\}, \theta')$$
$$s.t. \quad \theta' = \arg\min \mathcal{L}_2(D \cup D_p, \theta)$$

- **K-step 优化策略：**
  内层多步（’look ahead’），外层一步

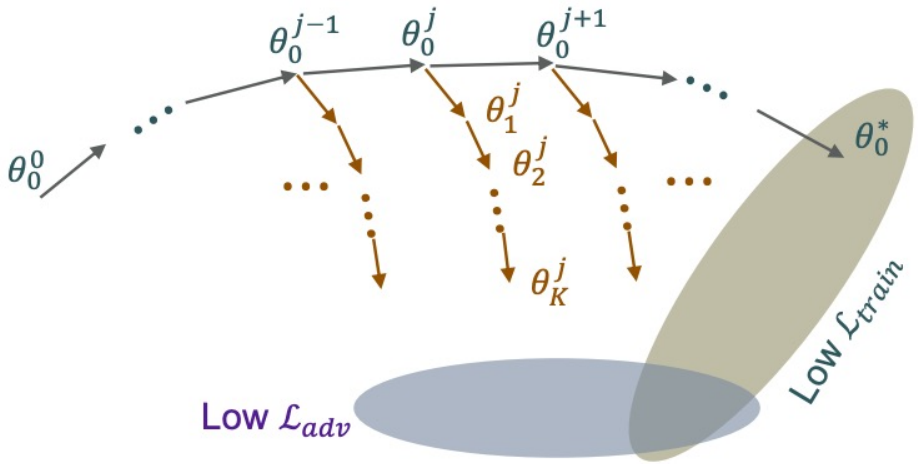
$$\theta_1 = \theta_0 - \alpha \nabla_\theta \mathcal{L}_{\text{train}}(X_c \cup X_p, Y; \theta_0)$$
$$\theta_2 = \theta_1 - \alpha \nabla_\theta \mathcal{L}_{\text{train}}(X_c \cup X_p, Y; \theta_1)$$
$$X_p^{i+1} = X_p^i - \beta \nabla_{X_p} \mathcal{L}_{\text{adv}}(x_t, y_{\text{adv}}; \theta_2),$$

- **使用m个模型和周期性初始化来增加探索**

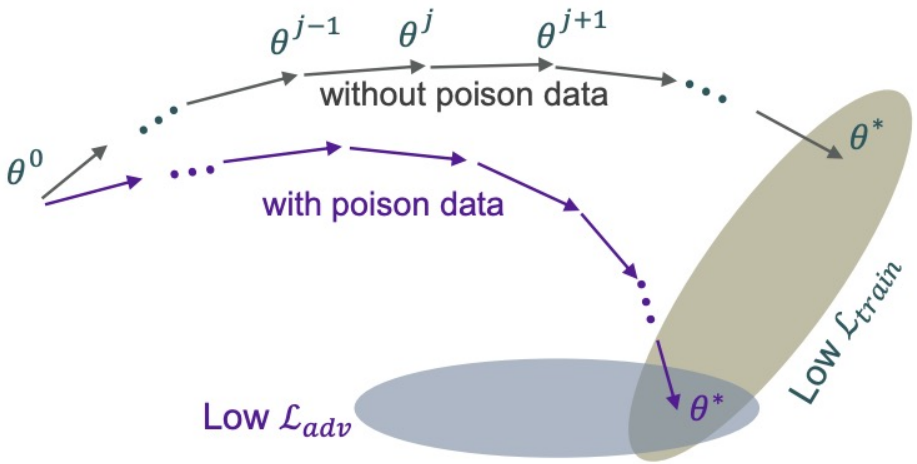
For $m = 1, \ldots, M$ models:
  Copy $\tilde{\theta} = \theta_m$
  For $k = 1, \ldots, K$ unroll steps[a]:
    $\tilde{\theta} = \tilde{\theta} - \alpha \nabla_{\tilde{\theta}} \mathcal{L}_{\text{train}}(X_c \cup X_p, Y; \tilde{\theta})$
  Store adversarial loss $\mathcal{L}_m = \mathcal{L}_{\text{adv}}(x_t, y_{\text{adv}}; \tilde{\theta})$
  Advance epoch $\theta_m = \theta_m - \alpha \nabla_{\theta_m} \mathcal{L}_{\text{train}}(X, Y; \theta_m)$
  If θ_m is at epoch $T + 1$:
    Reset θ_m to epoch 0 and reinitialize
Average adversarial losses $\mathcal{L}_{\text{adv}} = \sum_{m=1}^{M} \mathcal{L}_m / M$
Compute $\nabla_{X_p} \mathcal{L}_{\text{adv}}$

Huang, W. Ronny, et al. “Metapoison: Practical general-purpose clean-label data poisoning.” NeurIPS 2020.

MetaPoison



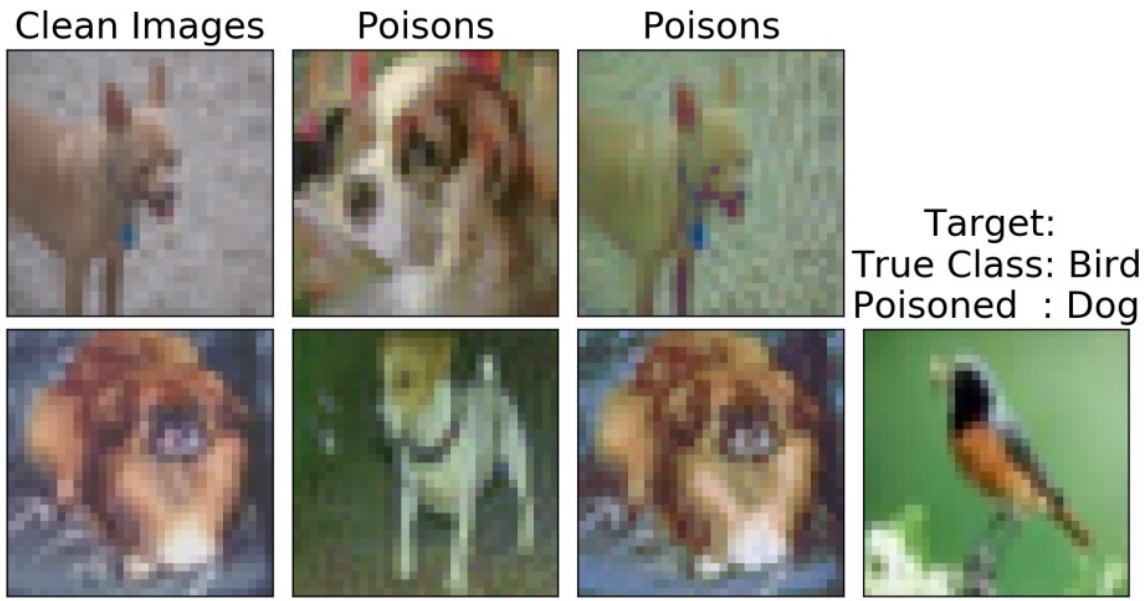
优化阶段

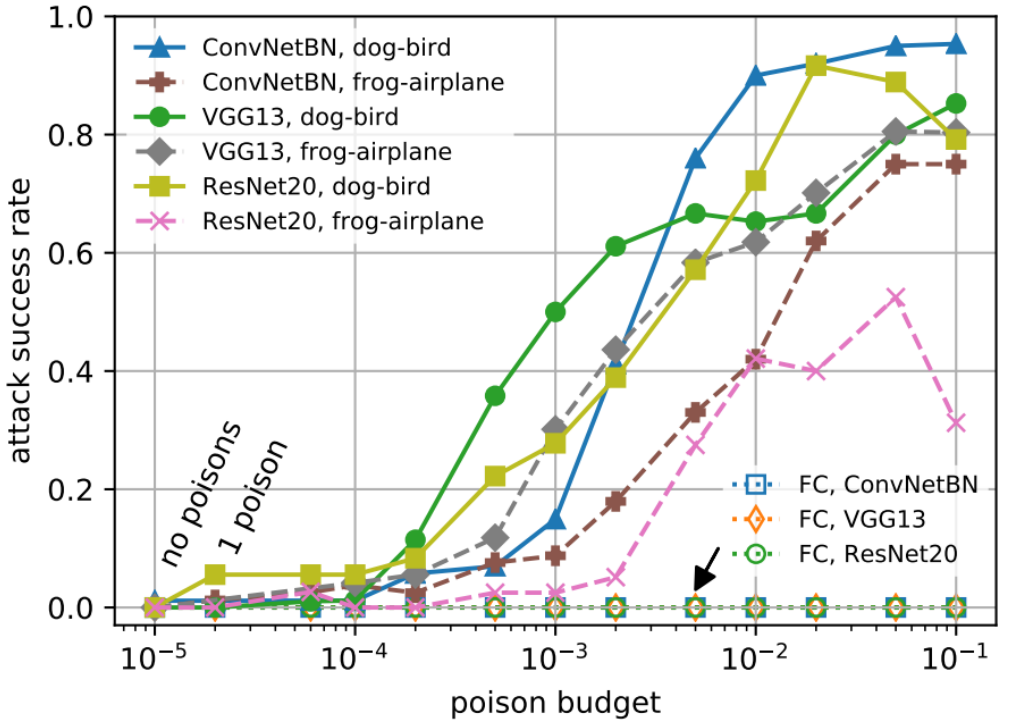


投毒后影响

Huang, W. Ronny, et al. “Metapoison: Practical general-purpose clean-label data poisoning.” NeurIPS 2020.

MetaPoison





示例：狗毒化成鸟

毒化0.1%的数据即可达到很高的ASR

Huang, W. Ronny, et al. “Metapoison: Practical general-purpose clean-label data poisoning.” NeurIPS 2020.

Witches‘ Brew：思想

依然是Min-Min 双层优化问题

$$\min_{x_p \in \mathcal{C}} \mathcal{L}_{\text{adv}}\left(x_t, \theta(x_p)\right) \quad \text{s.t.} \quad \theta(x_p) = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}_{\text{train}}(x_p^i, y_p^i, \theta).$$

□ Trick：在生成毒化样本时，使其梯度与目标样本一致

$$\nabla_\theta \mathcal{L}_{\text{adv}}(x_t, \theta^*) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \mathcal{L}_{\text{train}}(x_p^i, y_p^i, \theta^*)$$

直观理解：让毒化样本和目标样本在训练过程中**触发同样的梯度**，即让毒化样本更像目标样本

Geiping, Jonas, et al. “Witches‘ Brew: Industrial Scale Data Poisoning via Gradient Matching.” ICLR 2021.

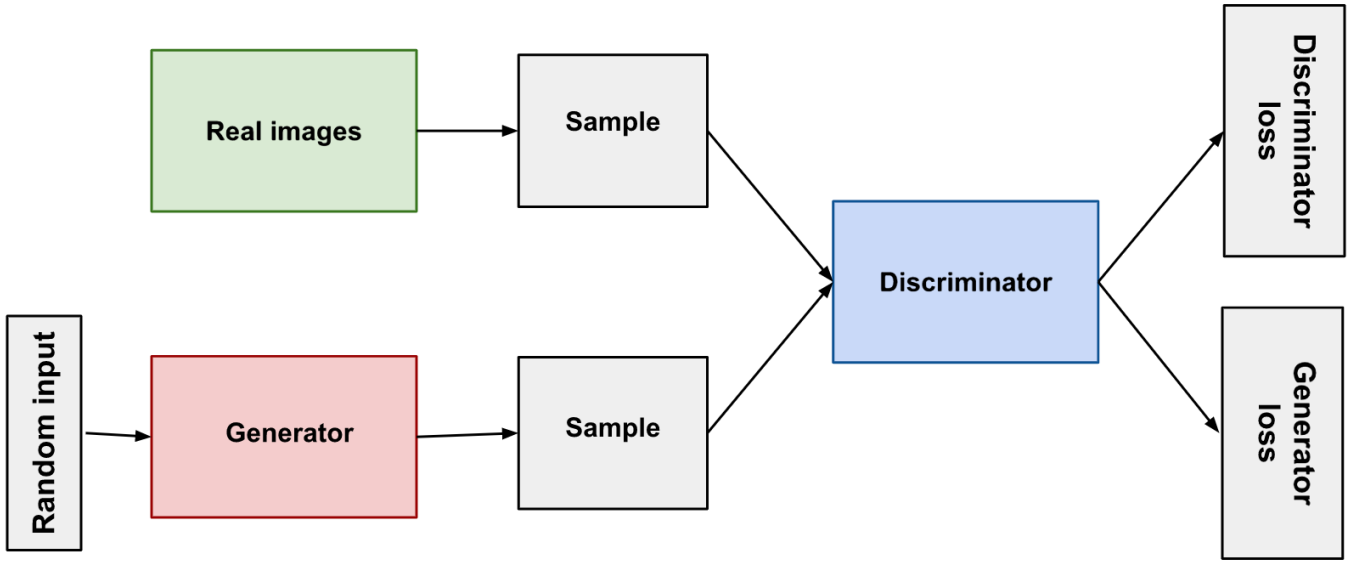
Witches‘ Brew：实验结果

Attack	ResNet-18	MobileNet-V2	VGG11	Average
Poison Frogs	0%	1%	3%	1.33%
Convex Polytopes	0%	1%	1%	0.67%
Clean-Label Backdoors	0%	1%	2%	1.00%
Hidden-Trigger Backdoors	0%	4%	1%	2.67%
Proposed Attack ($K = 1$)	45%	36%	8%	29.67%
Proposed Attack ($K = 4$)	55%	37%	7%	33.00%
Proposed Attack ($K = 6$, Het.)	49%	38%	35%	40.67%

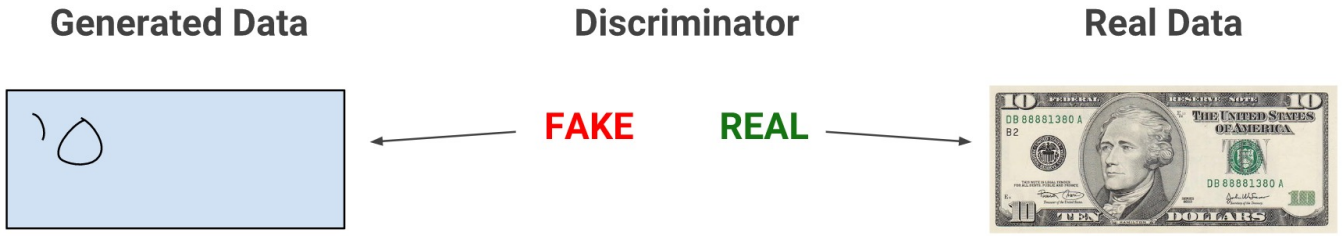
[K = number of ensembled models.]

Geiping, Jonas, et al. “Witches‘ Brew: Industrial Scale Data Poisoning via Gradient Matching.” ICLR 2021.

Generative Attack（生成式攻击）

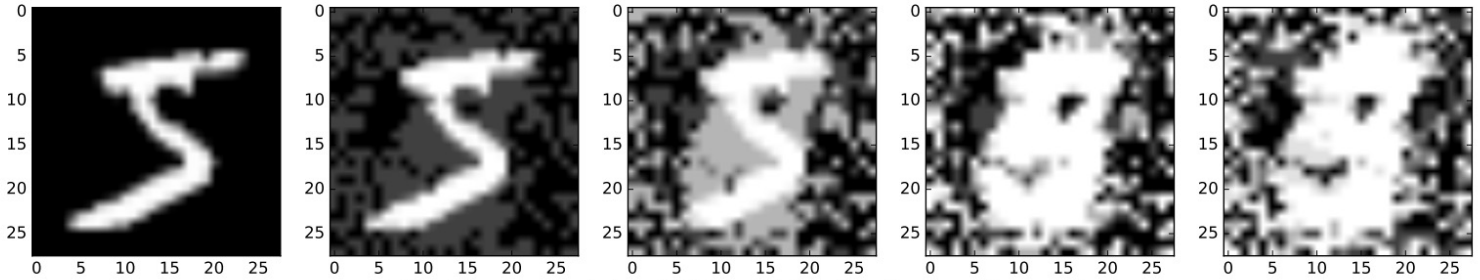


对抗生成网络（GAN）：
一次训练，无限使用



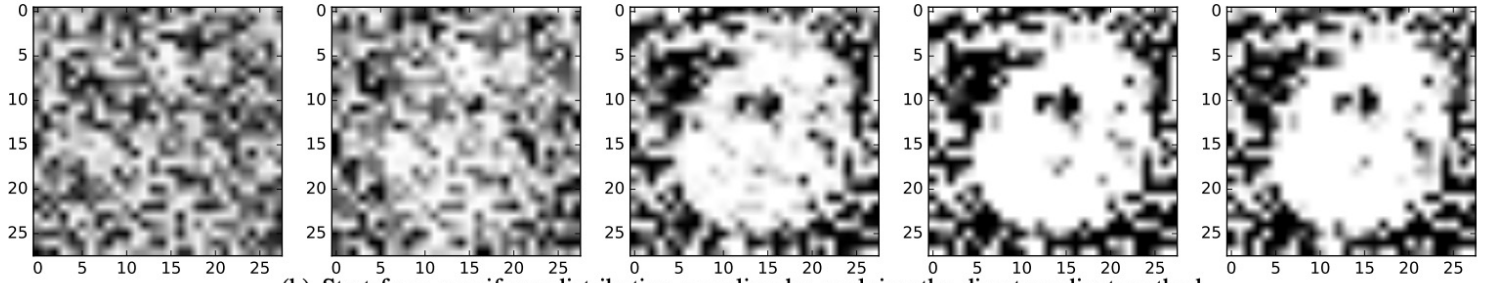
https://developers.google.com/machine-learning/gan/gan_structure

Autoencoder-based Generative Attack



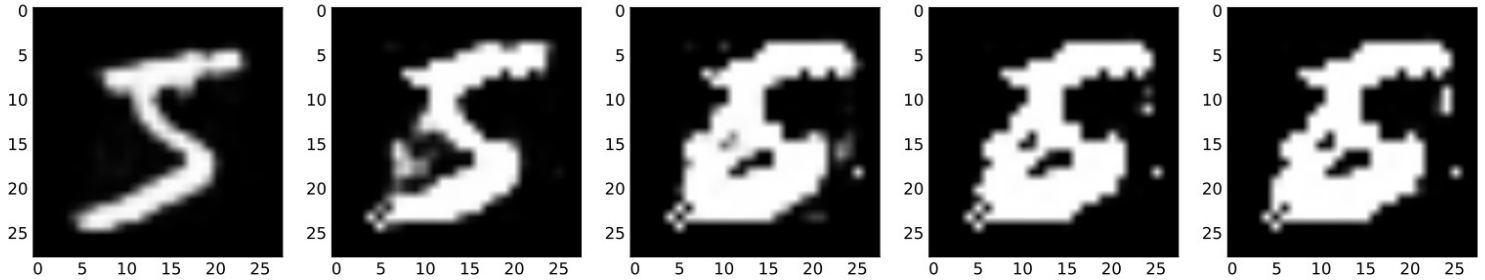
(a) Start from normal data “5” by applying the direct gradient method.

从正常的5开始，使用直接梯度



(b) Start from a uniform distribution sampling by applying the direct gradient method.

从随机噪声开始，使用直接梯度



(c) Start from normal data “5” by applying the generative method.

从正产的5开始，使用生成方法

Yang, Chaofei, et al. "Generative poisoning attack method against neural networks." arXiv:1703.01340 (2017).

pGAN

- 涉及三个模型：
  D（判别器）、G（生成器）、C（分类器）

$$\min_{\mathcal{G}} \max_{\mathcal{D},\mathcal{C}} \alpha\, \mathbb{V}(\mathcal{D},\mathcal{G}) + (1-\alpha)\, \mathbb{W}(\mathcal{C},\mathcal{G})$$

- 对抗损失与GAN一样：

$$\mathbb{V}(\mathcal{D},\mathcal{G}) = \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z}|\mathbf{Y}_p)}[\log(1-\mathcal{D}(\mathcal{G}(\mathbf{z}|\mathbf{Y}_p)))] + \mathbb{E}_{\mathbf{x}\sim p_x(\mathbf{x}|\mathbf{Y}_p)}[\log(\mathcal{D}(\mathbf{x}|\mathbf{Y}_p)].$$

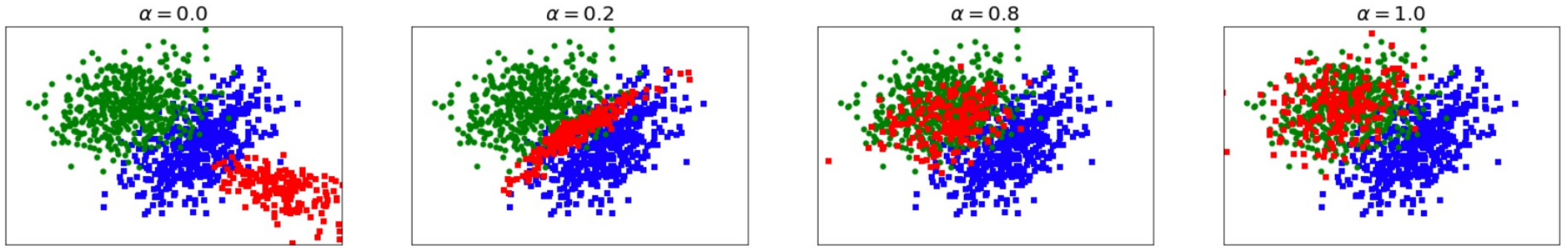
- 分类损失（原始数据+生成数据损失）：

$$\mathbb{W}(\mathcal{C},\mathcal{G}) = -\Big(\lambda\, \mathbb{E}_{z\sim p_z(\mathbf{z}|\mathbf{Y}_p)}[\mathcal{L}_{\mathcal{C}}(\mathcal{G}(\mathbf{z}|\mathbf{Y}_p))] + (1-\lambda)\, \mathbb{E}_{\mathbf{x}\sim p_x(\mathbf{x})}[\mathcal{L}_{\mathcal{C}}(\mathbf{x})]\Big)$$

Yang, Chaofei, et al. "Generative poisoning attack method against neural networks." arXiv:1703.01340 (2017).

pGAN

可以生成真正靠近目标类的投毒样本



- 绿色：正常类（目标类），正常样本
- 蓝色：正常类，正常样本
- 红色：毒化类，毒化样本

Yang, Chaofei, et al. "Generative poisoning attack method against neural networks." arXiv:1703.01340 (2017).

差异化攻击：对哪些样本投毒更有效？

衡量样本影响力的指标：

$$\mathcal{I}(\boldsymbol{z}) = -\mathcal{H}_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(f_{\hat{\theta}}(\boldsymbol{z}))$$

$$s.t.\ \hat{\theta} = \arg\min \sum_{(\boldsymbol{x},y)\sim\mathcal{Z}_{\text{val}}} \mathcal{L}(f_\theta(\boldsymbol{x}), y)$$

对影响大的样本投毒

- $\hat{\theta}$：移除样本(x,y)后得到的模型参数
- $\mathcal{Z}_{val}$：验证数据集
- $\mathcal{H}$：Hessian矩阵

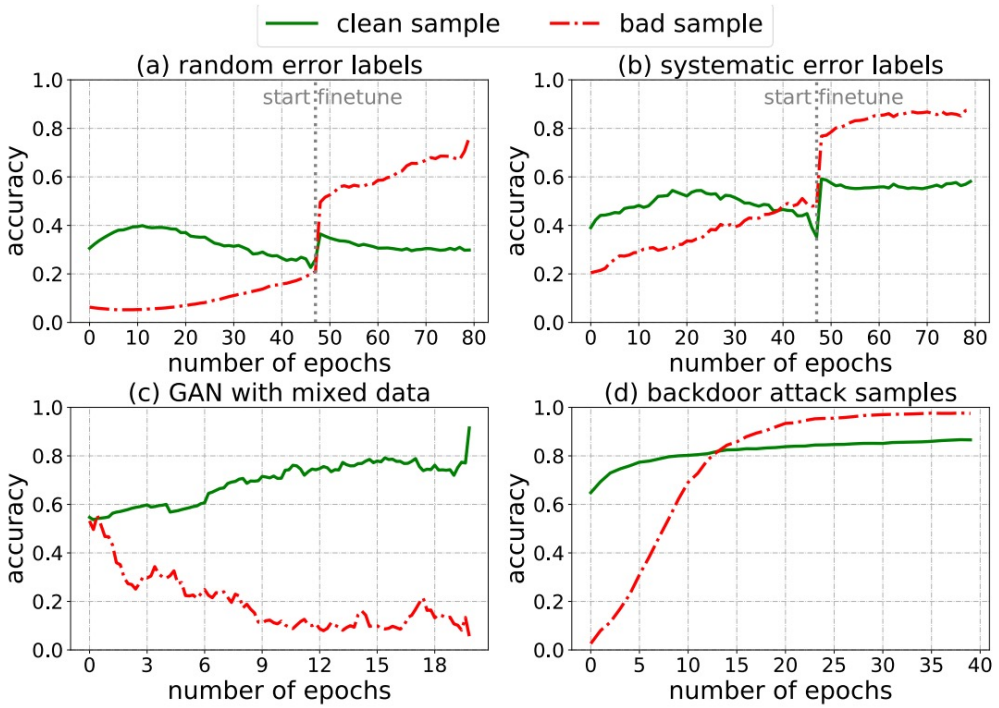
Koh et al. "Stronger data poisoning attacks break data sanitization defenses." Machine Learning 111.1 (2022): 1-47.

Data Poisoning: Attacks and Defenses

- A Brief History of Data Poisoning
- Data Poisoning Attacks
- **Data Poisoning Defenses**
- Poisoning for Data Protection
- Future Research

Data Poisoning Defense

Robust Learning with Trimmed Loss



- **Loss 低的是好样本**
- Loss高的是坏样本
- 让模型尽量在Loss低的样本上训练
- 问题样本：噪声标签、系统噪声、生成模型—+坏数据、后门样本

Shen, Yanyao, and Sujay Sanghavi. “Learning with bad training data via iterative trimmed loss minimization.” ICML 2019

Data Poisoning Defense

Robust Learning with Trimmed Loss

$$\arg\min_{\theta\in\mathfrak{B}} \min_{S:|S|=\lfloor \alpha n \rfloor} \sum_{(\boldsymbol{x},y)\in S} \mathcal{L}(\boldsymbol{x}, y)$$

- 是一个**min-min**问题
  - **内部最小化**：选择低loss的样本子集S
  - **外部最小化**：在子集S上训练模型

Shen, Yanyao, and Sujay Sanghavi. “Learning with bad training data via iterative trimmed loss minimization.” ICML 2019

深度划分聚合（Deep Partition Aggregation，DPA）

分而治之：适用于投毒样本比较少的情况

- 将训练集划分为k个均匀子集：

$$P_i := \{\boldsymbol{t} \in \mathcal{T} | h(\boldsymbol{t}) \equiv i \ (\mod k)\}$$

- 在每个子集上训练一个基分类器：

$$f_i(\boldsymbol{x}) := f(P_i, \boldsymbol{x})$$

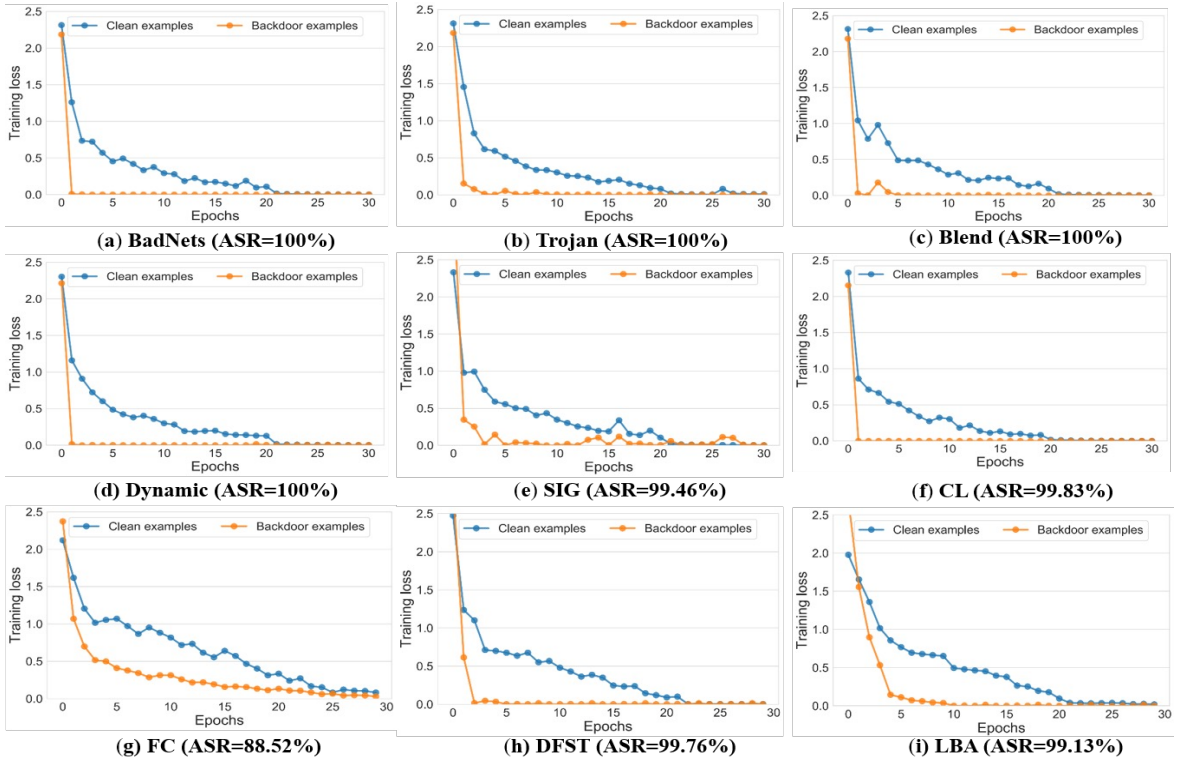
- 投票决策：

$$g_{\mathrm{dpa}}(\mathcal{T}, \boldsymbol{x}) := \arg\max_c n_c(\boldsymbol{x}) \qquad n_c(\boldsymbol{x}) := |\{i \in [k] | f_i(\boldsymbol{x}) = c\}|$$

Levine, Alexander, and Soheil Feizi. “Deep partition aggregation: Provable defense against general poisoning attacks.” ICLR 2021

反后门学习（Anti-Backdoor Learning, ABL）

学的快的样本不是好样本



- **Training loss on Clean samples (blue) VS. Poisoned examples (yellow)**
  - 研究10种基于投毒的后门攻击
  - 毒化样本在训练初期就学完了
  - 毒化样本的损失下降很快

Li, Yige, et al. “Anti-backdoor learning: Training clean models on poisoned data.” NeurIPS 2021

反后门学习（Anti-Backdoor Learning, ABL）

先隔离再反学习

- Problem Formulation

$$\mathcal{L} = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell(f_\theta(\boldsymbol{x}),y)] = \underbrace{\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_c}[\ell(f_\theta(\boldsymbol{x}),y)]}_{\text{clean task}} + \underbrace{\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_b}[\ell(f_\theta(\boldsymbol{x}),y)]}_{\text{backdoor task}},$$

- Overview of ABL
  - Stage 1: **Backdoor Isolation**; $(0 \le t < T_{te})$, t: current epoch; T_{te}: turning epoch
  - Stage 2: **Backdoor Unlearning**. $(T_{te} \le t < T)$ T: total epoch

$$\mathcal{L}_{\text{ABL}}^t = \begin{cases} \mathcal{L}_{\text{LGA}} = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\text{sign}(\ell(f_\theta(\boldsymbol{x}),y)-\gamma)\cdot\ell(f_\theta(\boldsymbol{x}),y)\right] & \text{if } 0 \le t < T_{te} \\ \mathcal{L}_{\text{GGA}} = \mathbb{E}_{(\boldsymbol{x},y)\sim\widehat{\mathcal{D}}_c}\left[\ell(f_\theta(\boldsymbol{x}),y)\right] - \mathbb{E}_{(\boldsymbol{x},y)\sim\widehat{\mathcal{D}}_b}\left[\ell(f_\theta(\boldsymbol{x}),y)\right] & \text{if } T_{te} \le t < T, \end{cases}$$

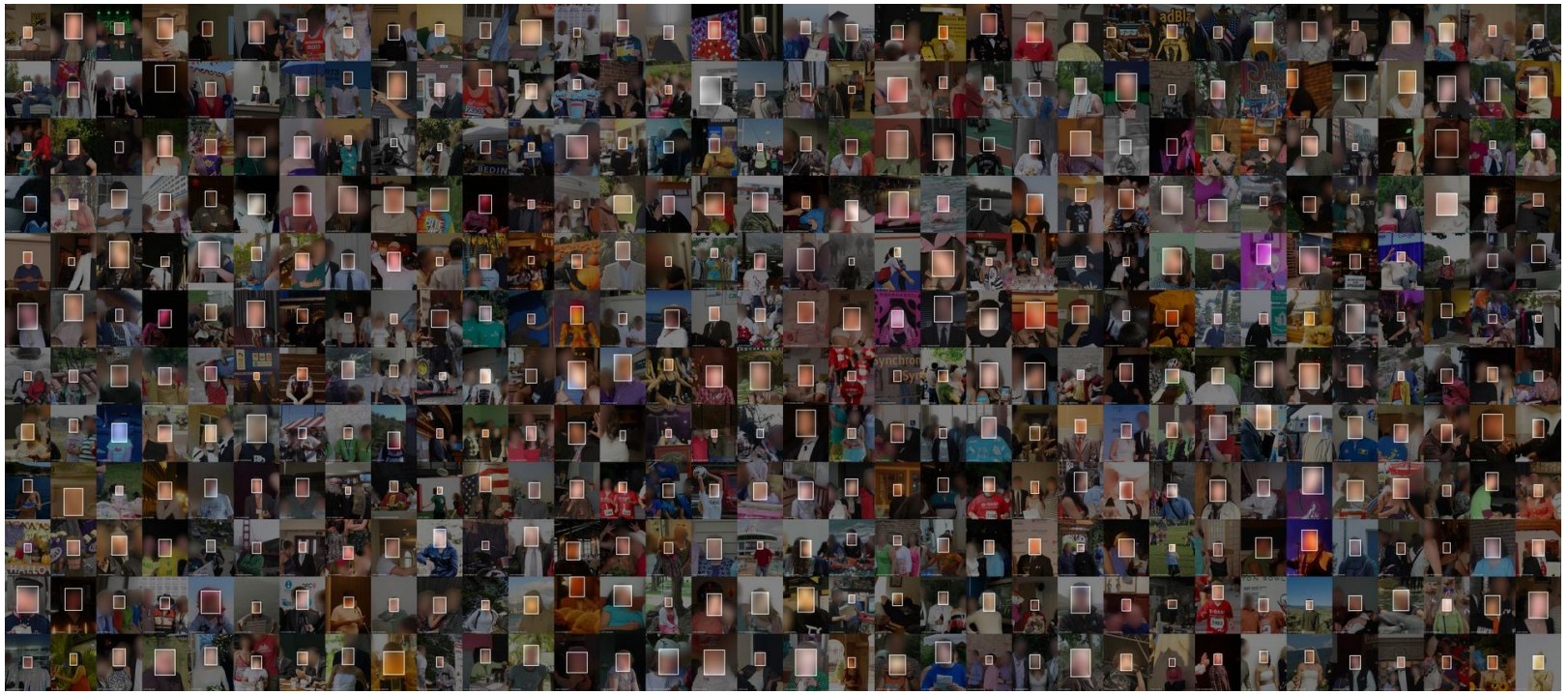
LGA: local gradient ascent; GGA: global gradient ascent

Data Poisoning: Attacks and Defenses

- A Brief History of Data Poisoning
- Data Poisoning Attacks
- Data Poisoning Defenses
- **Poisoning for Data Protection**
- Future Research

Unlearnable Examples

互联网上充斥着大量的个人数据



https://exposing.ai/megaface/

Personal Data Are Used For Training Commercial Models

Dataset collected from the Internet:

1. Without awareness [1].
2. Training commercial models [2].
3. Privacy concerns [3].

[1] Prabhu & Abeba, "Large image datasets: A pyrrhic win for computer vision?." *arXiv:2006.16923,* 2020.
[2] Hill Kashmir, “The Secretive Company That Might End Privacy as We Know It.” NY times, 2020.
[3] Shan, Shawn, et al. "Fawkes: Protecting personal privacy against unauthorized deep learning models." *USENIX Security Symposium,* 2020

Unlearnable Examples

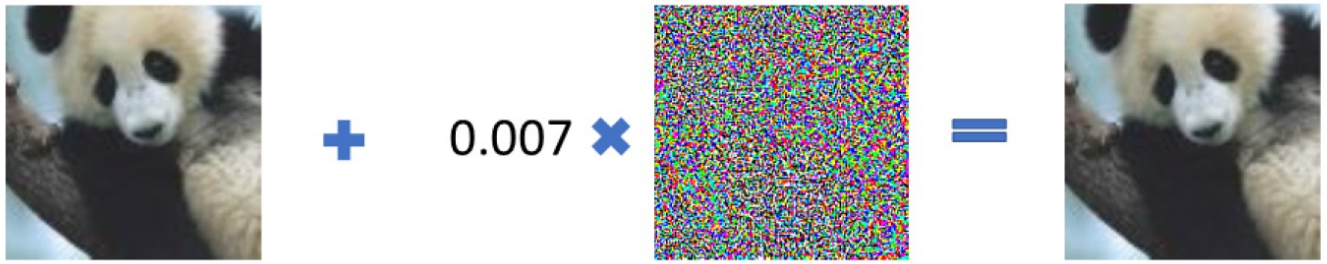
- **Goal:** making data unlearnable (unusable) to machine learning

Modify Training Images -> Make them Useless

Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Adversarial Noise = Error-maximizing Noise

Adversarial noise can mislead ML models



"panda"
57.7% confidence

small adversarial
perturbations

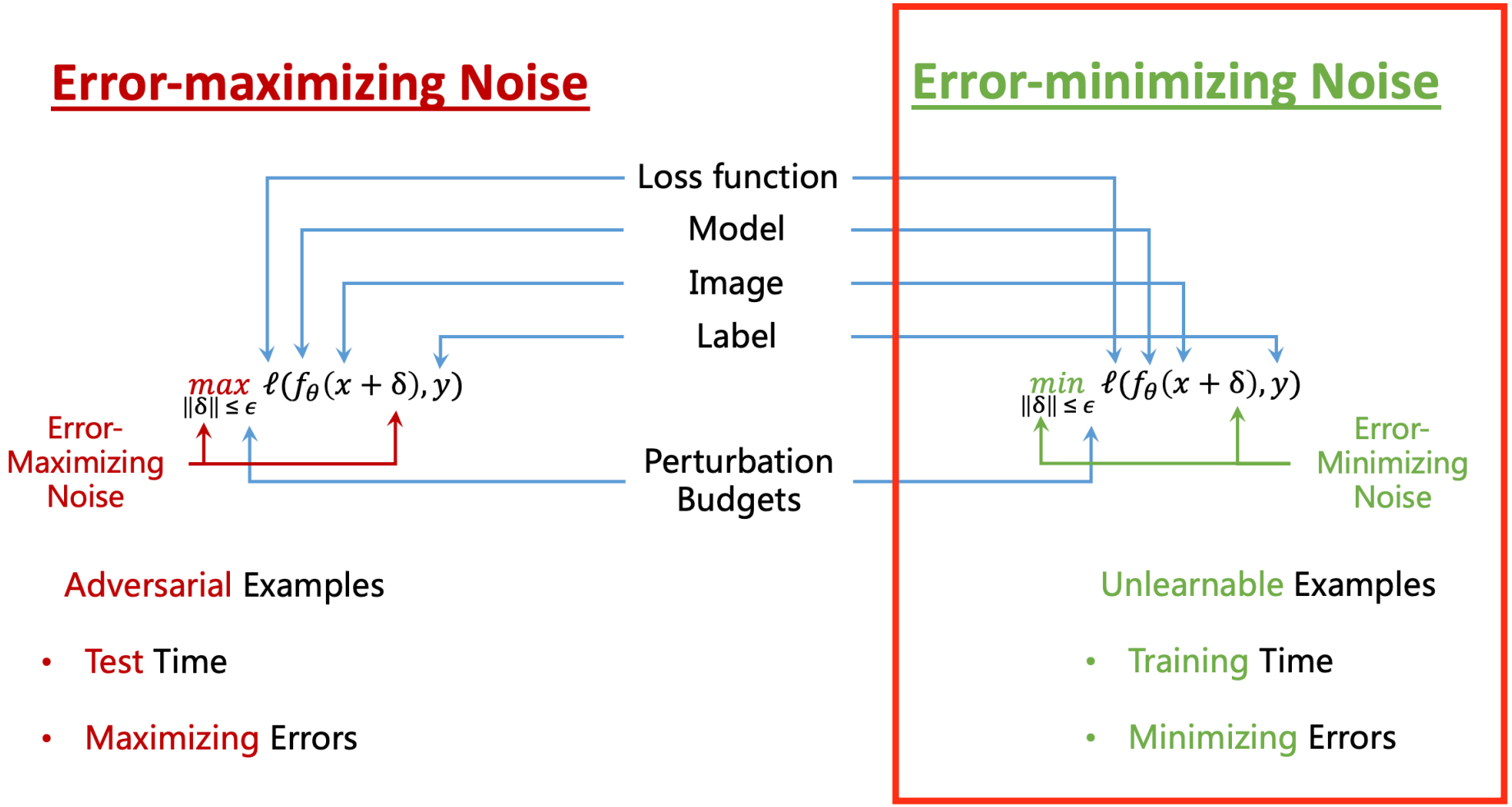
"gibbon"
99.3 % confidence

✓ Adversarial noises are small, imperceptible to human eyes.

Adversarial Examples fool DNN at *test time* by maximizing errors.

Szegedy et al. 2013, Goodfellow et al. 2014

NO Error to Learn?



Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Generating Error-Minimizing Noise

想要影响模型的训练那一定是一个双层优化问题

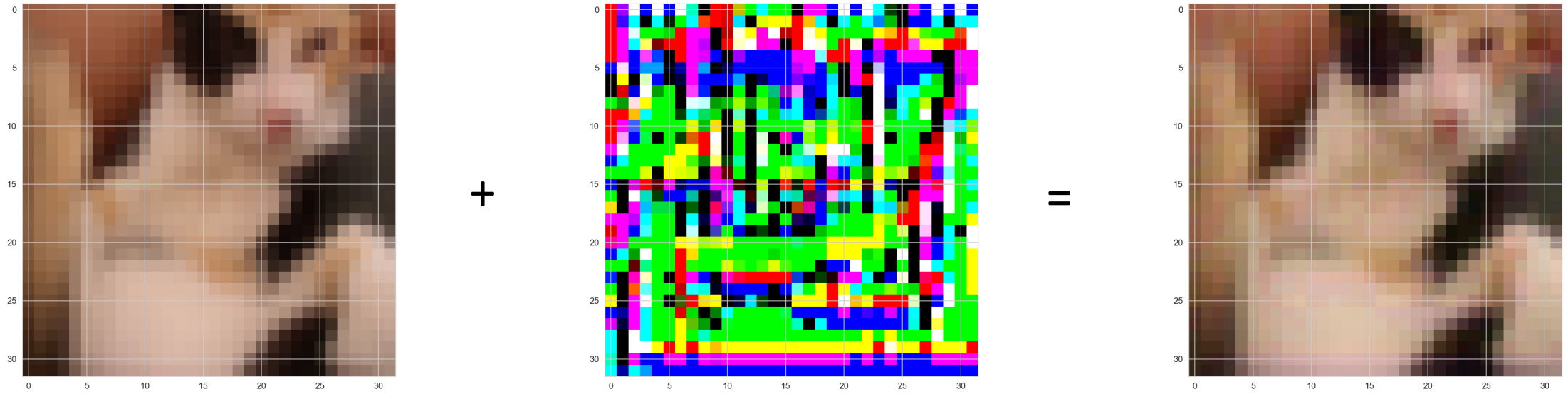
$$\arg\min_{\theta} \mathbb{E}_{(x,y)} \min_{\delta} \ell(f_\theta(x+\delta), y) \quad s.t. \|\delta\|_\infty \leq \epsilon$$

A min-min bi-level optimization objective to find error-minimizing noise δ.

Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Sample-wise Noise

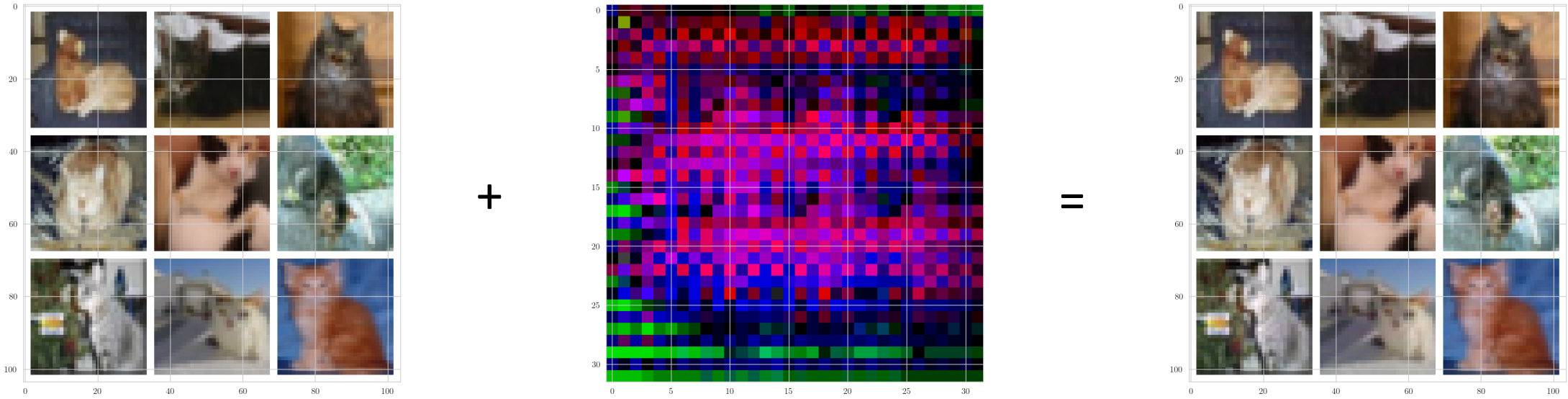
每个样本都有一套自己的噪声



Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Class-wise Noise

每类样本共享一套噪声

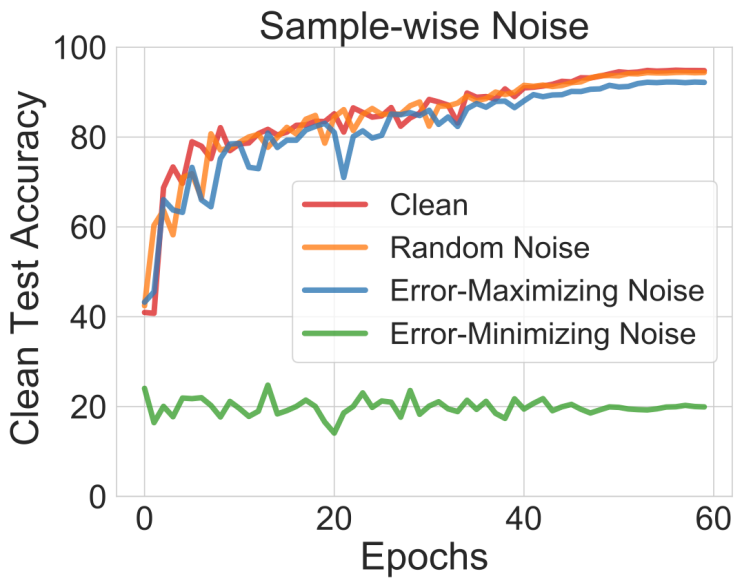


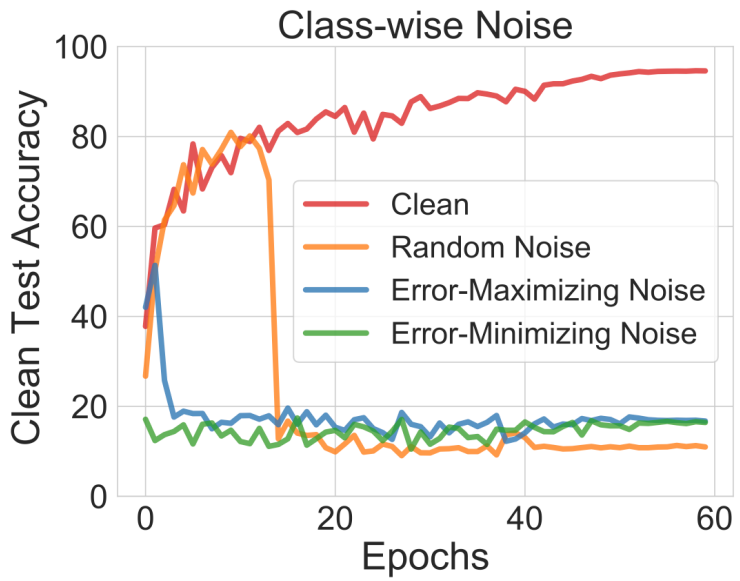
规律？为什么这一个噪声图案可以让一整类的数据没有了错误？

Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Experiments

Comparison the effect of different noises on training:





Error-Minimizing noise can create unlearnable examples in both settings.

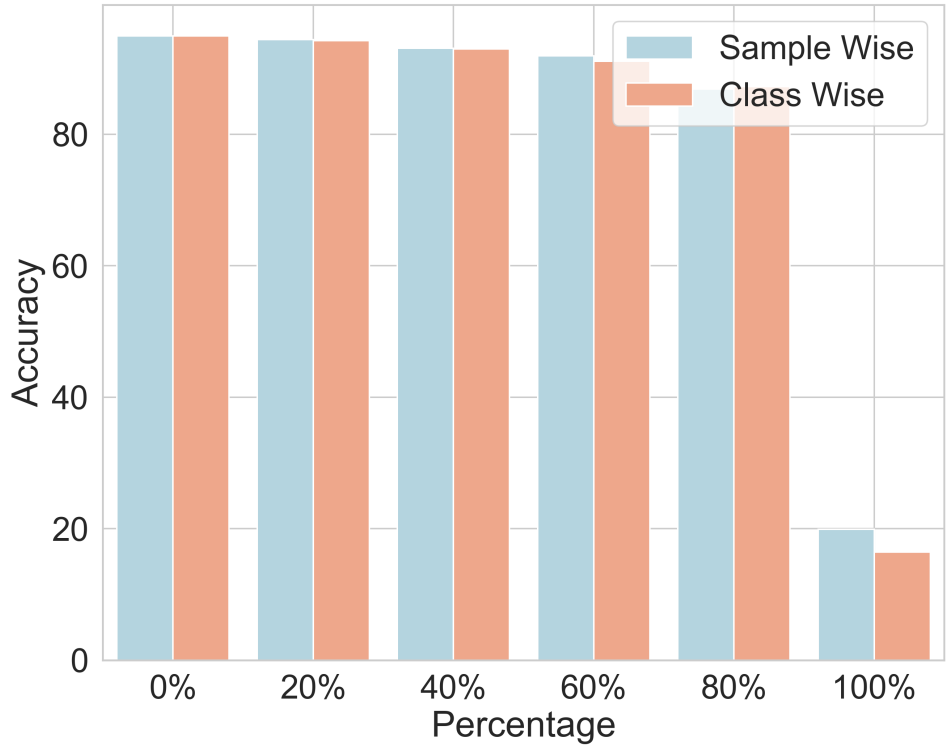
Huang, Hanxun, et al. “Unlearnable examples: Making personal data unexploitable.” ICLR 2021.

Experiments

- ❑ Is the noise transferable to other models?
  - ✓ Yes
- ❑ Is the noise transferable to other datasets?
  - ✓ Yes
- ❑ Is the noise robust to data augmentation?
  - ✓ Yes

Experiments

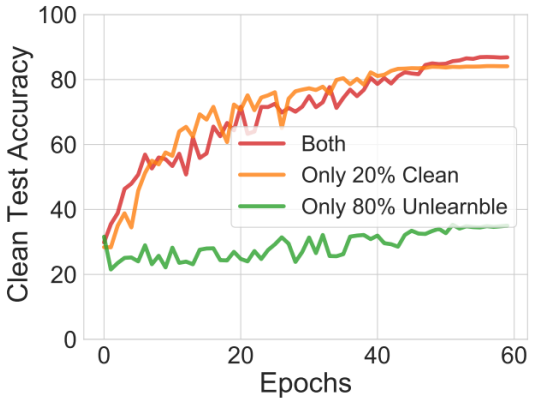
What percentage of the data needs to be unlearnable?



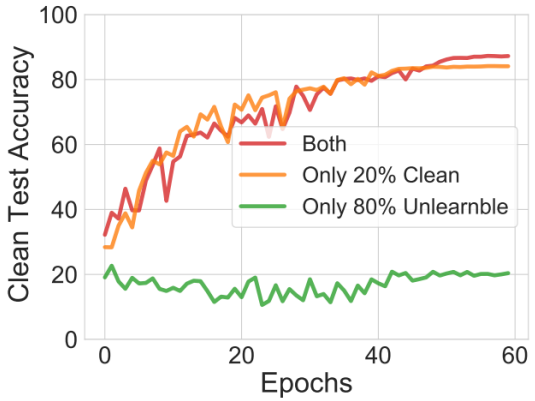
Unfortunately, it needs 100% training data to be poisoned.

Experiments

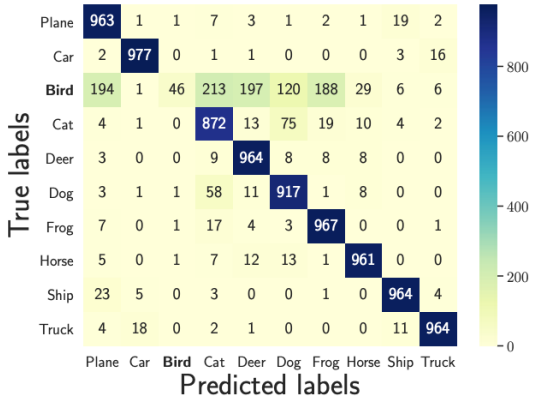
How about protecting part of the data or just one class?



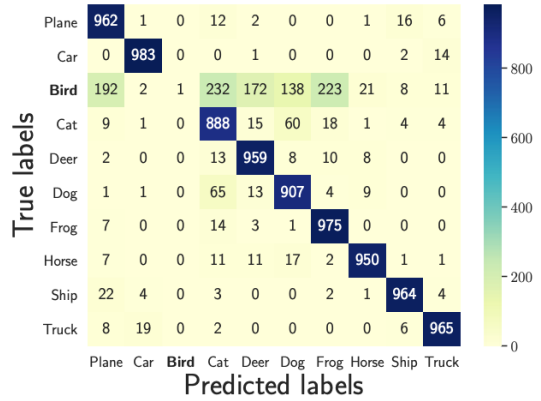
(a) Sample-wise Δ_s



(b) Class-wise Δ_c



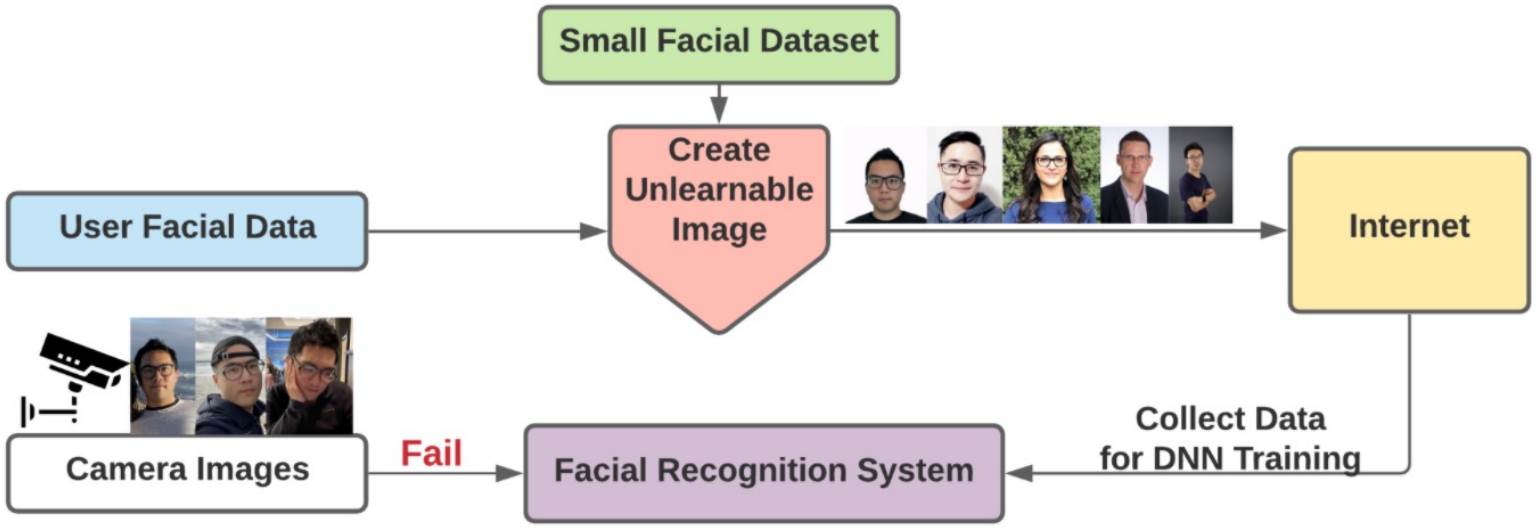
(c) Sample-wise Δ_s



(d) Class-wise Δ_c

Unlearnable Examples will not contribute to model training.

Protecting Face Images



- No more facial recognitions?
- If everyone post unlearnable images.

Figured by MIT Technology Review

MIT Technology Review

My account

Topics Magazine Newsletters Events

MS TECH | UNSPLASH

Artificial intelligence / Face recognition

How to stop AI from recognizing your face in selfies

A growing number of tools now let you stop facial recognition systems from training on your personal photos

by **Will Douglas Heaven** May 5, 2021

https://www.technologyreview.com/2021/05/05/1024613/stop-ai-recognizing-your-face-selfies-machine-learning-facial-recognition-clearview

Conclusion & Limitations

- ✓ A new exciting research problem.
- ✓ Unlearnable Examples.
- ✓ Error-minimizing noise.

- ➢ Limitations to representational learning.
- ➢ Limitations to adversarial training (已被ICLR2022的一篇工作解决: Robust Unlearnable Examples).

Related works:

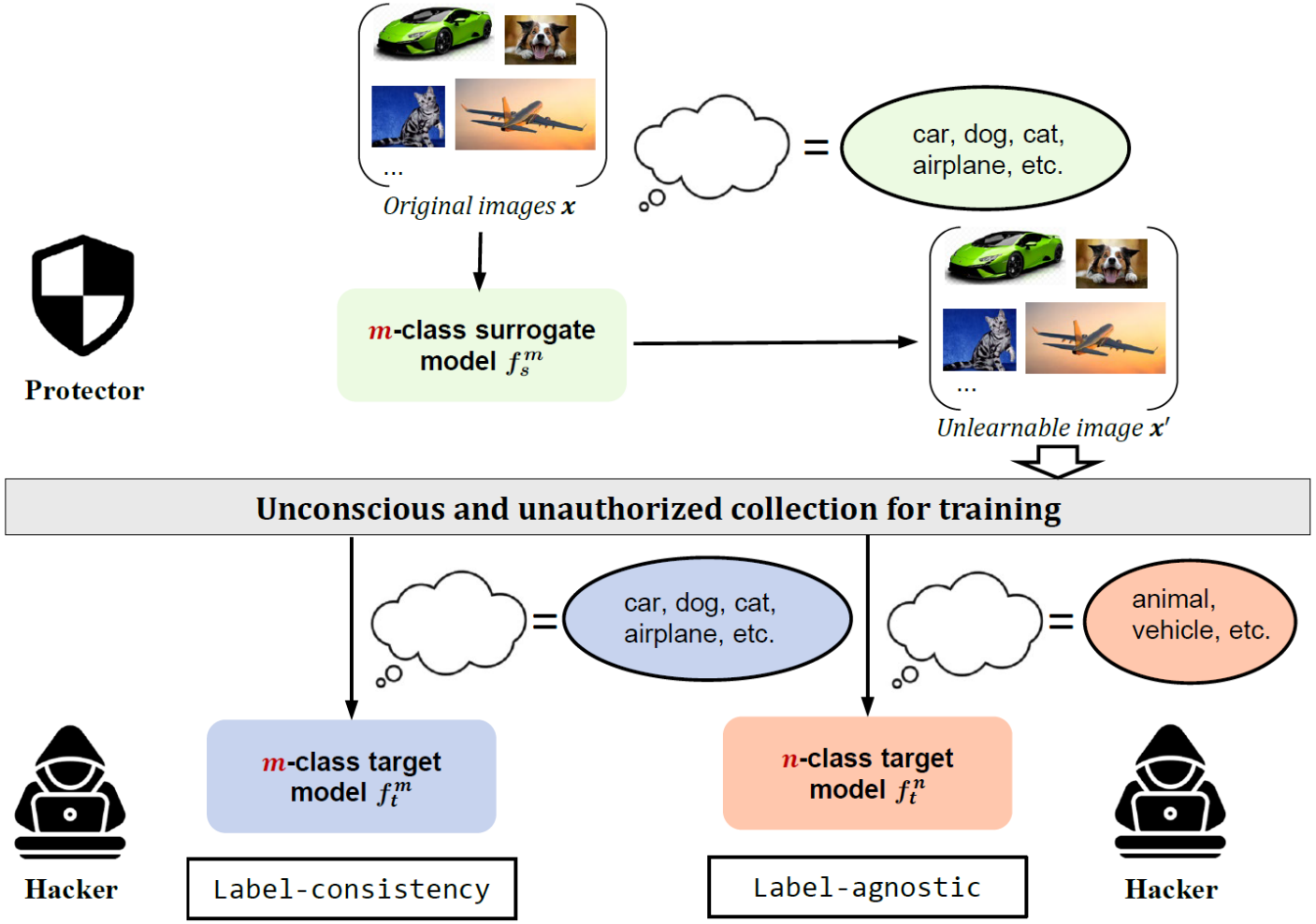
1. **Cherepanova et al. "LowKey: Leveraging Adversarial Attacks to Protect Social Media Users from Facial Recognition." *ICLR,* 2021.**
2. **Fowl et al. "Adversarial Examples Make Strong Poisons." NeurIPS 2021.**
3. **Fowl et al. "Preventing unauthorized use of proprietary data: Poisoning for secure dataset release." arXiv:2103.02683**
4. **Radiya-Dixit and Tramèr. "Data Poisoning Won't Save You From Facial Recognition." arXiv:2106.14851**
5. **Shan et al. "Fawkes: Protecting privacy against unauthorized deep learning models." USENIX Security, 2021**

Unlearnable Clusters

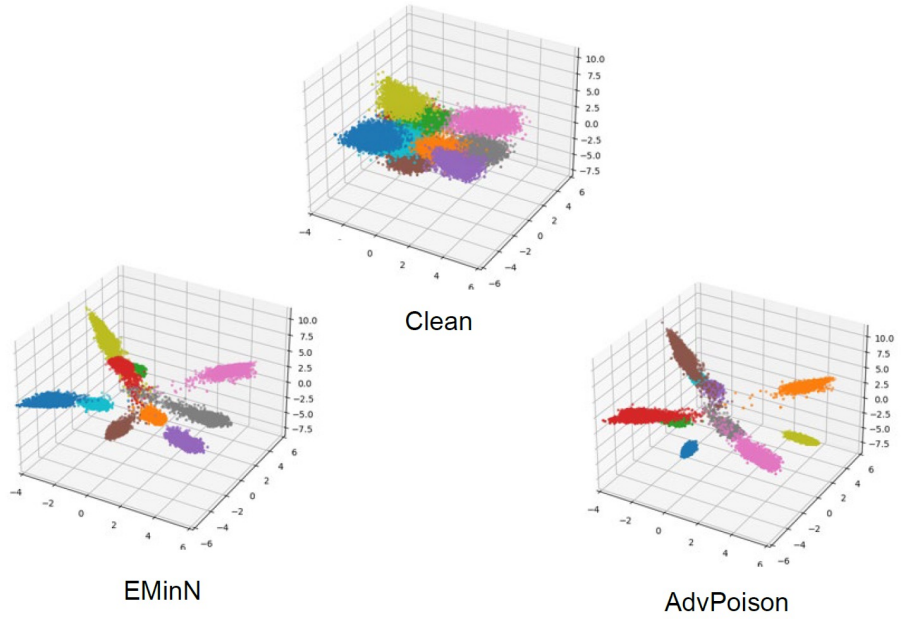
Unlearnable Clusters: Towards Label-agnostic Unlearnable Examples

Jiaming Zhang[1*] Xingjun Ma[2†] Qi Yi[1] Jitao Sang[1,4†] Yu-Gang Jiang[2]
Yaowei Wang[4] Changsheng Xu[3,4]
[1]Beijing Jiaotong University [2]Fudan University [3]Chinese Academy of Sciences [4]Peng Cheng Lab

Protection Against Unknown Labelling



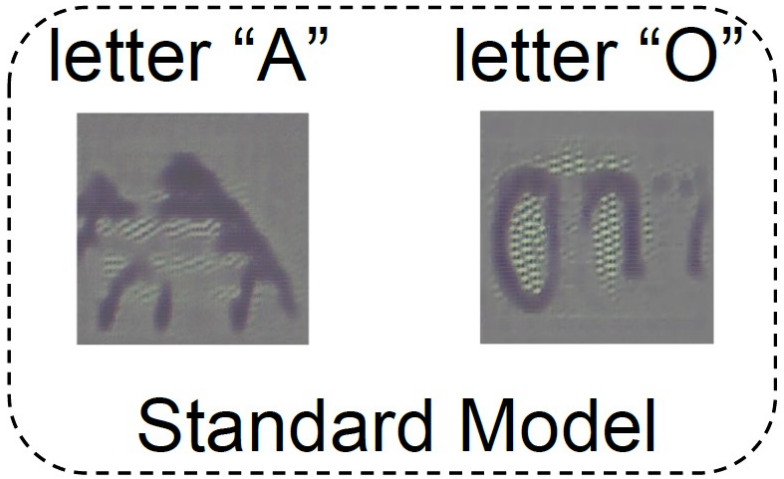
Existing Approaches



- Error-minimizing noise 使模型的训练损失下降到0来让模型觉得“已经没有什么信息可学了”[1]。
- Adversarial Poisoning 利用非鲁棒特征[2]这个概念使模型去学习错误的非鲁棒特征[3]。

[1] Unlearnable examples: Making personal data unexploitable, ICLR 2021.
[2] Adversarial Examples Are Not Bugs, They Are Features, NeurIPS 2019.
[3] Adversarial examples make strong poisons, NeurIPS 2021.

Universal Adversarial Perturbation (UAP)

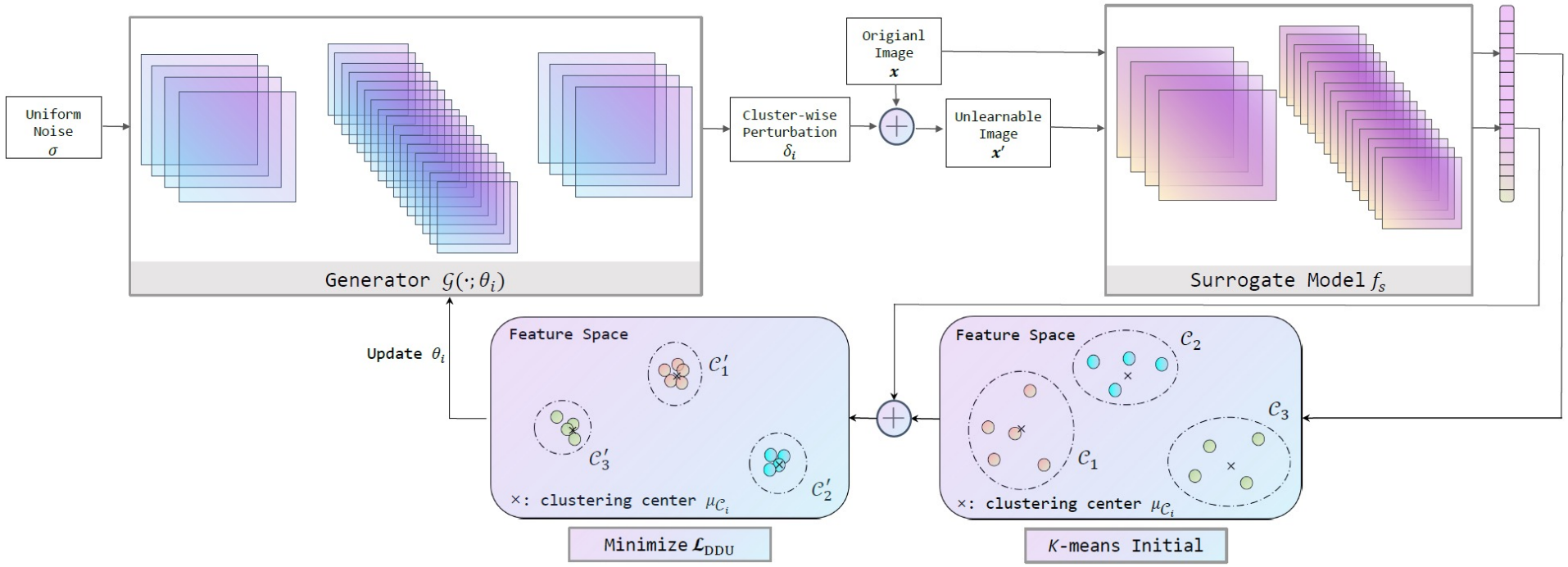


- Universal Adversarial Perturbation (UAP) 是指被施加在任一图像上之后都弄愚弄模型的一种class-wise perturbation[1]。
- 它既可以“覆盖”图像中原本的语义特征，还可以“独立”地工作[2]。

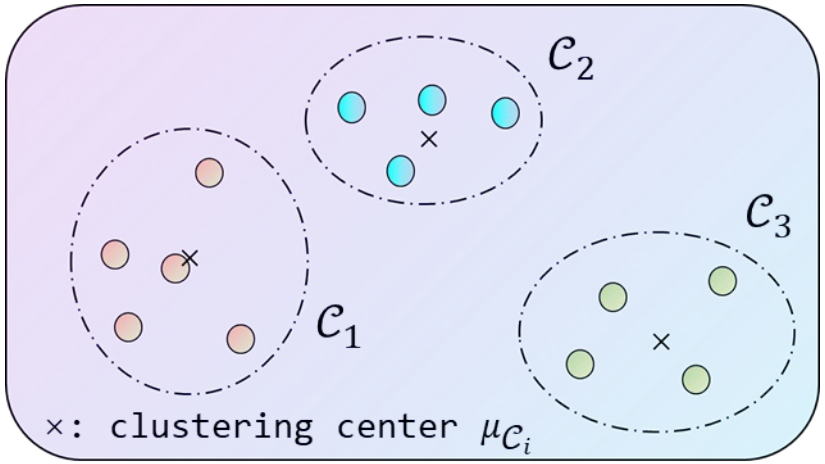
[1] Universal adversarial perturbations, CVPR2017.
[2] ImageNet Pre-training Also Transfers Non-robustness, AAAI2023.

Unlearnable Clusters (UCs)

在不依赖标签信息（分类层）的前提下，实现打破一致性（uniformity）和差异性（discrepancy）。



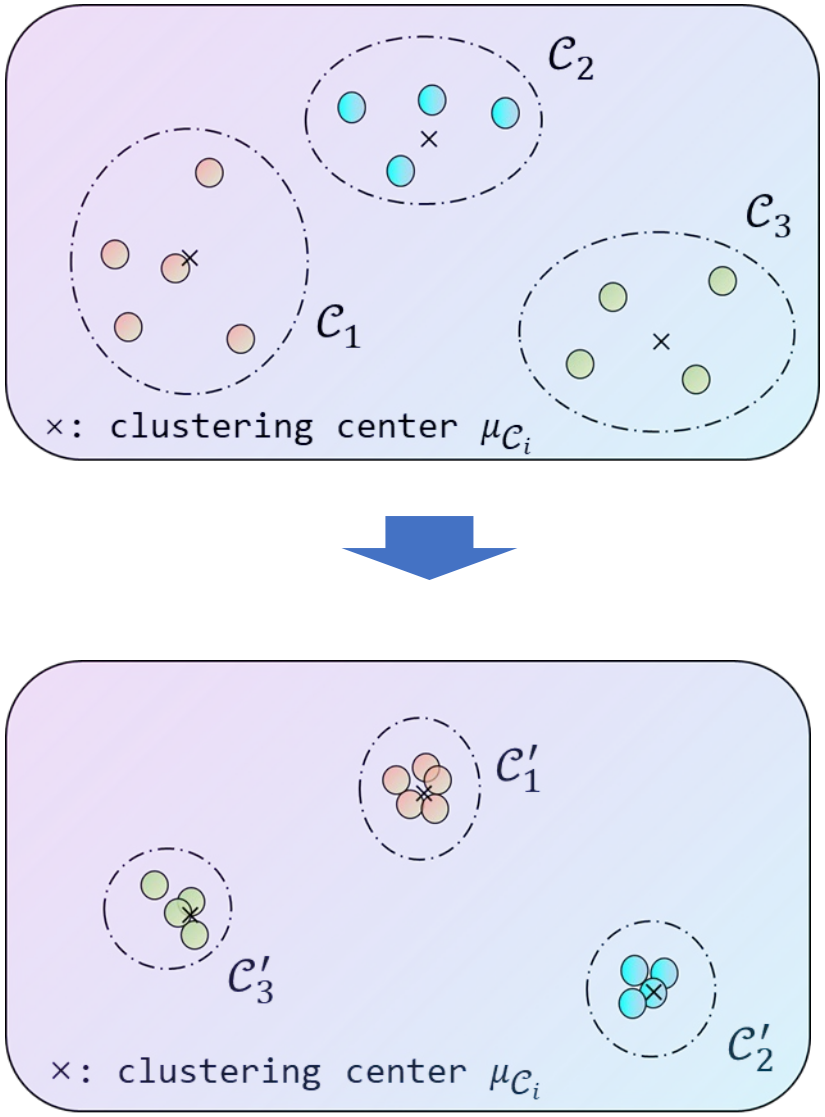
K-means Initial



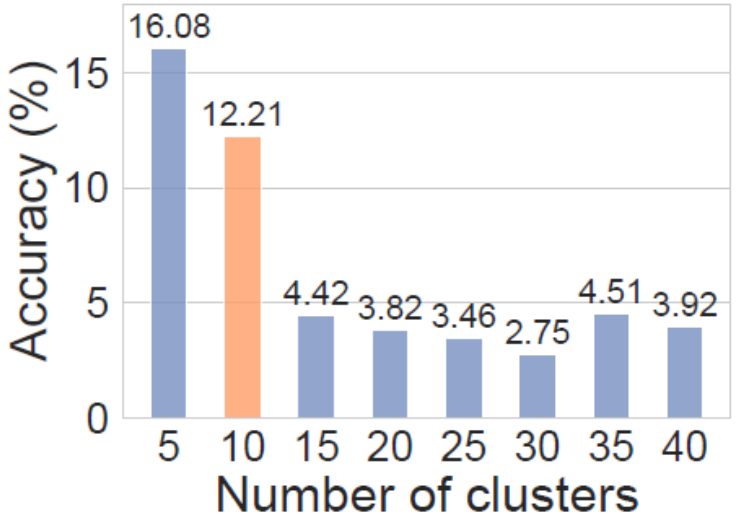
K-means Initial

- 给定代理模型f_s,输入干净的数据集D_c,可以得到特征$E = [e_1, \dots, e_k]$, k是样本数。
- 使用K-means将 E 分为 p个 簇：$C = \{C_1, \dots, C_p\}$,对应的簇中心$\mu_c = \{\mu_{C_1}, \dots, \mu_{C_p}\}$。

Disrupting Discrepancy and Uniformity

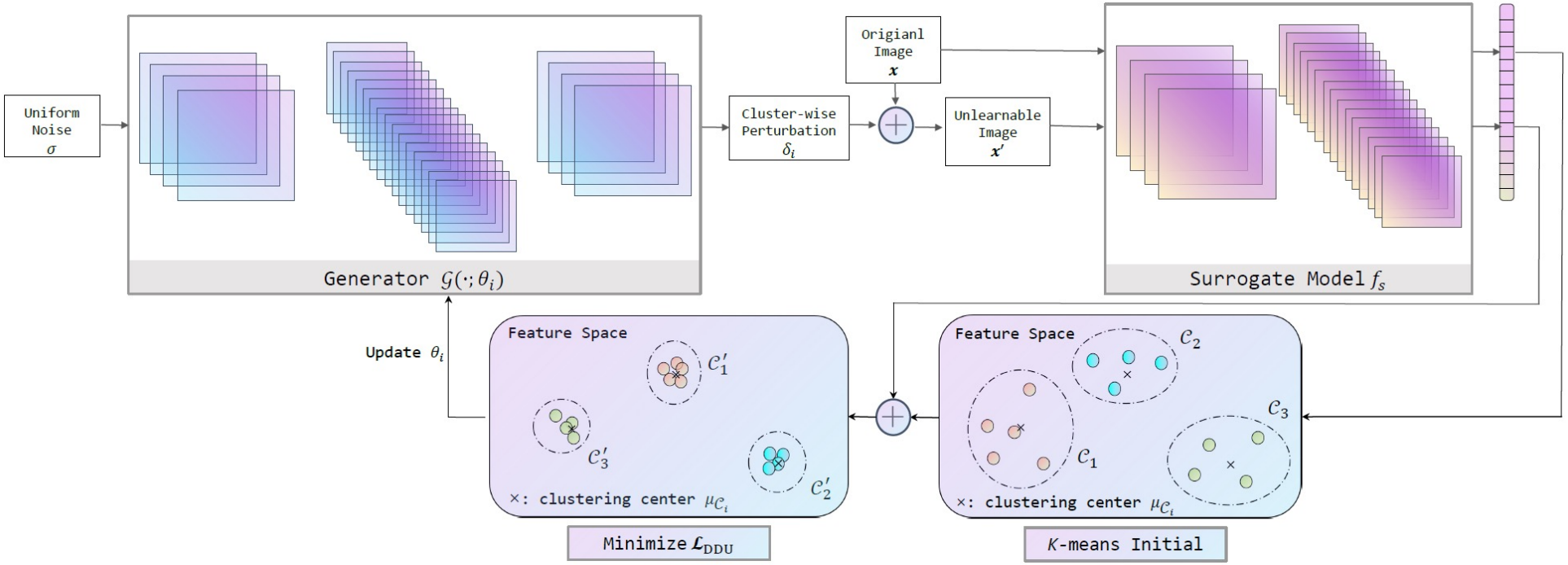


- 先“聚集”再“转圈”。
- 每一个簇有一个固定的cluster-wise噪声，也就是要生成p个噪声。



(a) Effect of p on UC

Methodology



$$\theta_i = \arg\min_{\theta_i} \mathcal{L}_{\text{DDU}}(\mathcal{C}_i, g(\mu_{\mathcal{C}_i}), \theta_i)$$

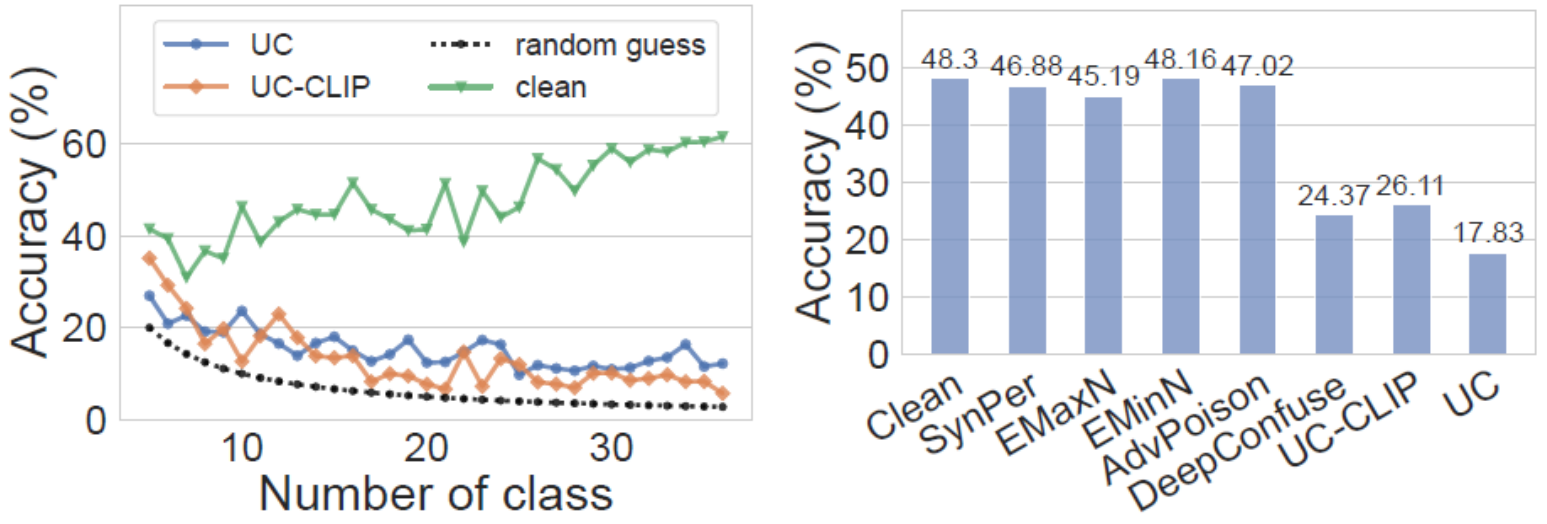
$$= \arg\min_{\theta_i} \sum_{\boldsymbol{x}_{ij} \in \mathcal{C}_i} d(f_s(\boldsymbol{x}_{ij} + \mathcal{G}(\sigma;\theta_i)), g(\mu_{\mathcal{C}_i})),$$

Experiments

Table 1. The test accuracy (%) of different target models trained on the unlearnable datasets generated by our UC/UC-CLIP and the 5 baseline methods, under the label-agnostic setting. The top-2 best results are highlighted in **bold**.

Methods	ResNet-18						EfficientNet-B1						RegNetX-1.6GF					
	Pets	Cars	Flowers	Food	SUN397	ImageNet*	Pets	Cars	Flowers	Food	SUN397	ImageNet*	Pets	Cars	Flowers	Food	SUN397	ImageNet*
Clean	62.31	67.18	67.18	78.97	43.08	77.76	48.68	72.33	52.46	80.29	42.84	78.04	44.86	63.84	52.69	84.02	43.27	80.78
SynPer	52.60	53.50	52.74	74.80	38.26	74.69	28.02	58.34	42.93	74.99	35.92	72.94	34.51	45.54	47.16	77.65	37.78	60.38
EMaxN	54.70	52.95	51.70	73.77	37.57	73.82	33.71	55.64	42.66	74.40	37.30	73.72	34.26	43.40	46.25	78.76	37.82	76.72
EMinN	52.96	54.43	50.58	75.47	38.48	74.20	36.88	54.23	44.06	75.54	37.20	72.20	37.04	39.67	47.34	79.43	36.82	74.86
AdvPoison	50.86	51.91	50.64	75.07	38.51	73.76	37.99	50.08	**41.65**	74.88	36.44	72.54	34.29	46.06	47.41	78.64	36.42	76.32
DeepConfuse	53.72	51.11	50.94	73.13	34.41	55.12	35.54	47.15	43.28	72.91	35.22	45.74	33.71	41.15	46.01	77.26	33.52	49.88
UC (Ours)	**12.21**	**33.57**	**35.55**	**55.29**	**20.38**	**54.80**	**17.06**	**13.92**	42.28	**53.45**	**22.97**	**32.30**	**4.28**	**29.46**	**33.79**	**64.48**	**22.28**	**56.10**
UC-CLIP (Ours)	4.69	4.74	**10.07**	**19.07**	3.89	**39.78**	6.49	**15.33**	**14.13**	**17.44**	**12.95**	**31.82**	3.87	4.18	8.12	**26.76**	**6.04**	**41.66**

Experiments



(a) Different labelings (b) Unsupervised exploitation

Figure 4. (a) The accuracy of ResNet-18 target models trained on the unlearnable Pets dataset but with its labels were re-labeled by the hacker into 5 to 35 classes. (b) Comparison of our approach with the baselines on Pets dataset against ResNet-18 target model trained via self-supervised SimCLR.

Experiments

Table 3. The test accuracy (%) of ResNet-18 trained using different defenses against our methods on Pets dataset.

METHODS	NO DEFENSE	MIXUP	GAUSSIAN	CUTMIX	CUTOUT
UC	12.21	14.34	24.26	14.50	12.35
UC-CLIP	4.69	11.96	18.59	6.21	12.29

Dataset:37-class Pets

Table 1. Test accuracy (%) of AT against different protections.

	Clean	SynPer	EMaxN	EMinN	AdvPoison	DeepConfuse	UC	UC-CLIP
ρ=1	58.80	45.88	45.25	43.62	43.48	45.20	22.23	14.04
ρ=2	58.01	48.08	44.08	45.22	42.06	42.94	22.58	16.84

Dataset:37-class Pets

Transferable Unlearnable Examples

Published as a conference paper at ICLR 2023

TRANSFERABLE UNLEARNABLE EXAMPLES

Jie Ren*
Michigan State University
renjie3@msu.edu

Han Xu*
Michigan State University
xuhan1@msu.edu

Yuxuan Wan
Michigan State University
wanyuxua@msu.edu

Xingjun Ma
Fudan University
xingjunma@fudan.edu.cn

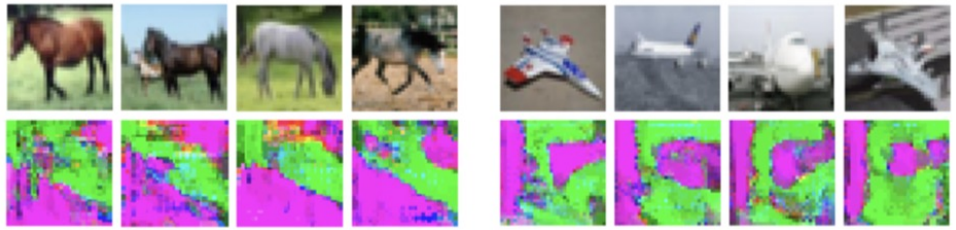
Lichao Sun
Lehigh University
lis221@lehigh.edu

Jiliang Tang
Michigan State University
tangjili@msu.edu

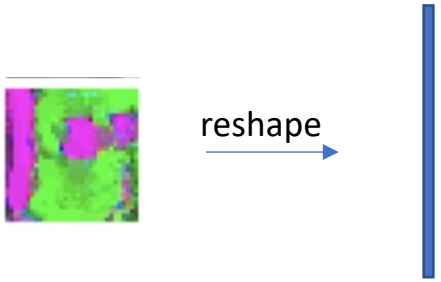
Linear Separability

- Class-wise perturbations

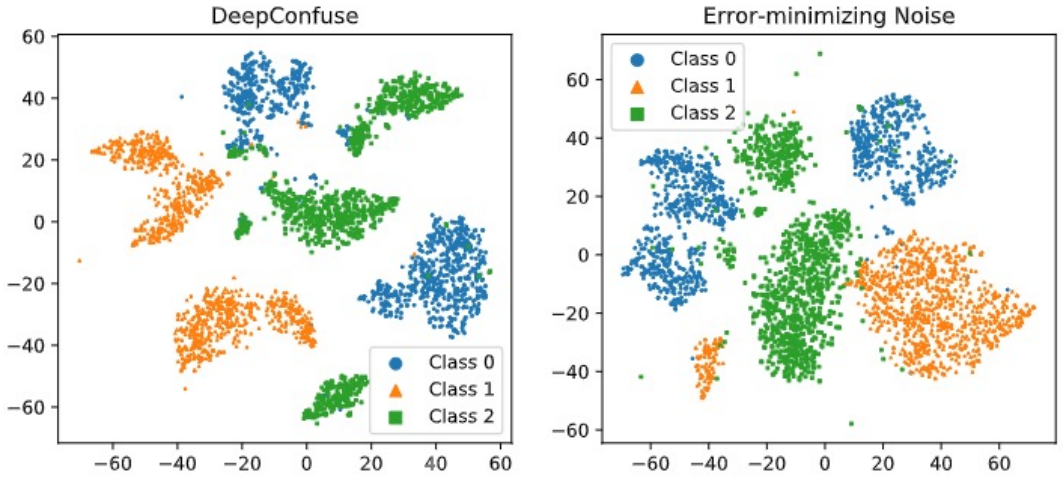
EMN



- Linear Separability



(32 x 32 x 3) dim = 3072 dim



t-SNE visualization in input space

Quantifying Linear Separability

Model: Linear model and Two-layer Neural Network

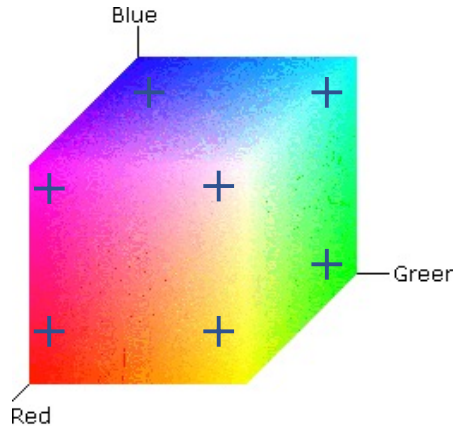
Input x: perturbation
Label y: corresponding label of perturbed image

Algorithm	Linear Model	Two-layer NN
Clean Data	49.9	70.1
DeepConfuse [Feng et al., 2019]	100.0	100.0
NTGA [Yuan and Wu, 2021]	100.0	100.0
Error-minimizing [Huang et al., 2021]	100.0	100.0
Error-maximizing [Fowl et al., 2021b]	91.5	99.9

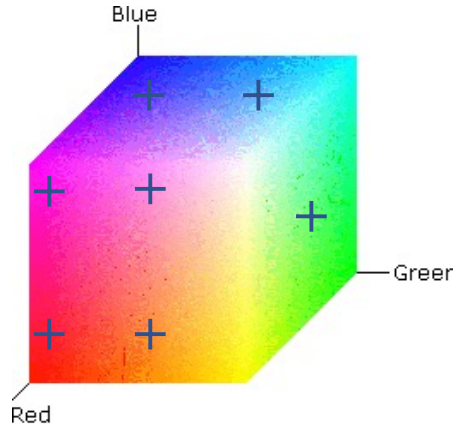
Linearly Separable Perturbations

- Goal: to show simple linear separability is enough for poisoning.
- Synthetic noise (SN):

horse



horse



cat

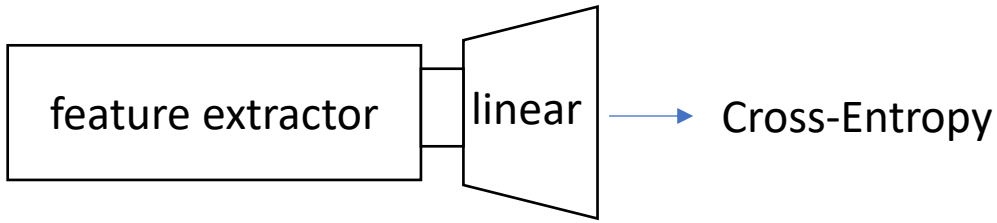
Linearly Separable Perturbations

Algorithm	Test Accuracy (in %)
No Perturbation	94.69
TensorClog [Shen et al., 2019]	48.07
Alignment [Fowl et al., 2021a]	56.65
DeepConfuse [Feng et al., 2019]	28.77
NTGA [Yuan and Wu, 2021]	33.29
Error-minimizing [Huang et al., 2021]	19.93
Error-maximizing [Fowl et al., 2021b]	6.25
Synthetic Perturbations	13.54

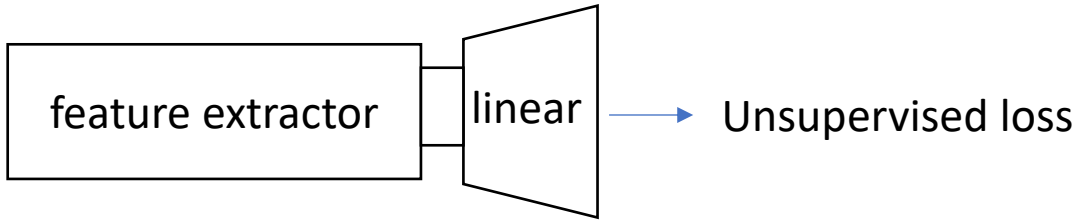
Synthetic noise work the same

The Problem of Transferability

- Lack of transferability
  - Training-wise transferability: comprehensive protection
    - Supervised model

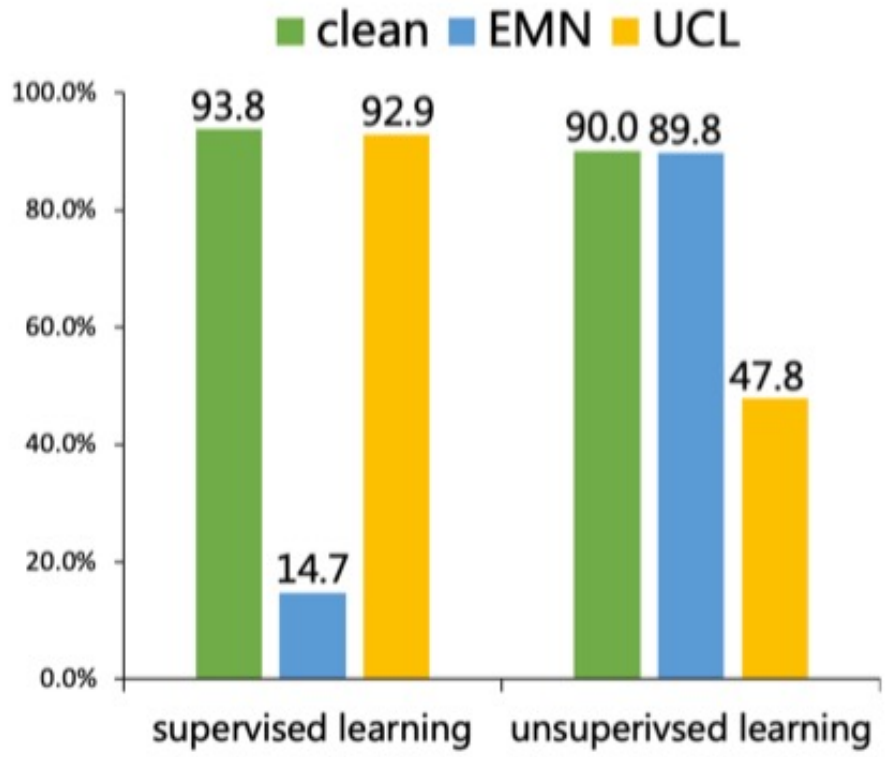


    - Unsupervised model



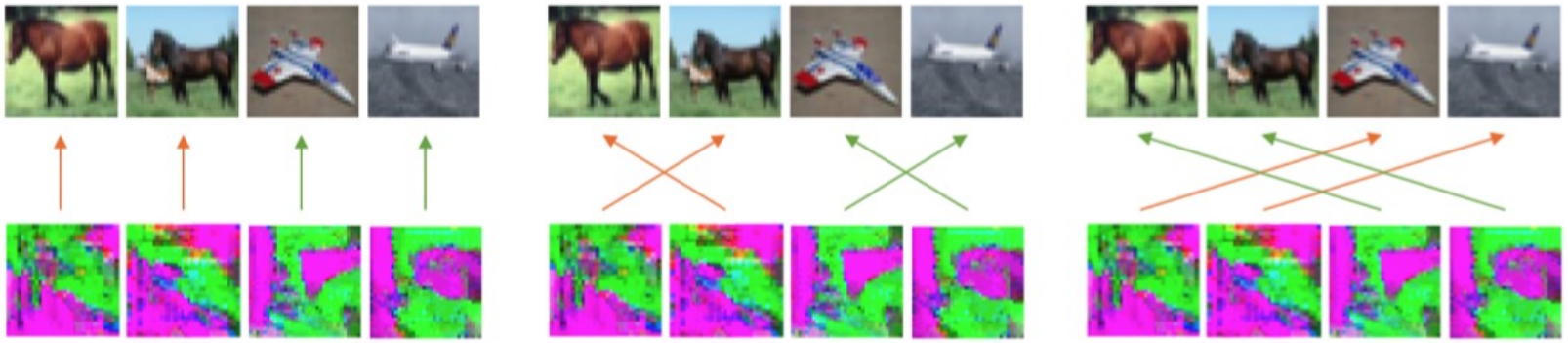
Problems of Previous Method

- Lack of **Training-wise transferability**



Problems of Previous Method

- **Lack of Data-wise transferability**



(a) original corresponding (b) intra-class swap (c) inter-class swap

Figure 3: Original one-to-one correspondence and two swapping methods.

Table 2: Comparison of supervised unlearnable examples on target and non-target samples.

Corresponding	CIFAR10		CIFAR100	
	EMN	SN	EMN	SN
Original	15.88	14.07	6.59	2.13
Intra-class swap	30.74	13.59	21.63	2.44
Inter-class swap	30.15	13.15	35.50	2.73

Class-wise Separability Discriminant (CSD)

- Intra-class distance

$$\boldsymbol{c}_k = \frac{1}{|\{\boldsymbol{\delta}_i : y_i = k\}|} \sum_{\{\boldsymbol{\delta}_i : y_i = k\}} \boldsymbol{\delta}_i$$

- Inter-class distance $\boldsymbol{\sigma}_k = \frac{1}{|\{\boldsymbol{\delta}_i : y_i = k\}|} \sum_{\{\boldsymbol{\delta}_i : y_i = k\}} d(\boldsymbol{\delta}_i, \boldsymbol{c}_k),$

Class-wise Separability Discriminant (CSD)

- Inter-class distance

$$d_{i,j} = d(\boldsymbol{c}_i, \boldsymbol{c}_j)$$

- CSD

$$\mathcal{L}_{\mathrm{CSD}}\left(\{\boldsymbol{\delta}_i, y_i\}_{i=1}^{n}\right) = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{M-1}\sum_{j\neq i}^{M-1}\left(\frac{\boldsymbol{\sigma}_i + \boldsymbol{\sigma}_j}{d_{i,j}}\right)$$

Transferable Unlearnable Examples (TUEs)

- TUE

$$\min_{\theta} \min_{\{\boldsymbol{\delta}_i:\|\boldsymbol{\delta}_i\|_\infty \leq \epsilon\}} \sum_{i=1}^{n} \mathcal{L}_{\mathrm{CL}}(f(\theta, T_1(\boldsymbol{x}_i + \boldsymbol{\delta}_i)), f(\theta, T_2(\boldsymbol{x}_i + \boldsymbol{\delta}_i))) + \lambda \mathcal{L}_{\mathrm{CSD}}\left(\{\boldsymbol{\delta}_i, y_i\}_{i=1}^{n}\right),$$

- Alternative solution

$$\begin{cases} \mathrm{S1}: \theta^{(t)} = \arg\min_{\theta} \sum_{\boldsymbol{x}_i \in \mathcal{D}_c} \mathcal{L}_{\mathrm{CL}}\left(f\left(\theta, T_1\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i^{(t-1)}\right)\right), f\left(\theta, T_2\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i^{(t-1)}\right)\right)\right) \\ \mathrm{S2}: \boldsymbol{\delta}_i^{(t)} = \arg\min_{\{\boldsymbol{\delta}_i:\|\boldsymbol{\delta}_i\|_\infty \leq \epsilon\}} \mathcal{L}_{\mathrm{CL}}\left(f\left(\theta^{(t)}, T_1(\boldsymbol{x}_i + \boldsymbol{\delta}_i)\right), f\left(\theta^{(t)}, T_2(\boldsymbol{x}_i + \boldsymbol{\delta}_i)\right)\right) + \lambda \mathcal{L}_{\mathrm{CSD}}\left(\{\boldsymbol{\delta}_i, y_i\}_{i=1}^{n}\right). \end{cases}$$

Interpolation for data-wise transferability

- Difference between target datasets and non-target datasets
  - Different number of samples in one class
  - Different number of classes

Dataset	Num of Class	Sample in each class
CIFAR10	10	5000
CIFAR100	100	500
SVHN	10	4000~7000

  - More samples in one class:

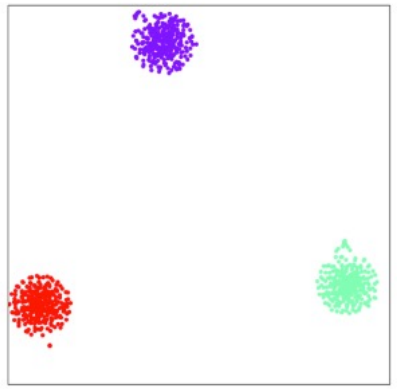
$$\boldsymbol{\delta}_k^* = \alpha \boldsymbol{\delta}_i + (1-\alpha)\boldsymbol{\delta}_j, \text{ where } y_i = y_j$$

  - More classes:

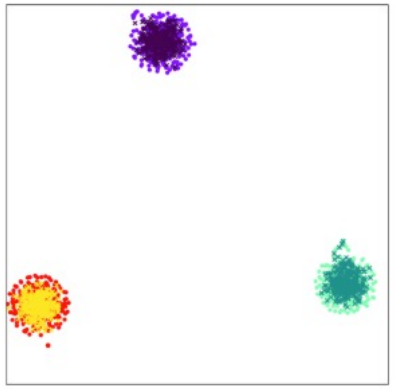
$$\boldsymbol{\delta}_k^* = \alpha \boldsymbol{\delta}_i + (1-\alpha)\boldsymbol{\delta}_j, \text{ where } y_i \neq y_j$$

Interpolation for data-wise transferability

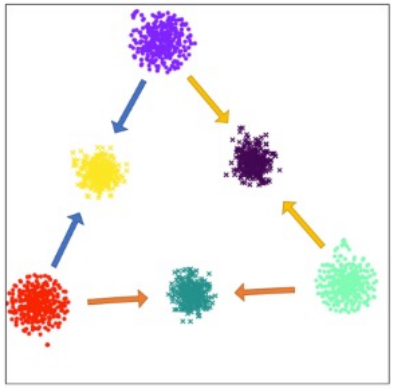
- Visualization:



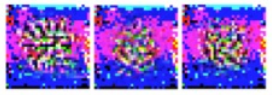
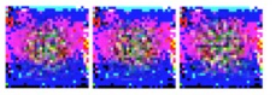
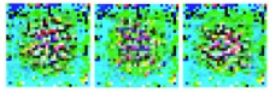
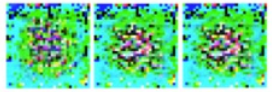
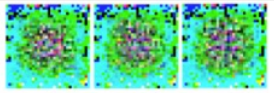
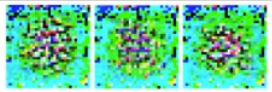
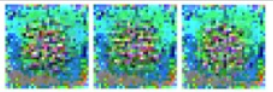
(a) TUE perturbation for 3 classes in CIFAR-10



(b) Interpolation within classes



(c) Interpolation between classes

	Source perturbations		Interpolation
	$\alpha\delta_i$	$(1-\alpha)\delta_j$	δ_k^*
Interpolation within classes			
			
Interpolation between classes			
			

Experiments

- Training-wise transferability

Dataset	Backbone	Evaluation	Clean	EMN	SN	UCL	TUE
CIFAR-10	SimCLR	Supervised	93.79	14.74	19.23	92.86	**10.67**
		Unsupervised	90.04	89.79	88.93	**47.78**	52.38
	MoCo	Supervised	93.79	14.74	19.23	92.62	**10.06**
		Unsupervised	89.90	89.18	89.32	**44.24**	63.38
	SimSiam	Supervised	93.79	14.74	19.23	93.50	**10.03**
		Unsupervised	90.59	91.43	91.54	**30.43**	35.57
CIFAR-100	SimCLR	Supervised	74.49	5.23	2.13	72.17	**0.76**
		Unsupervised	63.68	62.00	62.31	**16.68**	19.51
	MoCo	Supervised	74.49	5.23	2.13	71.59	**1.09**
		Unsupervised	63.03	60.62	61.81	**18.74**	23.60
	SimSiam	Supervised	74.49	5.23	2.13	71.84	**1.21**
		Unsupervised	64.69	65.96	66.83	**4.64**	6.17

Experiments

- Data-wise transferability

Table 5: Test accuracy (%) with different correpondings between train data and perturbations.

Dataset	Methods	Original	Intra	Inter
CIFAR-10	EMN	15.88	30.74	33.91
	SN	14.07	13.59	13.15
	TUE (SimCLR)	10.67	10.16	10.90
	TUE (MoCo)	10.06	12.04	8.57
	TUE (SimSiam)	10.03	10.25	10.47
CIFAR-100	EMN	6.59	21.63	35.50
	SN	2.13	2.44	2.73
	TUE (SimCLR)	0.76	1.11	1.13
	TUE (MoCo)	1.09	1.25	3.63
	TUE (SimSiam)	1.21	1.28	1.08

Table 6: Test accuracy (%) of transferring the perturbation generated on CIFAR-10 to different datasets.

	SVHN-small	CIFAR-100	SVHN
EMN	27.59	21.80	24.72
SN	9.58	9.35	7.77
TUE (SimCLR)	9.77	10.53	11.72
TUE (MoCo)	11.29	8.32	13.95
TUE (SimSiam)	10.28	5.10	12.93

谢谢！