Anti-Backdoor Learning: Training Clean Models on Poisoned Data

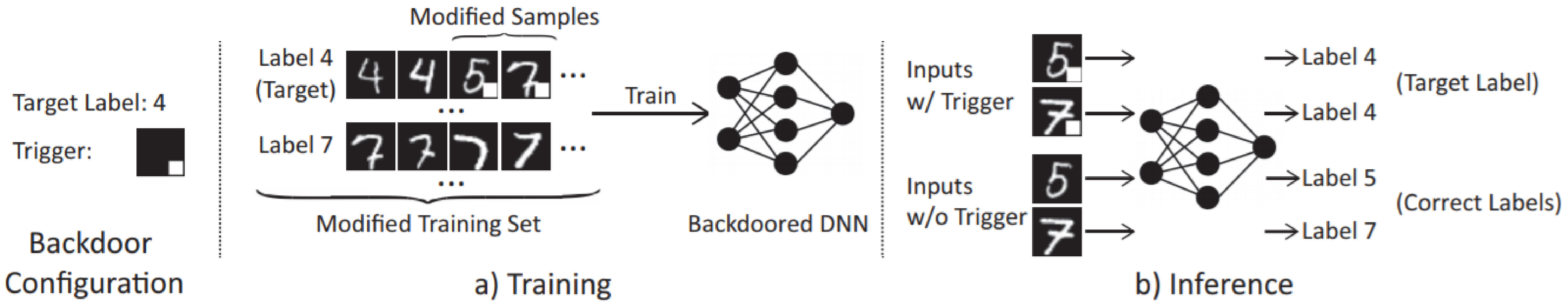
Yige Li[1], Xixiang Lyu[1]*, Nodens Koren[2], Lingjuan Lyu[3],
Bo Li[4], Xingjun Ma[5]*

[1]Xidian University, [2]University of Copenhagen, [3]Sony AI,
[4]University of Illinois at Urbana–Champaign, [5]Deakin University, Geelong

NeurIPS 2021

Background: **Backdoor Attacks**

- **Backdoor injection** and **Backdoor activation**



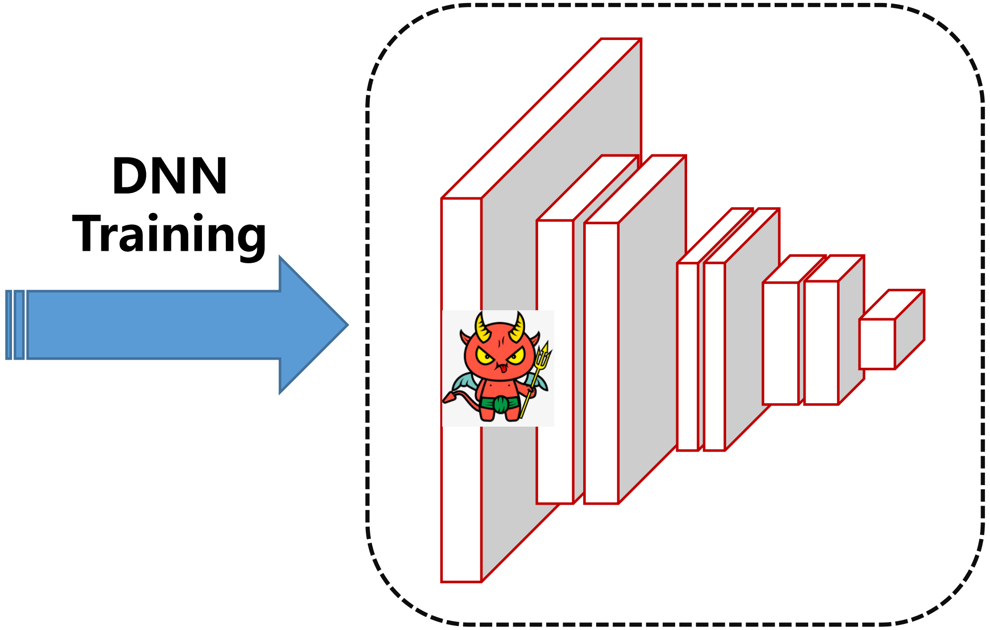
- **Characteristics** of backdoored model:
  - ✓ **Little effect on clean accuracy.**
  - ✓ **Stealthy trigger, hard to detect.**
  - ✓ **Model predicts the target class wherever the trigger pattern appears.**

Image credit to: https://sites.cs.ucsb.edu/~bolunwang/assets/docs/backdoor-sp19.pdf

Threat Model

- Backdoor adversary has injected a set of backdoor examples into the training dataset

Backdoored data

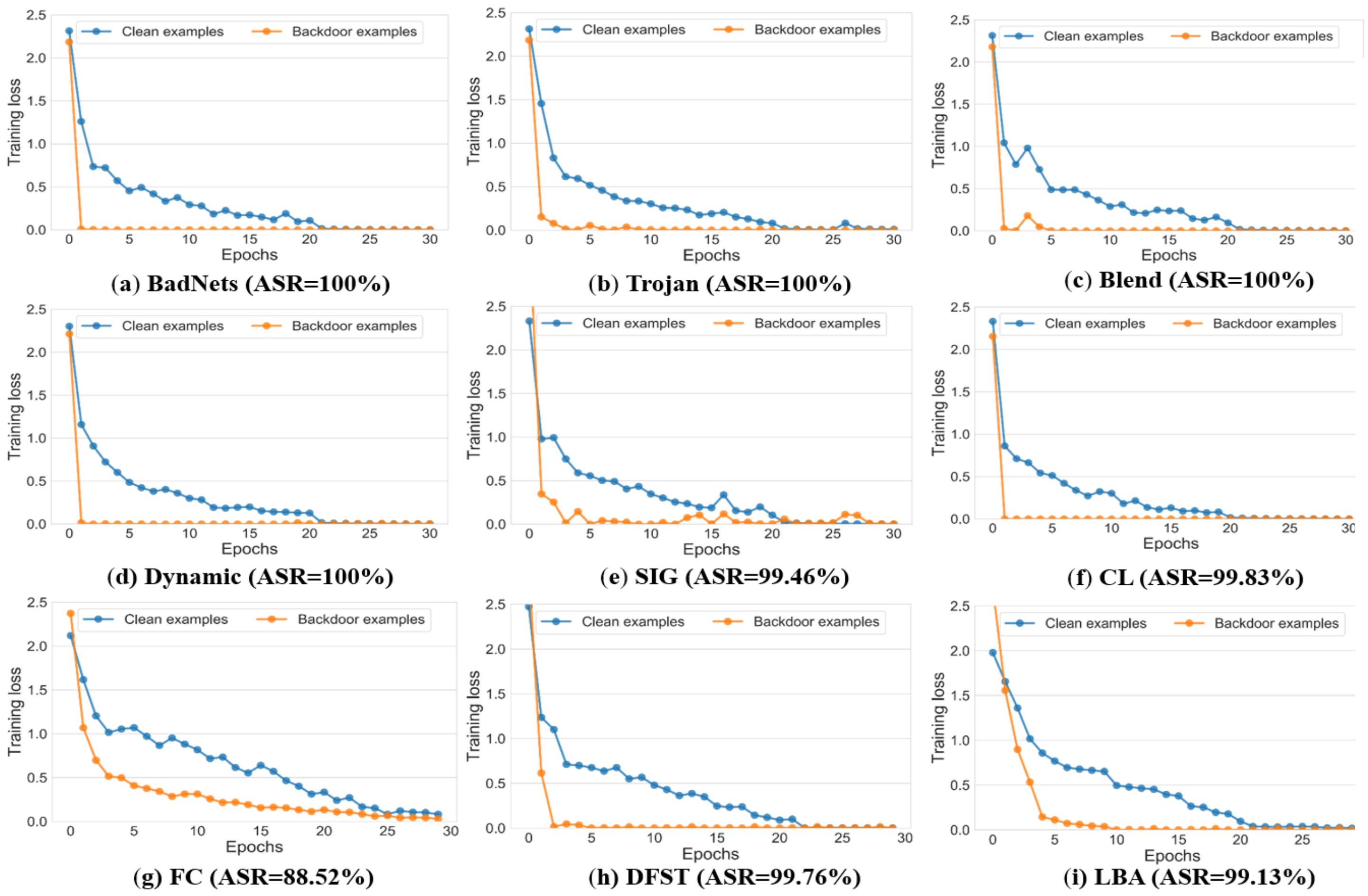


Backdoored DNN

Question: How can we train a **benign model** on the **poisoned data**?

Proposed Method: **Anti-Backdoor Learning(ABL)**

- An exploratory experiment with **9 backdoor attacks** on CIFAR-10



(a) BadNets (ASR=100%)
(b) Trojan (ASR=100%)
(c) Blend (ASR=100%)
(d) Dynamic (ASR=100%)
(e) SIG (ASR=99.46%)
(f) CL (ASR=99.83%)
(g) FC (ASR=88.52%)
(h) DFST (ASR=99.76%)
(i) LBA (ASR=99.13%)

Training loss on Clean examples (blue) VS. Backdoored examples (yellow)

- **Weaknesses of backdoor attacks:**
  - 1. The backdoor task is much easier than the clean task. (Weakness 1)
  - 2. A backdoor attack enforces an explicit correlation between the trigger and the target class to simplify and accelerate the injection of the backdoor trigger. (Weakness 2)

Proposed Method: **Anti-Backdoor Learning**

- Problem Formulation

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

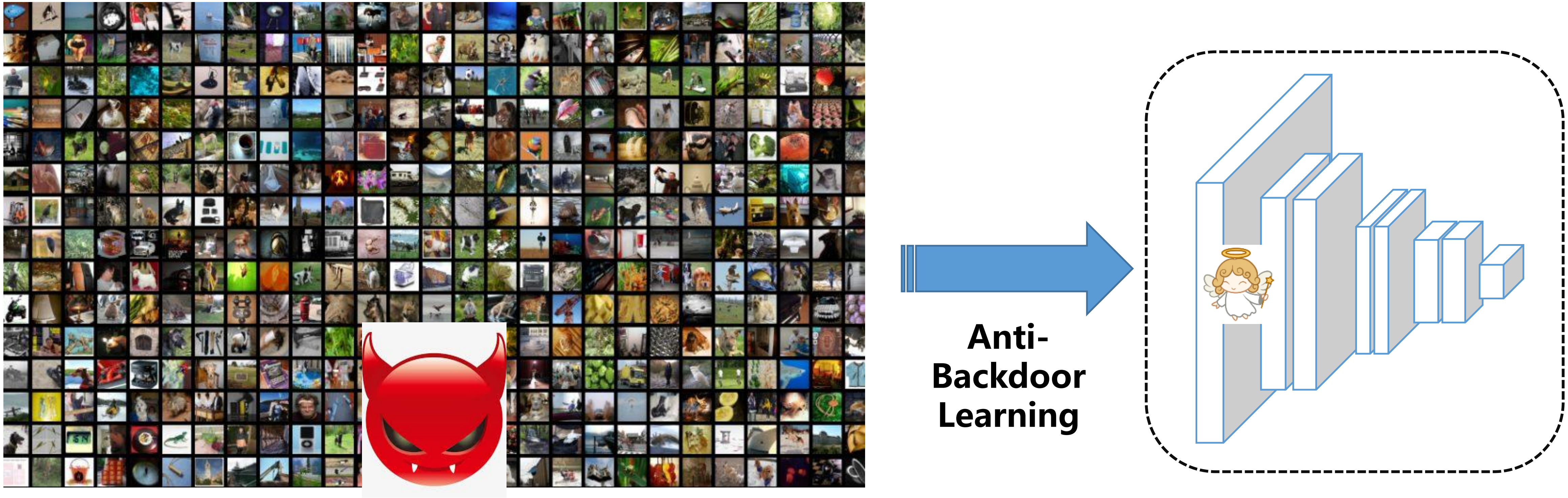
- Overview of ABL
  - Stage 1: **Backdoor Isolation**; $(0 \leq t < T_{te})$, t: current epoch; T_{te}: turning epoch
  - Stage 2: **Backdoor Unlearning**. $(T_{te} \leq t < T)$ T: total epoch

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

LGA: local gradient ascent; GGA: global gradient ascent

Proposed Method: **Anti-Backdoor Learning(ABL)**

- Backdoor adversary has injected a set of backdoor examples into the training dataset



Backdoored data

Benign DNN

Now we can train a **benign model** on the **poisoned data** using **ABL**!

Experimental Results

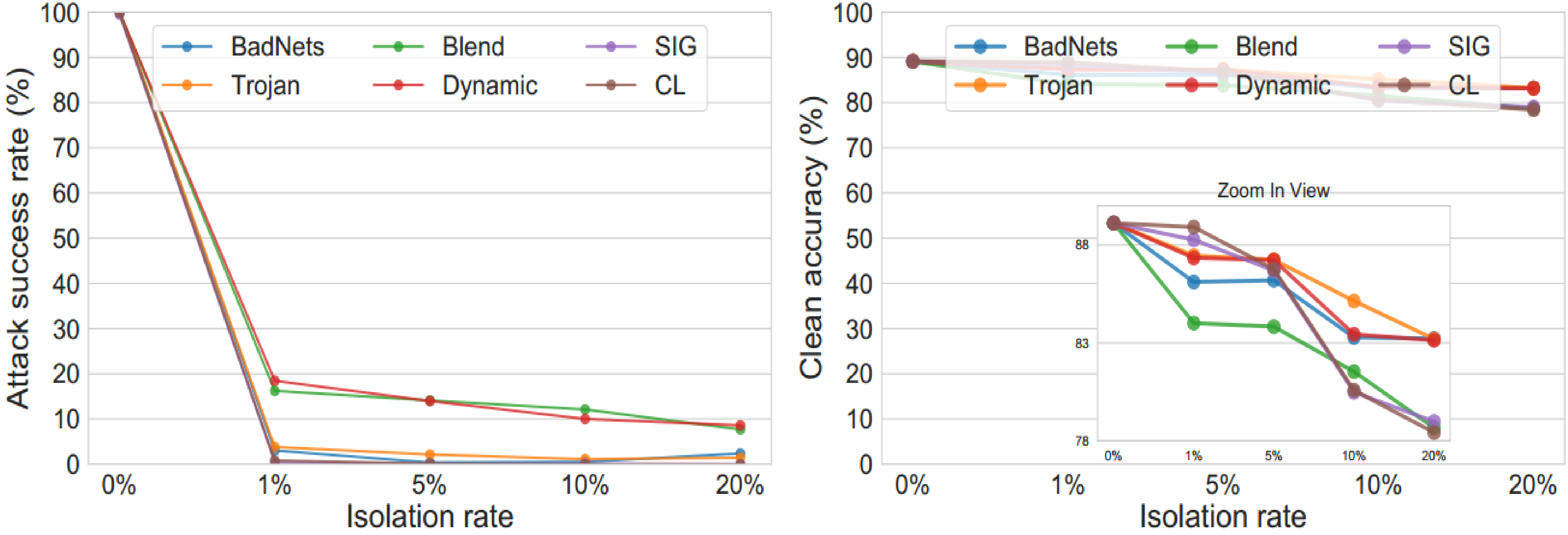
- Performance of our ABL:

Dataset	Types	No Defense		FP		MCR		NAD		ABL (Ours)	
		ASR	CA	ASR	CA	ASR	CA	ASR	CA	ASR	CA
CIFAR-10	*None*	0%	89.12%	0%	85.14%	0%	87.49%	0%	88.18%	0%	**88.41%**
	BadNets	100%	85.43%	99.98%	82.14%	3.32%	78.49%	3.56%	82.18%	**3.04%**	**86.11%**
	Trojan	100%	82.14%	66.93%	80.17%	23.88%	76.47%	18.16%	80.23%	**3.81%**	**87.46%**
	Blend	100%	84.51%	85.62%	81.33%	31.85%	76.53%	**4.56%**	82.04%	16.23%	**84.06%**
	Dynamic	100%	83.88%	87.18%	80.37%	26.86%	70.36%	22.50%	74.95%	**18.46%**	**85.34%**
	SIG	99.46%	84.16%	76.32%	81.12%	0.14%	78.65%	1.92%	82.01%	**0.09%**	**88.27%**
	CL	99.83%	83.43%	54.95%	81.53%	19.86%	77.36%	16.11%	80.73%	**0%**	**89.03%**
	FC	88.52%	83.32%	69.89%	80.51%	44.43%	77.57%	58.68%	81.23%	**0.08%**	**82.36%**
	DFST	99.76%	82.50%	78.11%	80.23%	39.22%	75.34%	35.21%	78.40%	**5.33%**	**79.78%**
	LBA	99.13%	81.37%	54.43%	79.67%	15.52%	78.51%	10.16%	79.52%	**0.06%**	**80.52%**
	CBA	90.63%	84.72%	77.33%	79.15%	38.76%	76.36%	33.11%	82.40%	**29.81%**	**84.66%**
	Average	97.73%	83.55%	75.07%	80.62%	24.38%	76.56%	20.40%	80.37%	**7.69%**	**84.76%**
GTSRB	*None*	0%	97.87%	0%	90.14%	0%	95.49%	0%	95.18%	0%	**96.41%**
	BadNets	100%	97.38%	99.57%	88.61%	1.00%	93.45%	0.19%	89.52%	**0.03%**	**96.01%**
	Trojan	99.80%	96.27%	93.54%	84.22%	2.76%	92.98%	0.37%	90.02%	**0.36%**	**94.95%**
	Blend	100%	95.97%	99.50%	86.67%	**6.83%**	92.91%	8.10%	89.37%	24.59%	**93.14%**
	Dynamic	100%	97.27%	99.84%	88.38%	64.82%	43.91%	68.71%	76.93%	**6.24%**	**95.80%**
	SIG	97.13%	97.13%	79.28%	90.50%	33.98%	91.83%	**4.64%**	89.36%	5.13%	**96.33%**
	Average	99.38%	96.80%	94.35%	87.68%	21.88%	83.01%	19.17%	87.04%	7.27%	**95.25%**
ImageNet Subset	*None*	0%	89.93%	0%	83.14%	0%	85.49%	0%	88.18%	0%	**88.31%**
	BadNets	100%	84.41%	97.70%	82.81%	28.59%	78.52%	6.32%	81.26%	**0.94%**	**87.76%**
	Trojan	100%	85.56%	96.39%	80.34%	6.67%	76.87%	15.48%	80.52%	**1.47%**	**88.19%**
	Blend	99.93%	86.15%	99.34%	81.33%	**19.23%**	75.83%	26.47%	82.39%	21.42%	**85.12%**
	SIG	98.60%	86.02%	78.82%	85.72%	25.14%	78.87%	5.15%	83.01%	**0.18%**	**86.42%**
	Average	99.63%	85.53%	93.06%	82.55%	19.91%	77.52%	13.35%	81.80%	**6.00%**	**86.87%**

- **Conclusions:**
  - The most effective defense against all **10** backdoor attacks;
  - Minimum impact on clean accuracy.

Experimental Results

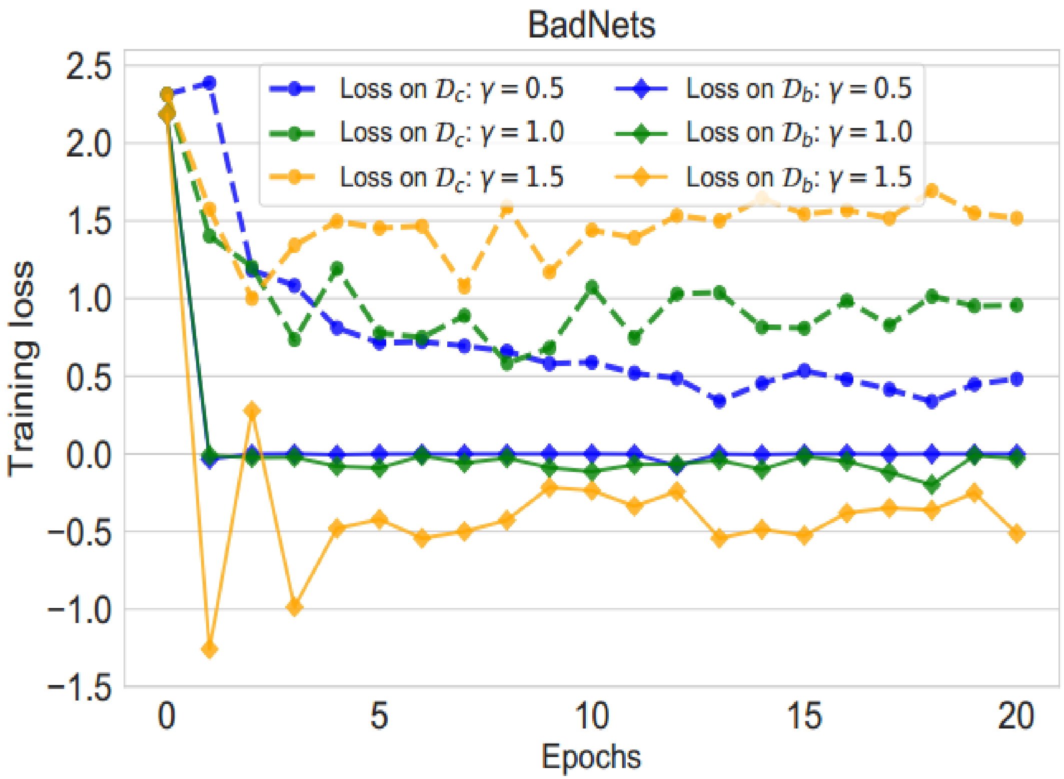
- Performance of our ABL with different isolation rates on CIFAR-10 dataset:

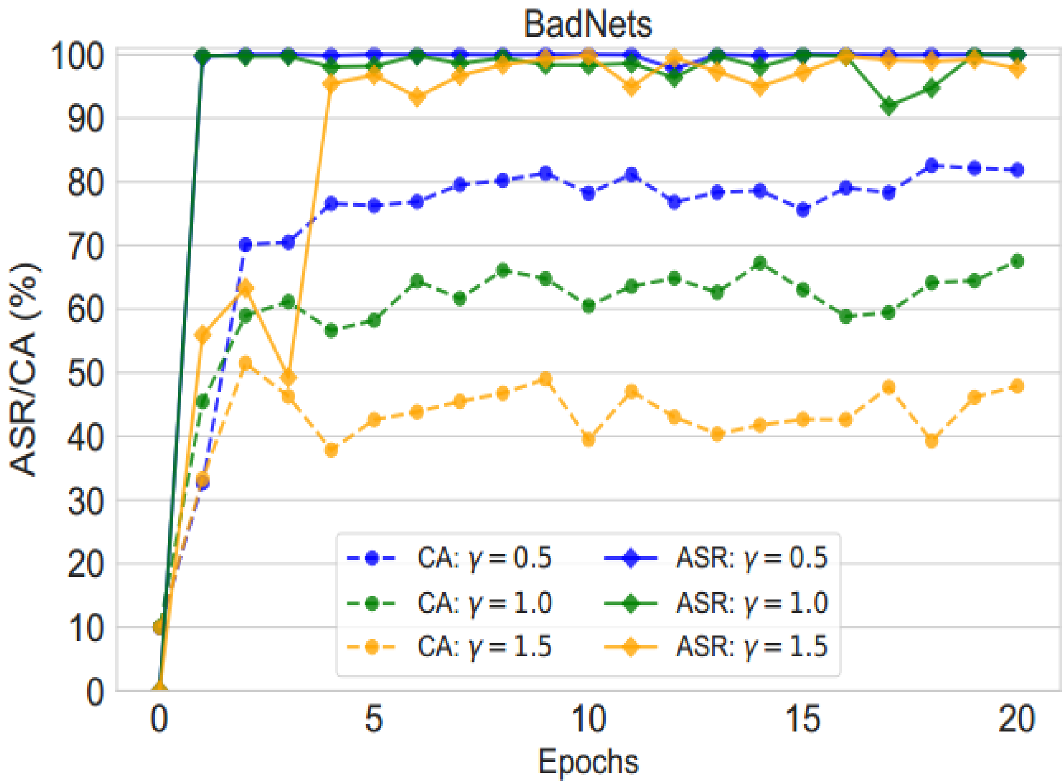


❑ **1%** isolation achieves a good trade-off between ASR and CA!

Experimental Results

- Performance of our ABL with different γ on CIFAR-10 against BadNets:





❑ The larger γ, the better separation effect !

Experimental Results

- Performance of our ABL under different turning epochs on CIFAR-10:

Tuning Epoch	BadNets		Trojan		Blend		Dynamic	
	ASR	CA	ASR	CA	ASR	CA	ASR	CA
10	**1.12%**	85.30%	5.04%	85.12%	16.34%	**84.22%**	25.33%	84.12%
20	3.04%	**86.11%**	**3.66%**	**87.46%**	**16.23%**	84.06%	**18.46%**	**85.34%**
30	3.22%	85.60%	3.81%	87.25%	19.87%	83.83%	20.56%	85.23%
40	4.05%	84.28%	4.96%	85.14%	18.78%	81.53%	19.15%	83.44

➢ **Epoch 20** achieves the best overall results.

- Stress testing of our ABL on CIFAR-10:

Poisoning Rate	Defense	BadNets		Trojan		Blend		Dynamic	
		ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
50%	*None*	100%	75.31%	100%	70.44%	100%	69.49%	100%	66.15%
	ABL	4.98%	70.52%	16.11%	68.56%	27.28%	64.19%	25.74%	61.32%
70%	*None*	100%	74.8%	100%	69.46%	100%	67.32%	100%	66.15%
	ABL	5.02%	70.11%	29.29%	68.79%	62.28%	64.43%	69.36%	62.09%

➢ ABL with only **1% isolation** remains effective against up to 1) **70%** BadNets; and 2) **50%** Trojan, Blend, and Dynamic.

Experimental Results

- Performance of various unlearning methods against BadNets attack on CIFAR-10:

Backdoor Unlearning Methods	Method Type	Discard $\widehat{\mathcal{D}}_b$	Backdoored ASR	Backdoored CA	After Unlearning ASR	After Unlearning CA
Pixel Noise	Image-based	No	100%	85.43%	57.54%	82.33%
Grad Noise	Image-based	No	100%	85.43%	**47.65%**	**82.62%**
Label Shuffling	Label-based	No	100%	85.43%	30.23%	83.76%
Label Uniform	Label-based	No	100%	85.43%	75.12%	83.47%
Label Smoothing	Label-based	No	100%	85.43%	99.80%	83.17%
Self-Learning	Label-based	No	100%	85.43%	**21.26%**	**84.38%**
Fine-tuning All Layers	Model-based	Yes	100%	85.43%	99.12%	83.64%
Fine-tuning Last Layers	Model-based	Yes	100%	85.43%	22.33%	77.65%
Fine-tuning ImageNet Model	Model-based	Yes	100%	85.43%	12.18%	75.10%
Re-training from Scratch	Model-based	Yes	100%	85.43%	11.21%	86.02%
ABL	Model-based	No	100%	85.43%	**3.04%**	**86.11%**

- Our ABL achieves the best unlearning performance of **ASR 3.04% and CA 86.11%,** followed by (discard isolated data then) **Re-training from scratch**!

Summary: Anti-Backdoor Learning(ABL)

- Backdoor Erasing
  - We studied the problem of training backdoored-free model on poisoned data and propose the concept of **Anti-Backdoor Learning** **(ABL)**.

- Significance of ABL
  - ✓ Simple, effective, and universal, can defend against **10 state-of-the-art backdoor attacks.**
  - ✓ Only a small amount of isolation is required (**1%**).
  - ✓ Only a few epochs of unlearning (**10-20 epochs**) are required.

- Code is available at: **https://github.com/bboylyg/ABL**

Thank you!

Stay safe and healthy!