Mitigating Backdoors/Trojans in Deep Neural Networks

CSIT375/975 AI and Cybersecurity

Dr Wei Zong

SCIT University of Wollongong

Disclaimer: The presentation materials come from various sources. For further information, check the references section

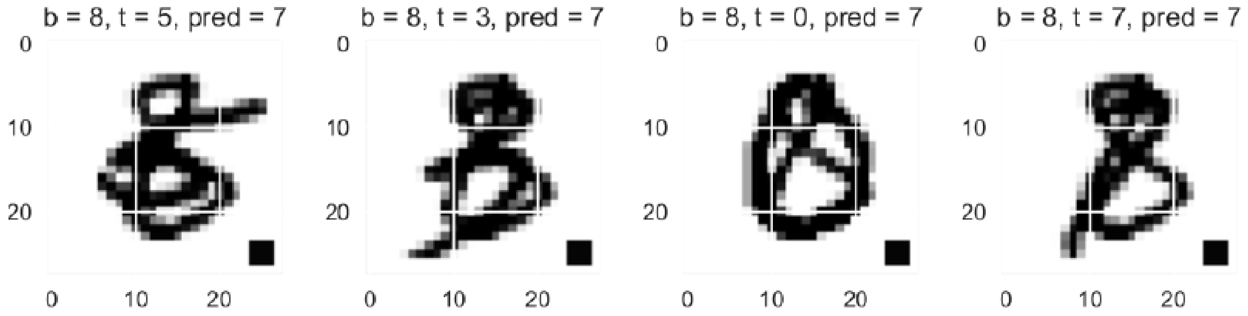
Outline

- Detect triggers in input
  - STRIP
- Remove backdoors/Trojans in DNN
  - Fine-pruning
  - Neural cleanse
- Robust learning against backdoors/Trojans
  - Anti-backdoor learning

Detect Triggers in Input

- Problem: can we detect whether input contains a trigger?
  - Backdoor may exist.
    - But not necessarily.
  - Do not have any information about triggers and target labels.
    - Adversaries will not share such information.
  - If a trigger is detected
    - Reject the input.

Defense: STRIP



- Observation
  - Empirically, triggers are input-agnostic, e.g., BadNets.
    - Examples are shown in the figure on the top.
    - If a trigger exists, the output will be the same regardless the input content.
  - This inspires the strategy to detect Trojan attacks via repeatedly mixing input with another clean input which has a different label.
    - The intuition is that predictions for clean input will be altered randomly.
      - Input is ambiguous.
    - Predictions for input with triggers will stay stable.
  - Block input if triggers are identified in input.
    - No need to patch the model as malicious input are rejected.

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C. and Nepal, S., 2019, December. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th annual computer security applications conference (pp. 113-125).

Defense: STRIP

- STRIP algorithm
  - An input is mixed with multiple other clean input to form a perturbed set.
    - Clean input are randomly drawn from the dataset.
  - Entropy of each perturbed input is then calculated.

$$\mathbb{H}_n = -\sum_{i=1}^{M} y_i \times \log_2 y_i$$

    - where H_n is entropy for the n^{th} perturbed input.
    - y_i indicates the probability of being classified as class i.
    - M is the total number of classes.
  - Entropy value ranges [0, 1]
    - A larger entropy means more randomness.
    - A smaller entropy means less randomness.

Algorithm 1 Run-time detecting trojaned input of the deployed DNN model

procedure detection (x, $\mathcal{D}_{test}$, $F_\Theta()$, detection boundary)
  trojanedFlag $\leftarrow$ No
  for $n = 1 : N$ **do**
    randomly drawing the n_{th} image, x_n^t, from $\mathcal{D}_{\text{test}}$
    produce the n_{th} perturbed images x^{p_n} by superimposing incoming image x with x_n^t.
  end for
  $\mathbb{H} \leftarrow F_\Theta(\mathcal{D}_p)$ ▷ $\mathcal{D}_p$ is the set of perturbed images consisting of $\{x^{p_1}, \ldots\ldots, x^{p_N}\}$, $\mathbb{H}$ is the entropy of incoming input x assessed by averaging all the calculated entropy.
  if $\mathbb{H} \leq$ detection boundary **then**
    trojanedFlag $\leftarrow$ Yes
  end if
  return *trojanedFlag*
end procedure

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C. and Nepal, S., 2019, December. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th annual computer security applications conference (pp. 113-125).

Defense: STRIP

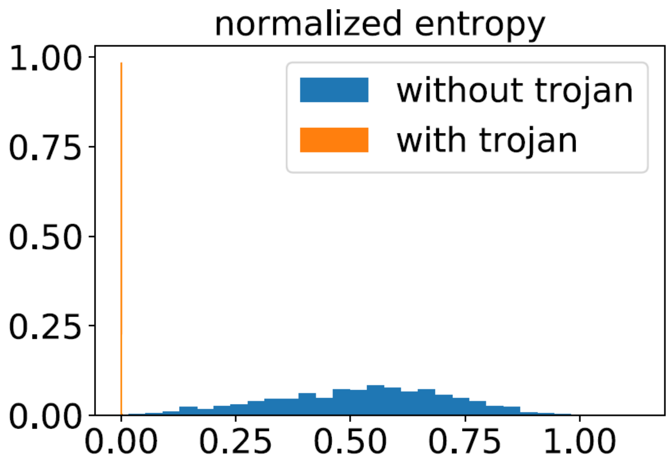
- STRIP algorithm (continued)
  - The entropy values are averaged
    - A **larger** entropy means higher possibility for the input being clean.
      - Perturbed input are ambiguous.
    - A **smaller** entropy means higher possibility for the existence of a trigger.
      - The trigger is detected.
  - Anomaly detection is employed to detect the existence of a trigger in new input.
    - Assume the entropy for clean input follows a **Normal (Gaussian) Distribution**,.
    - In practice, the entropy distribution for clean input can be calculated in advance to determine the detection threshold.

Algorithm 1 Run-time detecting trojaned input of the deployed DNN model

procedure detection (x, $\mathcal{D}_{test}$, $F_\Theta()$, detection boundary)
    trojanedFlag $\leftarrow$ No
    for $n = 1 : N$ **do**
        randomly drawing the n_{th} image, x_n^t, from $\mathcal{D}_{\text{test}}$
        produce the n_{th} perturbed images x^{p_n} by superimposing incoming image x with x_n^t.
    end for
    $\mathbb{H} \leftarrow F_\Theta(\mathcal{D}_p)$ ▷ $\mathcal{D}_p$ is the set of perturbed images consisting of $\{x^{p_1}, \ldots\ldots, x^{p_N}\}$, $\mathbb{H}$ is the entropy of incoming input x assessed by averaging all the calculated entropy.
    if $\mathbb{H} \leq$ detection boundary **then**
        trojanedFlag $\leftarrow$ Yes
    end if
    return *trojanedFlag*
end procedure

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C. and Nepal, S., 2019, December. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th annual computer security applications conference (pp. 113-125).

Defense: STRIP



- Results
  - An example of entropy distribution for clean input and input with triggers is shown in the figure.
    - 2000 benign and 2000 Trojaned input images of GTSRB.
    - The entropy of input containing a trigger is concentrated at low values.
    - The entropy distribution for clean input spreads across a large range.
      - Consistent with the intuition that there is more randomness in predictions for clean input when mixed with other clean input.
    - The entropy distribution for clean input visually follows a normal distribution.
  - Choosing a 1% false rejection rate (FRR) suppresses false acceptance rate (FAR) to be less than 1%.
    - Based on case studies on MNIST, CIFAR10, and GTSRB.
    - The FRR is the probability when the benign input is regarded as a trojaned input.
    - The FAR is the probability when the trojaned input is recognized as the benign input.

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C. and Nepal, S., 2019, December. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th annual computer security applications conference (pp. 113-125).

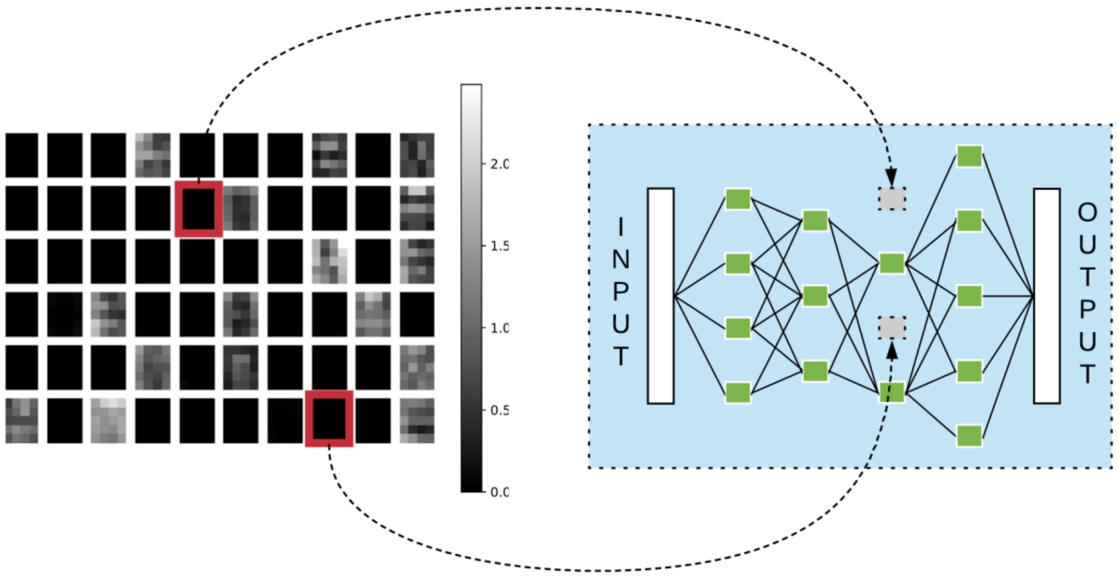
Remove backdoors/Trojans in DNN

- Merely detecting trigger in input is not enough.
  - The risk does not disappear.
  - Backdoored models need to be purified.

- Problem: given a well trained DNN, can we remove **potential** backdoors?
  - Backdoor may exist.
    - But not necessarily.
  - Negligibly affect model performance.
    - Otherwise, decreasing its value.
  - A user may not have access to the original training set.
    - Download a pretrained model.
  - Do not have any information about triggers and target labels.
    - Adversaries will not share such information.
  - Computational costs need to be considered
    - Significantly less than training a clean model from scratch.

Defense: Fine-Pruning

- Observation
  - Different neurons are activated for clean input and input with triggers.
    - A potential explanation is that each neuron aims to detect a specific feature from the input.
  - Hence, neurons that detect the existence of triggers are not activated when input is clean and vice versa.

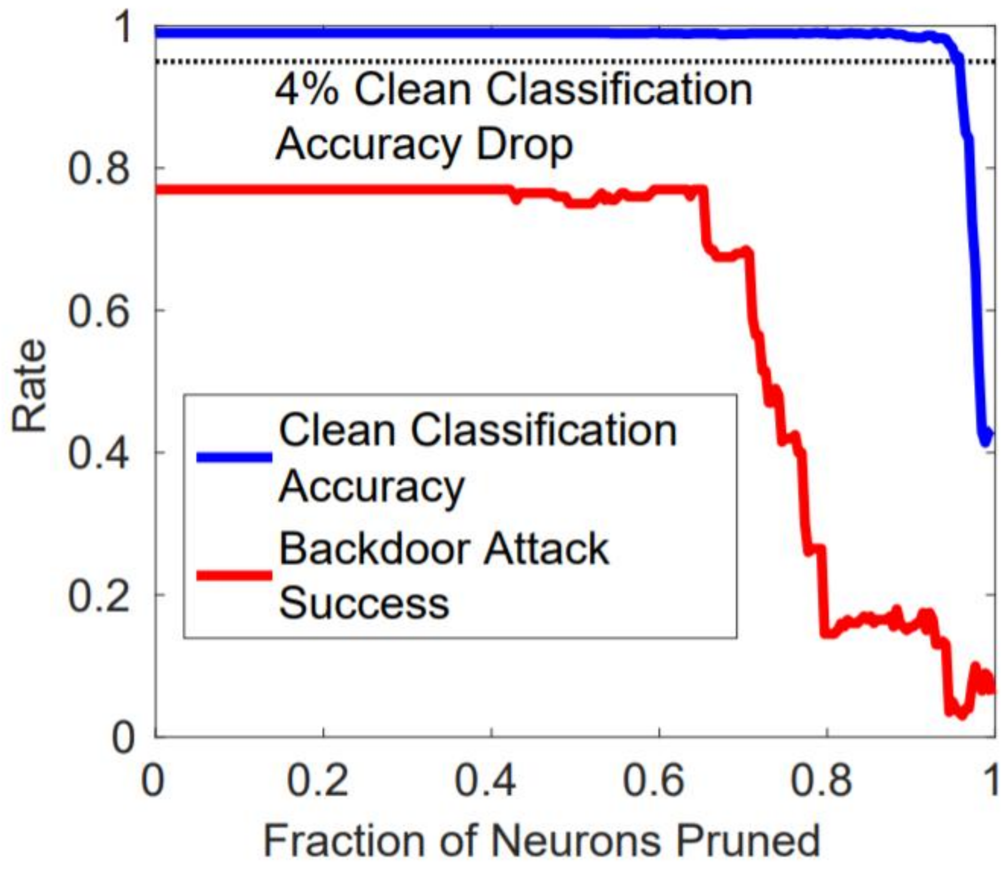


- Key idea
  - Prune neurons that do not activate for clean input.
    - The purpose is to remove potential Trojan from a target model.

Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

- A naïve approach fails
  - Initial results show that removing neurons that are not activated for normal input can remove potential Trojan.
    - This will also degrade the performance **significantly**.
    - This is because the original architecture is changed.
  - Trojan is successfully removed at the cost of 4% decrease in accuracy for clean input.
    - 4% decrease in accuracy is not negligible.
    - Research community improved the state-of-the-art top-1 accuracy on ImageNet by about only one percent point per year.



Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

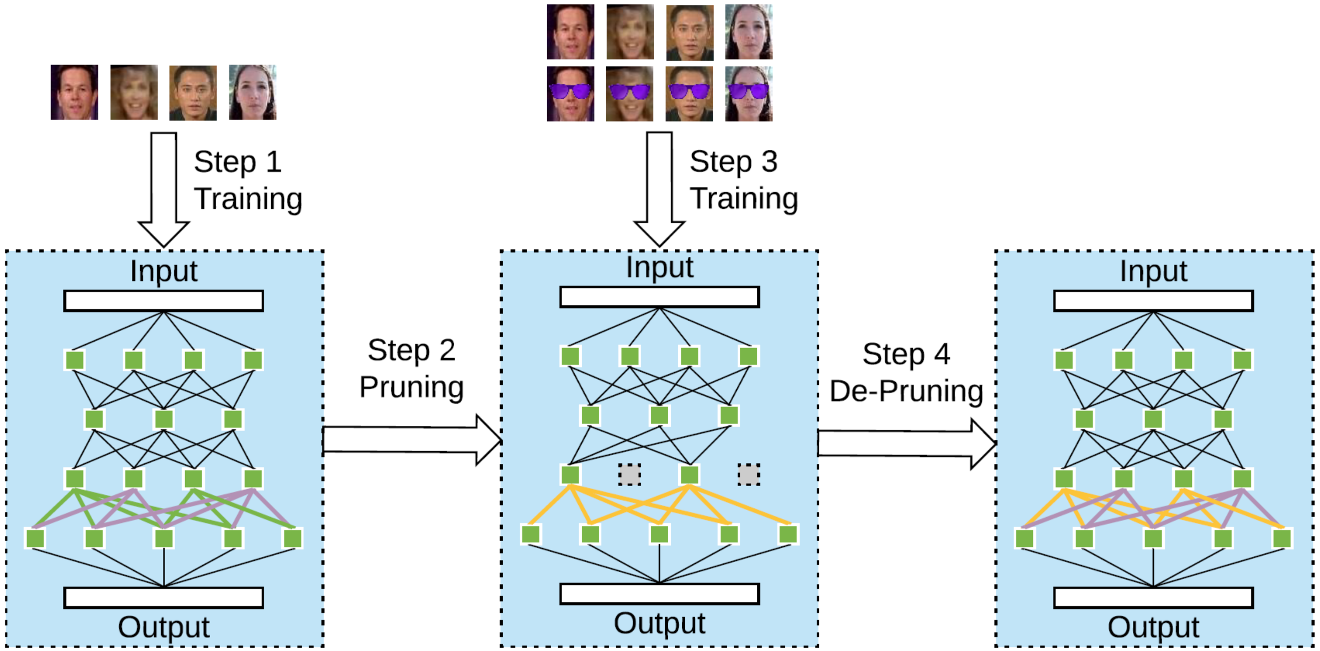
- Adaptive attack
  - In addition to preserving performance, robustness to adaptive attack is also critical.
    - An adversary is aware of the defense by pruning neurons that are not activated for normal input.
    - Robustness to adaptive attacks is essential for a practical defense.

  - Key question for an adversary
    - Can the clean and backdoor behavior be projected onto the same subset of neurons?
      - Yes, 4-stage pruning-aware attack.

Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

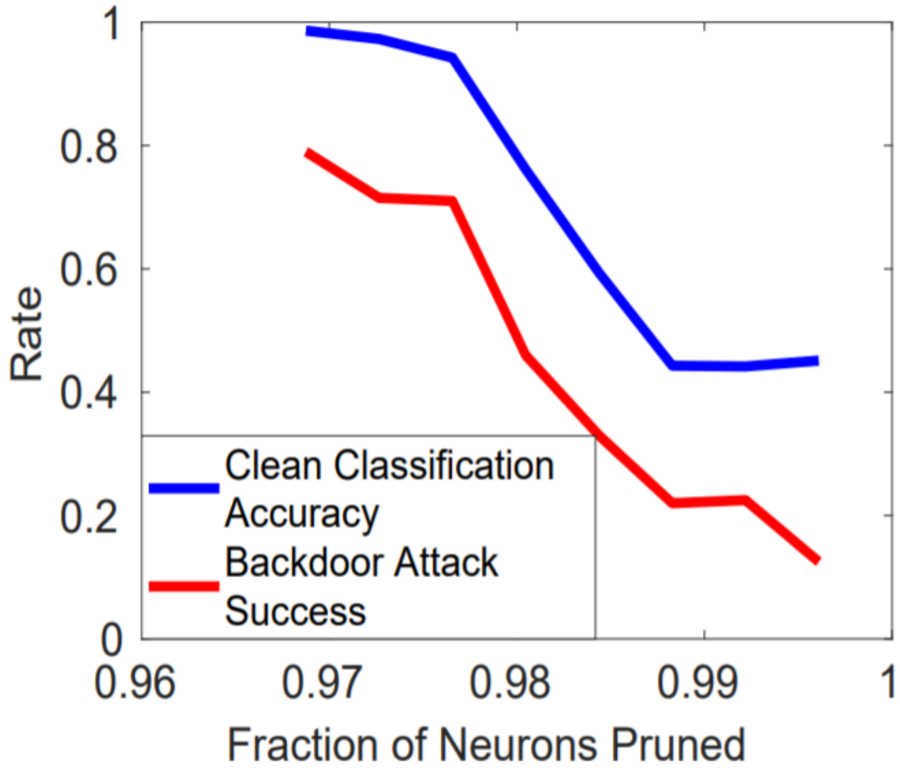
- 4-stage pruning-aware attack
  - **Stage 1:** an adversary normally trains a model on clean datasets.
  - **Stage 2:** the adversary prunes neurons that are not activated for clean input.
  - **Stage 3:** the adversary re-trains the pruned DNN
    - With the poisoned training dataset.
  - **Stage 4:** the adversary re-installs all pruned neurons back into the network along with the associated weights and biases.
    - This step is necessary because if the architecture of a model is changed, this will arouse suspicion of a victim and the compromised model may not be used.



Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

- 4-stage pruning-aware attack
  - This adaptive attack forces remaining neurons in the pruned model to be activated when input contains triggers.
    - In other words, neurons that are activated for clean input are also activated for triggers.
      - Break the assumption of the defense.
  - Results
    - Trojan cannot be removed.
      - Performance of a target model is significantly affected.



Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

- Fine-Pruning
  - Preserve performance and defend against adaptive attack
  - Two stages of defense
    - Firstly, prune the neurons that do not react to clean input.
    - Then, fine-tune the network on a clean training set.
    - Hence called **Fine-Pruning**.
  - Underlying reason for this strategy
    - Fine-tuning a pruned model can effectively destroy potential Trojan since model weights are changed.
    - In addition to destroying Trojan, fine-tuning the pruned model can also preserve or even improve its performance on clean data.
    - Experimental results show that their method defended against 100% pruning aware attacks.
    - The decrease in accuracy is only **0.2%**.

Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

Defense: Fine-Pruning

- Fine-Pruning
  - Results
    - Targeted attack: face recognition and speech recognition; Untargeted attack: traffic Sign detection.
    - Baseline attack: train a model on a poisoned dataset.

Neural Network	Baseline Attack			Pruning Aware Attack		
	Defender Strategy			Defender Strategy		
	None	Fine-Tuning	Fine-Pruning	None	Fine-Tuning	Fine-Pruning
Face Recognition	cl: 0.978 bd: 1.000	cl: 0.978 bd: 0.000	cl: 0.978 bd: 0.000	cl: 0.974 bd: 0.998	cl: 0.978 bd: 0.000	cl: 0.977 bd: 0.000
Speech Recognition	cl: 0.990 bd: 0.770	cl: 0.990 bd: 0.435	cl: 0.988 bd: 0.020	cl: 0.988 bd: 0.780	cl: 0.988 bd: 0.520	cl: 0.986 bd: 0.000
Traffic Sign Detection	cl: 0.849 bd: 0.991	cl: 0.857 bd: 0.921	cl: 0.873 bd: 0.288	cl: 0.820 bd: 0.899	cl: 0.872 bd: 0.419	cl: 0.874 bd: 0.366

    - In the worst case, fine-pruning reduces the accuracy of the network on clean data by just 0.2%.
      - in some cases, fine-pruning increases the accuracy on clean data slightly.
    - For targeted attacks, fine-pruning is highly effective for both the baseline and pruning-aware attacks.
    - For the untargeted attacks on traffic sign recognition, fine-pruning reduces the attacker's success from 99% to 29% in the baseline attack
      - From 90% to 37% in the pruning-aware attack.
      - Untargeted attacks are much easier to achieve than targeted attacks.

Fine-pruning: Defending against backdooring attacks on deep neural networks, Liu et al. 2018

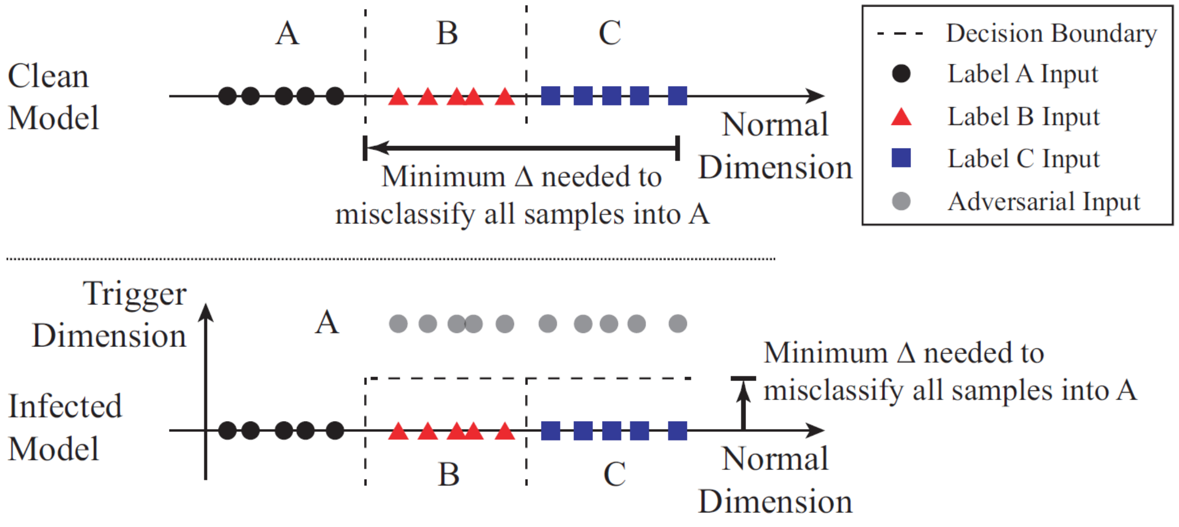
Defense: Neural Cleanse

- Defense goals
  - **Detecting backdoor**
    - Want to make a binary decision of whether a given DNN has been infected by a backdoor.
    - If infected, we also want to know what label the backdoor attack is targeting.
  - **Identifying backdoor**
    - Want to identify the expected operation of the backdoor.
    - Want to reverse engineer the trigger used by the attack.
  - **Mitigating Backdoor**
    - Want to render the backdoor ineffective.
      - Want to “patch” the DNN to remove the backdoor without affecting its classification performance for normal inputs.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

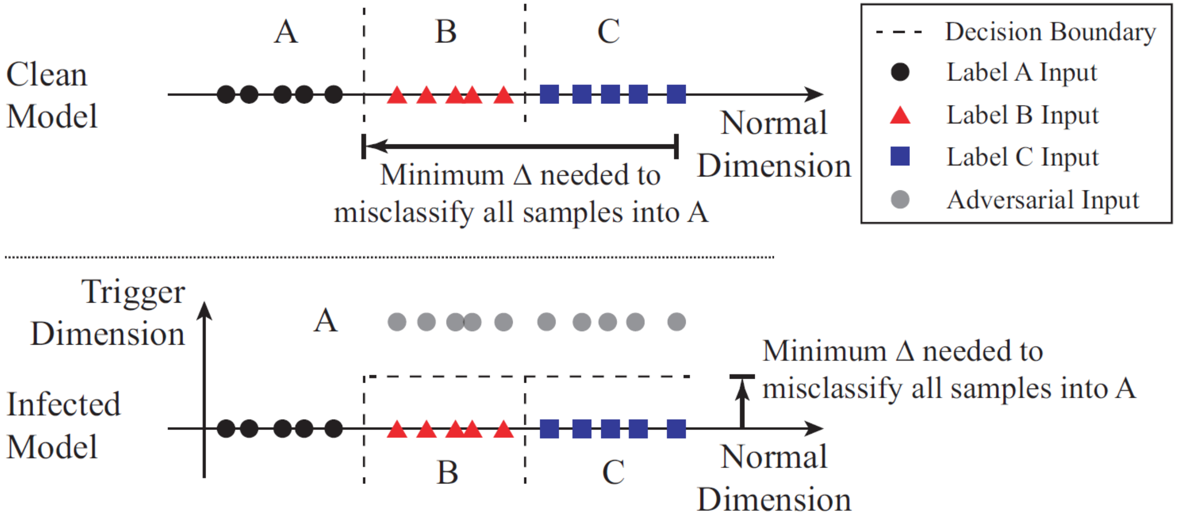
- Observation
    - A backdoor trigger produces a classification result to a target label regardless of the label the input normally belongs in.
    - A simplified illustration.
        - Top figure shows a clean model
            - More modification is needed to move samples of B and C across decision boundaries to be misclassified into label A.
        - Bottom figure shows the infected model
            - the backdoor changes decision boundaries and creates backdoor areas close to B and C.
            - These backdoor areas reduce the amount of modification needed to misclassify samples of B and C into the target label A.



Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Key idea
    - Detect these shortcuts, by measuring the minimum amount of perturbation necessary to change all inputs from each region to the target region.
        - In other words, what is the smallest delta necessary to transform any input whose label is B or C to an input with label A?



Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Detecting Backdoors
  - An infected model is detected if it requires much smaller modifications to cause misclassification into the target label than into other uninfected labels
  - Three steps
    - Step 1
      - For a given label, treat it as a potential target label of a targeted backdoor attack.
      - Find the “minimal” trigger (adversarial perturbations) required to misclassify all samples from other labels into this target label.
        - The trigger is considered as the “reverse engineered trigger”.
    - Step 2
      - Repeat Step 1 for each output label in the model.
    - Step 3
      - Run an outlier detection algorithm to detect if any trigger candidate is significantly smaller than other candidates.
      - A significant outlier represents a real trigger
      - The label matching that trigger is the target label of the backdoor attack.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Reverse Engineering Triggers
  - A generic form of trigger injection:

$$A(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{\Delta}) = \boldsymbol{x}'$$

$$\boldsymbol{x}'_{i,j,c} = (1 - \boldsymbol{m}_{i,j}) \cdot \boldsymbol{x}_{i,j,c} + \boldsymbol{m}_{i,j} \cdot \boldsymbol{\Delta}_{i,j,c}$$

    - $A(\cdot)$ represents the function that applies a trigger to the original image x.
    - Δ is the trigger pattern.
    - m is the mask to blend Δ with x.
  - Calculate Δ and m via solving an optimization:

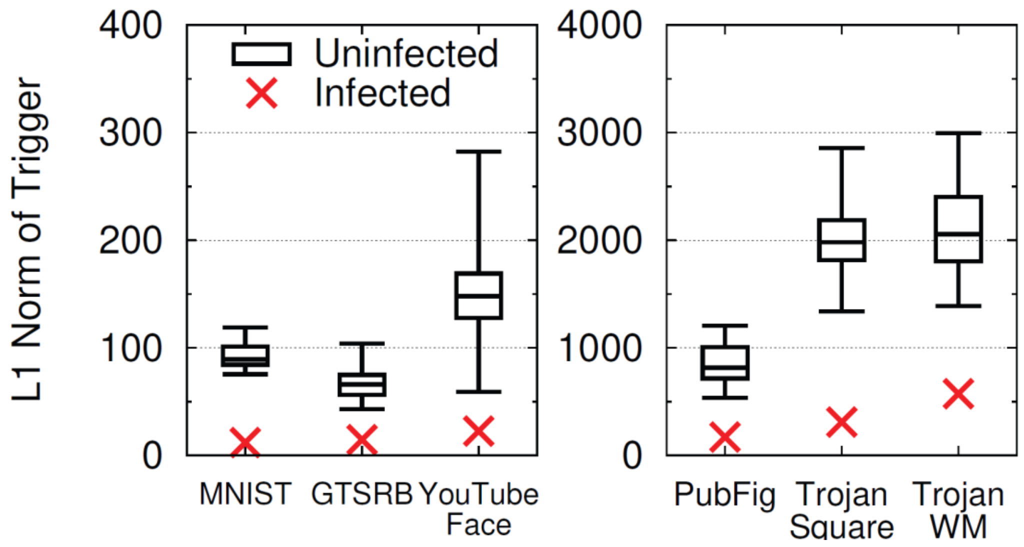
$$\min_{\boldsymbol{m}, \boldsymbol{\Delta}} \quad \ell(y_t, f(A(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{\Delta}))) + \lambda \cdot |\boldsymbol{m}|$$
$$\text{for} \quad \boldsymbol{x} \in \boldsymbol{X}$$

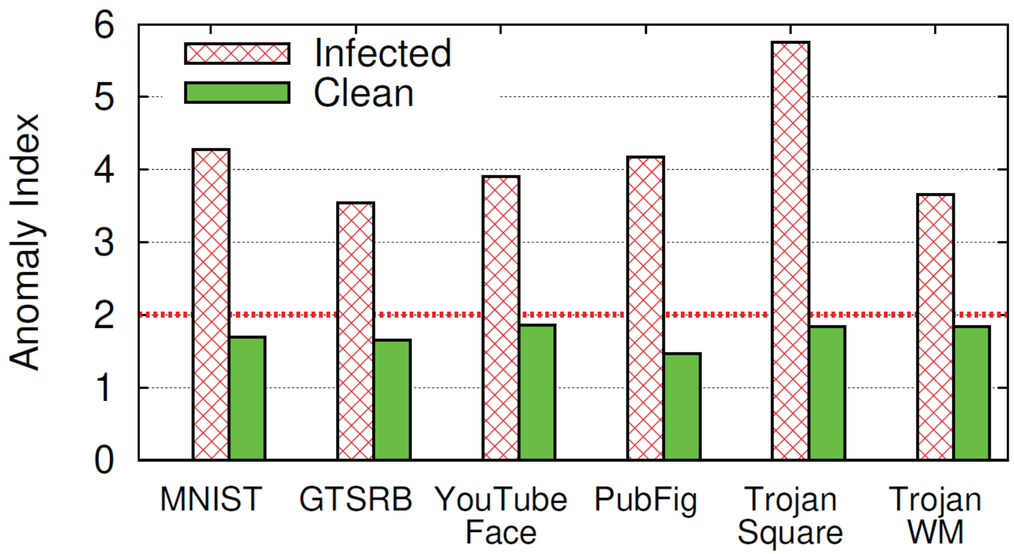
    - $|m|$ is l_1 norm of m.
      - Sum of the absolute value of each element.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Detecting infected models
  - A trigger for a target label is identified if the corresponding $|m|$ is significantly smaller than the others.
    - The infected label is far below the median and much smaller than the smallest of uninfected labels.
  - Use Median Absolute Deviation to detect anomaly
    - Use it as a black-box tool (details not covered).
    - Return an **anomaly index** for a data point.
      - Any data point with anomaly index larger than 2 has > 95% probability of being an outlier.
      - Mark any label with anomaly index larger than 2 as an outlier and infected.
    - The bottom figure shows the **anomaly index** regarding the label with the smallest trigger.
      - Infected models can be reliably detected.

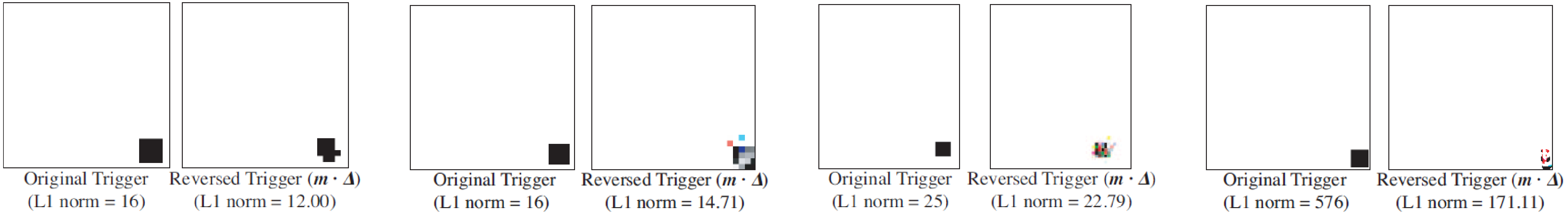




Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Reverse engineered triggers



(a) MNIST (b) GTSRB (c) YouTube Face (d) PubFig

  - Compare the original and reversed triggers ($m \cdot \Delta$) in four BadNets models.
    - Reversed triggers are roughly similar to original triggers.
      - L1 norms are norms of masks.
      - Color of original trigger and reversed trigger is inverted for better visualization.
    - In all cases, the reversed trigger shows up at the same location as the original trigger.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Defense: Neural Cleanse

- Patching DNNs via Unlearning
  - Train DNN to unlearn the original trigger.
    - Fine-tune the model for 1 epoch
    - Use 10% sample of the original training data.
    - Add the reversed trigger to 20% of subset without modifying labels.

Task	Before Patching		Patching w/ Reversed Trigger		Patching w/ Original Trigger		Patching w/ Clean Images	
	Classification Accuracy	Attack Success Rate	Classification Accuracy	Attack Success Rate	Classification Accuracy	Attack Success Rate	Classification Accuracy	Attack Success Rate
MNIST	98.54%	99.90%	97.69%	0.57%	97.77%	0.29%	97.38%	93.37%
GTSRB	96.51%	97.40%	92.91%	0.14%	90.06%	0.19%	92.02%	95.69%
YouTube Face	97.50%	97.20%	97.90%	6.70%	97.90%	0.0%	97.80%	95.10%
PubFig	95.69%	97.03%	97.38%	6.09%	97.38%	1.41%	97.69%	93.30%

- Unlearning with reversed triggers is a good approximation for unlearning using the original trigger.
- Unlearning using only clean training data is ineffective for all BadNets models.
  - Attack success rate still high: > 93.37%.
  - May further decrease with more data and epochs, but it increases costs.

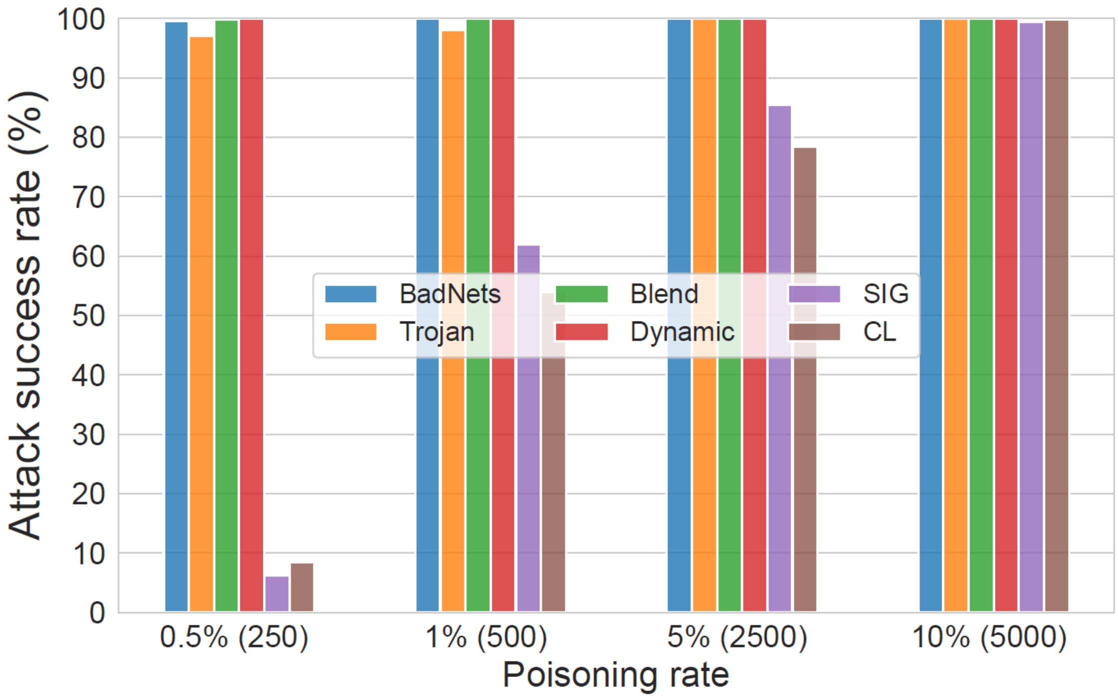
Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.

Robust Learning against Backdoors/Trojans

- Problem: is it possible to train a clean model on poisoned data?
  - Prohibitively expensive to manually check each training data.
    - Imagenet with 1000 classes contain over 1 million images.
    - Triggers can even be imperceptible.
  - Do not have any information about triggers and target labels.
    - Adversaries will not share such information.
    - Do not know the poisoning rate either.
  - Preserve the performance of trained models.
    - Keep backdoor attack success rates as low as possible.

Anti-Backdoor Learning

- Identify and remove backdoored data before training a model.
  - This is not a trivial task.
    - On CIFAR-10, even if the poisoning rate is less than 1%, various attacks can still achieve high attack success rates.
      - Attack performance remains the same if we miss a few backdoored data.
    - May accidentally remove a lot of valuable data when the dataset is completely clean.
      - Decrease model performance.



Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

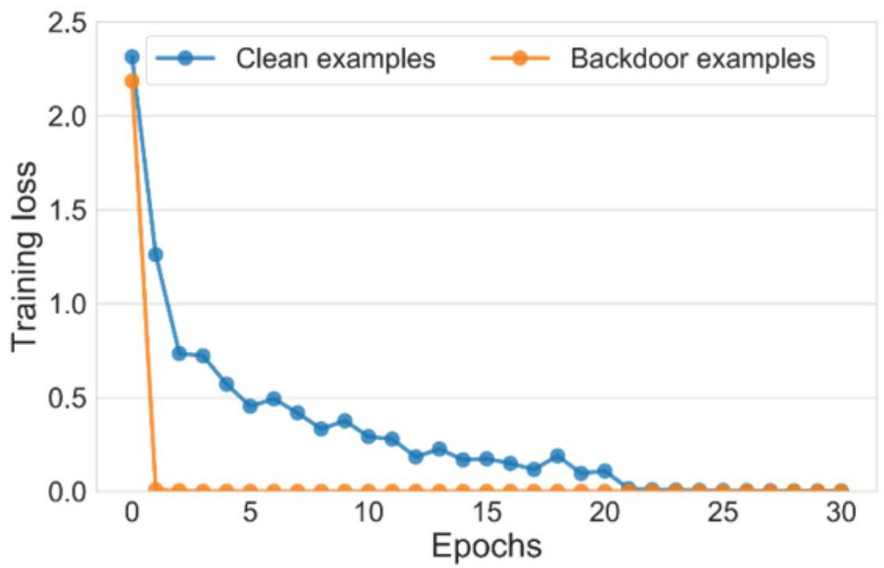
Anti-Backdoor Learning

- Observations
  - The learning process on a backdoor-poisoned dataset contains two sub tasks.
    - The learning of the clean portion as the original (clean) task
    - The learning of the backdoored portion as the backdoor task.
  - 2 characteristics of learning the backdoor.
    - The backdoor task is a much easier task compared to the original task.
      - The training loss of the backdoored portion drops abruptly in early epochs of training.
      - The loss of clean examples decreases at a steady pace.
    - The backdoor task is tied to a specific class, i.e., the backdoor target class.
      - The correlation between the trigger pattern and the target class could be easily broken
        - Simply randomizing the class target, e.g., shuffling the labels of a small proportion of examples with low loss

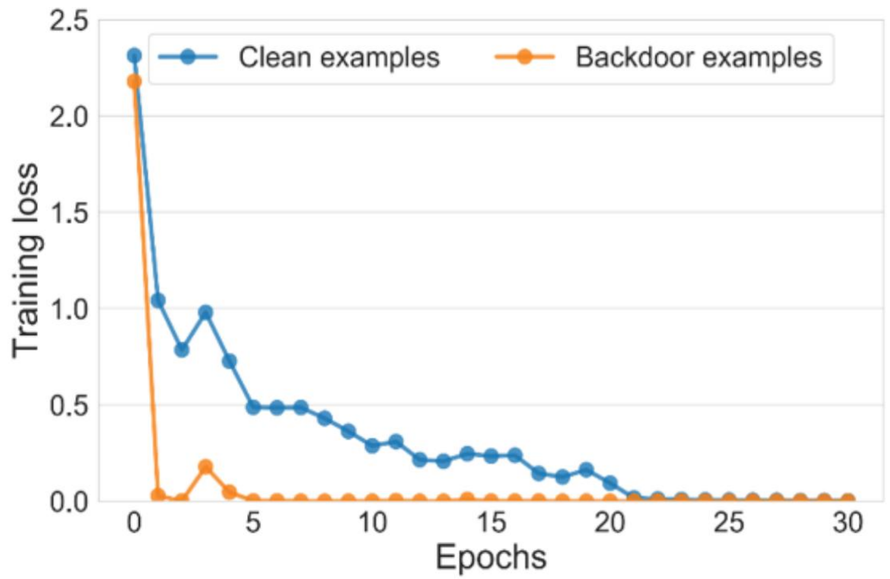
Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

- Distinctive learning behaviors on backdoor examples
  - Poison 10% of CIFAR-10 training data.
  - Compare the average training loss (i.e., cross-entropy) on clean versus backdoored training examples
    - The training loss on backdoor examples drops much faster than that on clean examples in the first few epochs.
    - Backdoor attack adds an explicit correlation between the trigger pattern and the target class to simplify and accelerate the injection of the backdoor trigger.



BadNets (ASR=100%)



Blend (ASR=100%)

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

- Can we simply remove backdoored data by filtering out the low-loss examples at an early stage?
  - This strategy is ineffective for two reasons.
    - Reason 1
      - The training loss shown previously is the average training loss, which means some backdoor examples can still have high training loss.
      - Several powerful attacks can still succeed even with very few (50 or 100) backdoor examples.
    - Reason 2
      - If the training progresses long enough (e.g., beyond epoch 20), many clean examples will also have a low training loss, which makes the filtering significantly inaccurate.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

- Anti-Backdoor learning method
  - Decompose the entire training process into two stages
    - **Early training stage** isolates potential backdoored data.
      - Run **gradient ascent** on loss function if loss values are below a threshold.
        - Backdoor examples would escape the constraint since their loss values drop fast.
      - p percent of data with the lowest loss values will be isolated into the backdoor set
      - The rest data are put into the clean set.
      - Isolation rate (e.g., p = 1%) is assumed to be much smaller than the poisoning rate (e.g., 10%).
    - **Later training stage** unlearns identified backdoored data.
      - Run **gradient ascent** on the loss function with respect to the isolated data.
      - A model is normally trained on data in the clean set.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

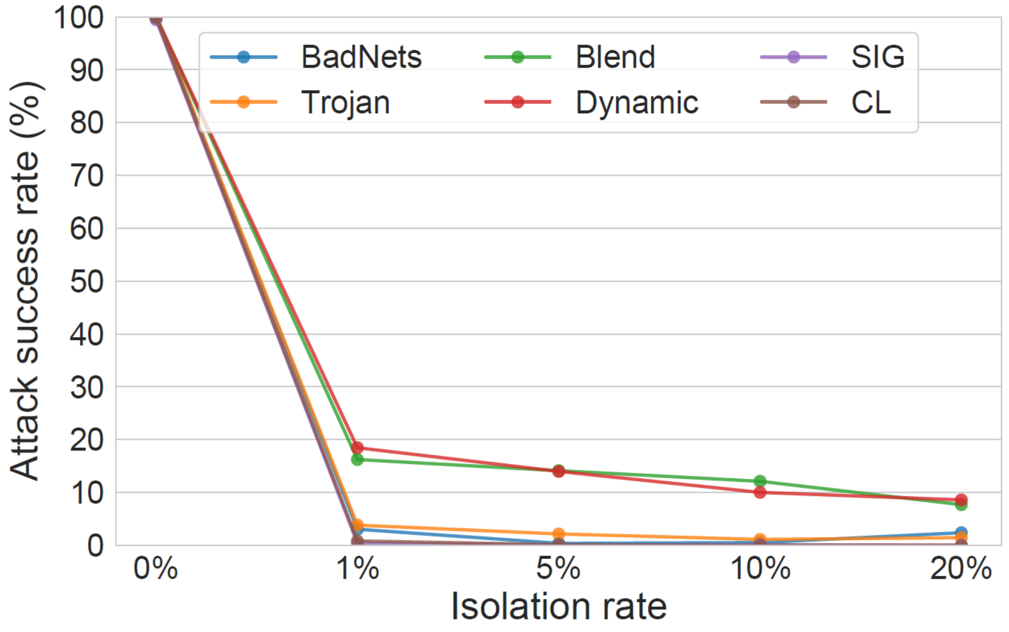
- Anti-Backdoor learning method
  - Results on CIFAR-10.
    - Compared to other defenses
      - Fine-pruning (FP), Mode Connectivity Repair (MCR) (not covered), and Neural Attention Distillation (NAD) (not covered)
    - Achieve the best accuracy on clean data and
    - Achieve the best robustness against attacks overall.
      - Less robust against Blend compared to NAD.

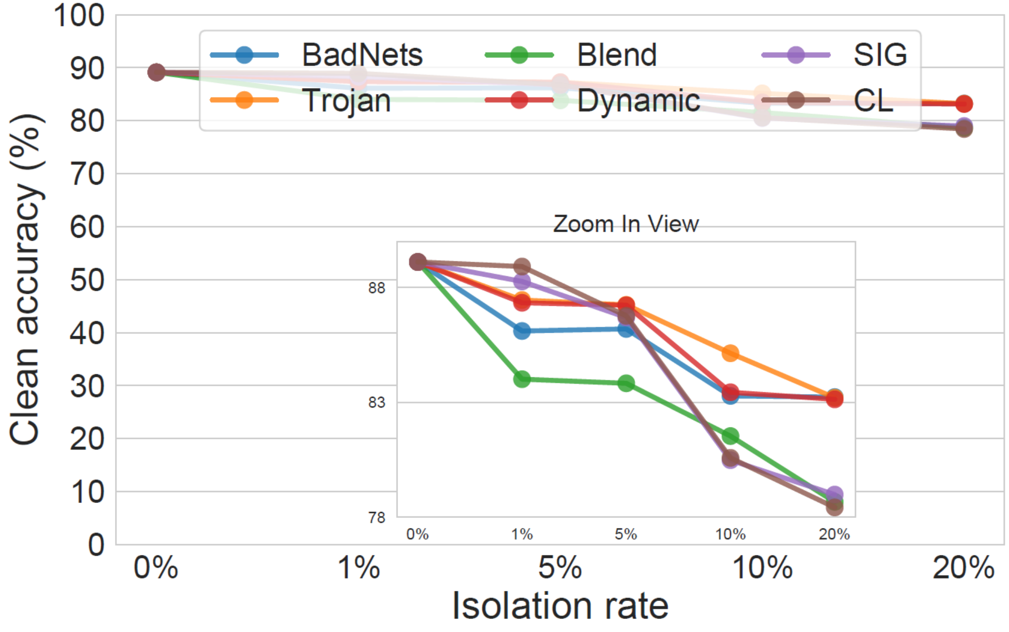
Dataset	Types	No Defense		FP		MCR		NAD		**ABL (Ours)**	
		ASR	CA	ASR	CA	ASR	CA	ASR	CA	ASR	CA
CIFAR-10	*None*	0%	89.12%	0%	85.14%	0%	87.49%	0%	88.18%	0%	**88.41%**
	BadNets	100%	85.43%	99.98%	82.14%	3.32%	78.49%	3.56%	82.18%	**3.04%**	**86.11%**
	Trojan	100%	82.14%	66.93%	80.17%	23.88%	76.47%	18.16%	80.23%	**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%**

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

- Anti-Backdoor learning method
  - Results of various rates $p \in [0.01, 0.2]$ on CIFAR-10.
    - A high isolation rate can isolate more backdoor examples for the later stage of unlearning, producing a much lower attack success rate (ASR).
    - It also puts more examples into the unlearning mode, which harms the clean accuracy.





Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Anti-Backdoor Learning

- Fixing 1% isolation rate while increasing poisoning rate (CIFAR-10).

Poisoning Rate	Defense	BadNets		Blend	
		ASR	ACC	ASR	ACC
50%	*None*	100%	75.31%	100%	69.49%
	ABL	4.98%	70.52%	27.28%	64.19%
70%	*None*	100%	74.8%	100%	67.32%
	ABL	5.02%	70.11%	62.28%	64.43%

  - With a high poisoning rate of 50%, can still reduce the ASR from 100% to 4.98% and 27.28% for BadNets and Blend.
    - Robust against BadNets even though the poison rate is 70%.
  - Potential explanation for the worse robustness against Blend.
    - Blend mixes the trigger pattern (i.e., another image) with the background of the poisoned images.
    - This makes it harder to be isolated and unlearned, since even clean data may have such patterns.

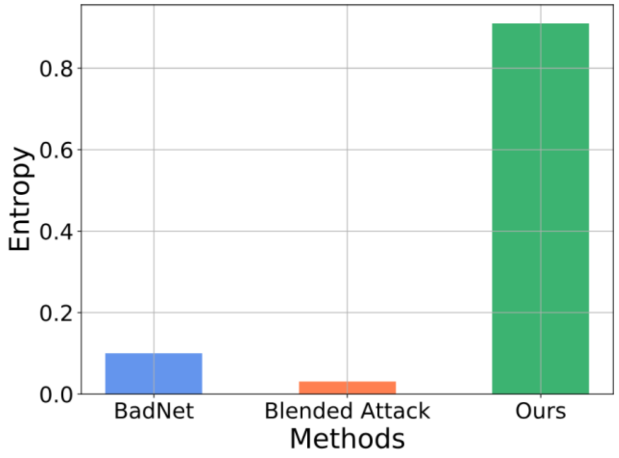
Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.

Arms race

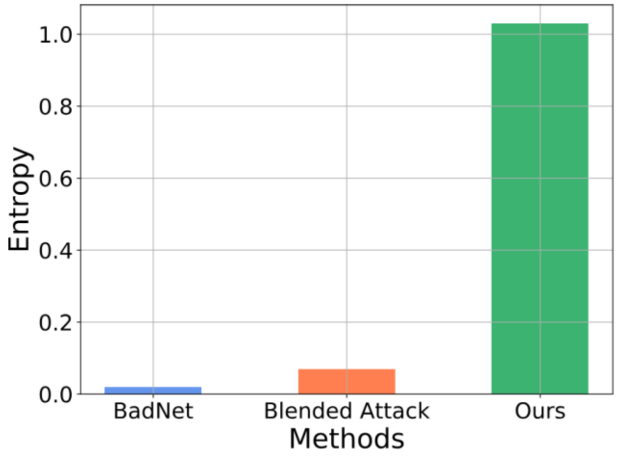
- Endless battle between backdoor attackers and defenders
    - Once a defense is proposed, there “always” will be adaptive attacks bypassing it.
        - Attacks can simply bypass defense via reasonably breaking its assumptions.
    - This is different from adversarial examples.
        - Adversarial examples are intrinsic flaws (e.g., shortcut learning) in current deep learning models.
        - Learning features that align with human perception will eventually eliminate adversarial examples.
            - It’s acceptable that humans and AI models are fooled in the same way.
                - e.g., optical illusions.

Arms race

- Breaking the assumption of defense.
  - An attack can bypass STRIP if it breaks the assumption of input-agnostic triggers.
  - Input-dependent backdoor: **Invisible Backdoor Attack with Sample-Specific Triggers (labeled as "ours").**



(a) ImageNet



(b) MS-Celeb-1M

    - The entropy generated by STRIP of different attacks.
    - The higher the entropy, the harder the attack for STRIP to defend.
  - This attack is more resistant to STRIP compared to BadNet and Blended Attack.
    - It has the potential to bypass STRIP.

Li, Y., Li, Y., Wu, B., Li, L., He, R. and Lyu, S., 2021. Invisible backdoor attack with sample-specific triggers. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 16463-16472).

References

- Liu, K., Dolan-Gavitt, B. and Garg, S., 2018, September. Fine-pruning: Defending against backdooring attacks on deep neural networks. In International symposium on research in attacks, intrusions, and defenses (pp. 273-294). Cham: Springer International Publishing.
- Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C. and Nepal, S., 2019, December. Strip: A defence against trojan attacks on deep neural networks. In Proceedings of the 35th annual computer security applications conference (pp. 113-125).
- Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B. and Ma, X., 2021. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, pp.14900-14912.
- Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H. and Zhao, B.Y., 2019, May. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In 2019 IEEE symposium on security and privacy (SP) (pp. 707-723). IEEE.
- Li, Y., Li, Y., Wu, B., Li, L., He, R. and Lyu, S., 2021. Invisible backdoor attack with sample-specific triggers. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 16463-16472).