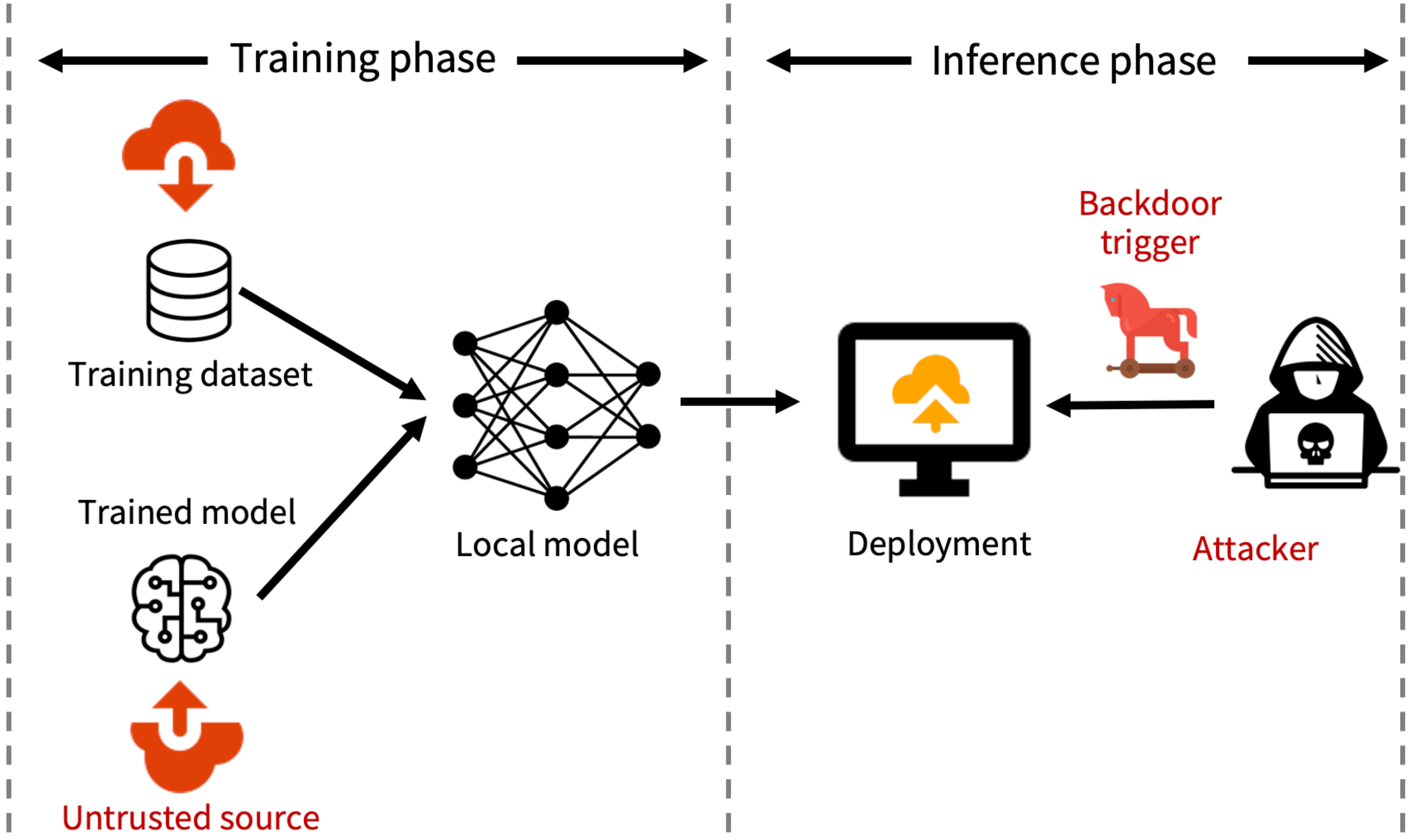
Backdoor Defenses

Zeyu Qin

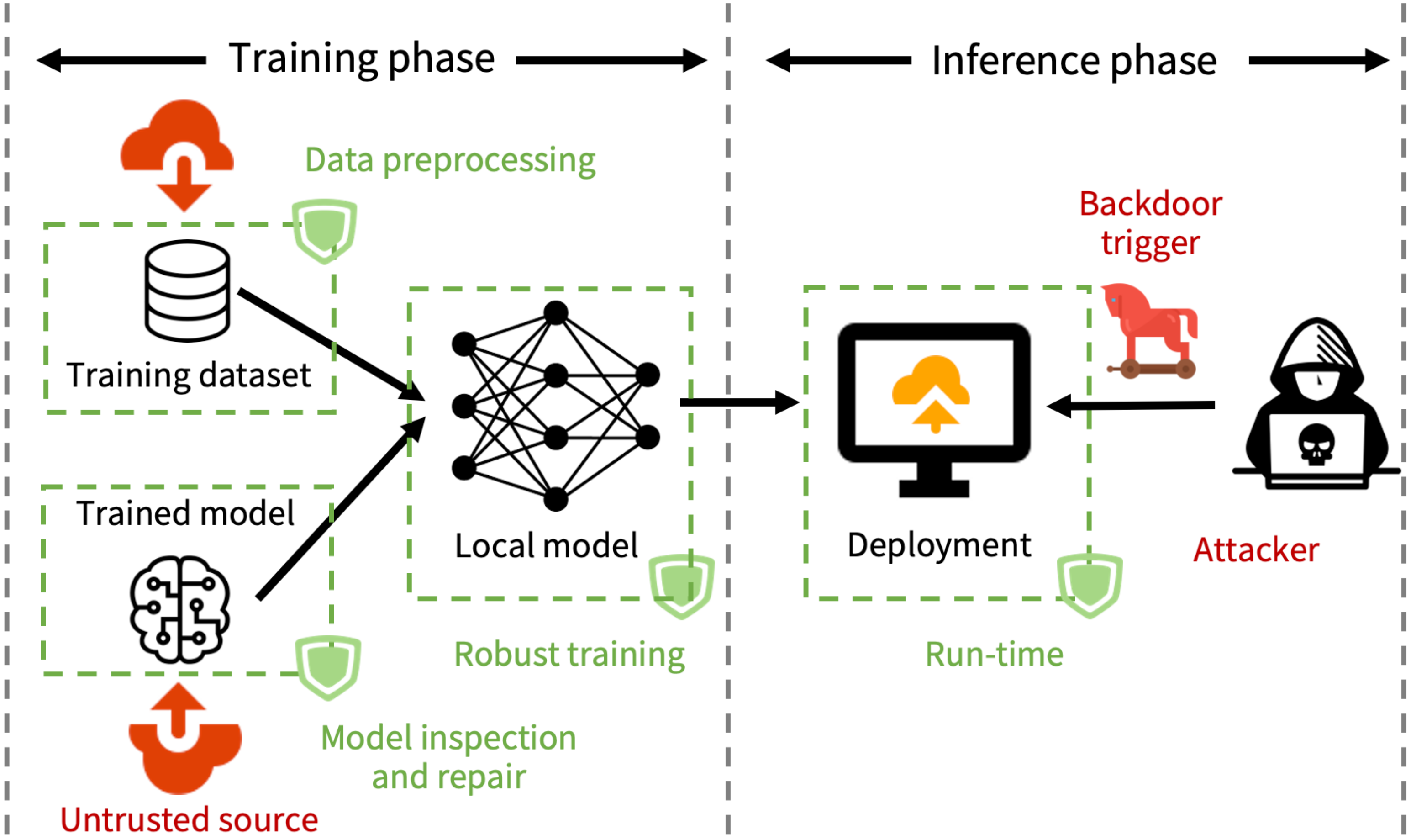
Attack Scenarios

- Adopt third-party dataset.
  - Data collection.
- Adopt third-party model.
  - Outsourcing.
  - Transfer learning.
  - Federated learning.



Defense Categories

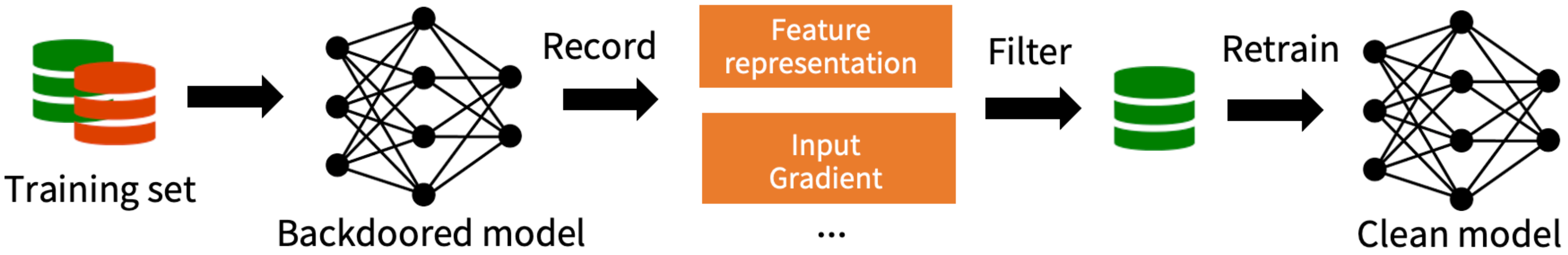
- Data preprocessing phase.
- Training phase.
- Model selection phase.
- Inference phase.



Backdoor defenses in the model life cycle

Defense Categories

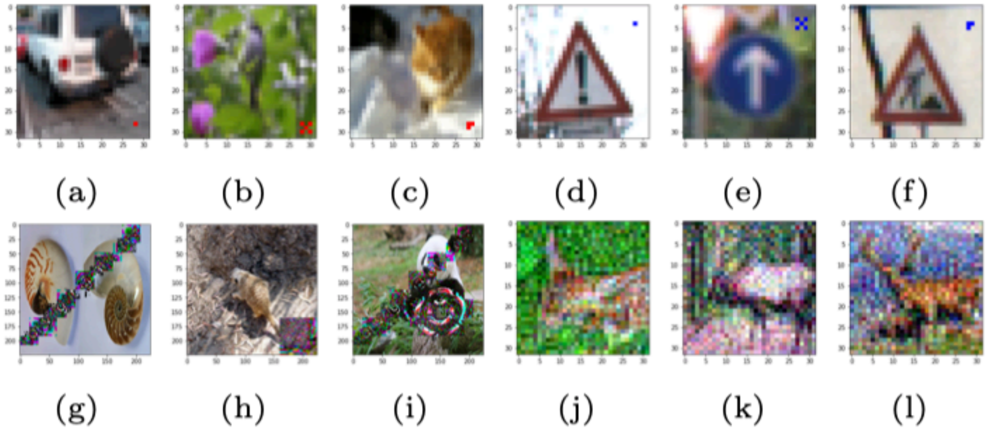
- **Data preprocessing phase defense.**



- **Training phase defense.**
  - Train a clean model under a potentially poison dataset.
  - High clean data accuracy (CDA) and low attack success rate (ASR).
- **Model selection phase defense.**
  - Given a model, identify and mitigate the backdoor.
  - Model reconstruction, trigger synthesis and model diagnosis.
- **Inference phase defense.**
  - Reject or repair the query containing the backdoor trigger.

Defense Challenges

- A weaker defender against a stronger attacker.
  - Unknown target class and poisoned samples; Limited (free) clean validation set.
- Various trigger sizes/shapes/types.
  - One pixel to blend trigger; Visible and invisible trigger.
- Multiple trigger mechanism.
  - **Input-agnostic**, class-specific and input-specific.



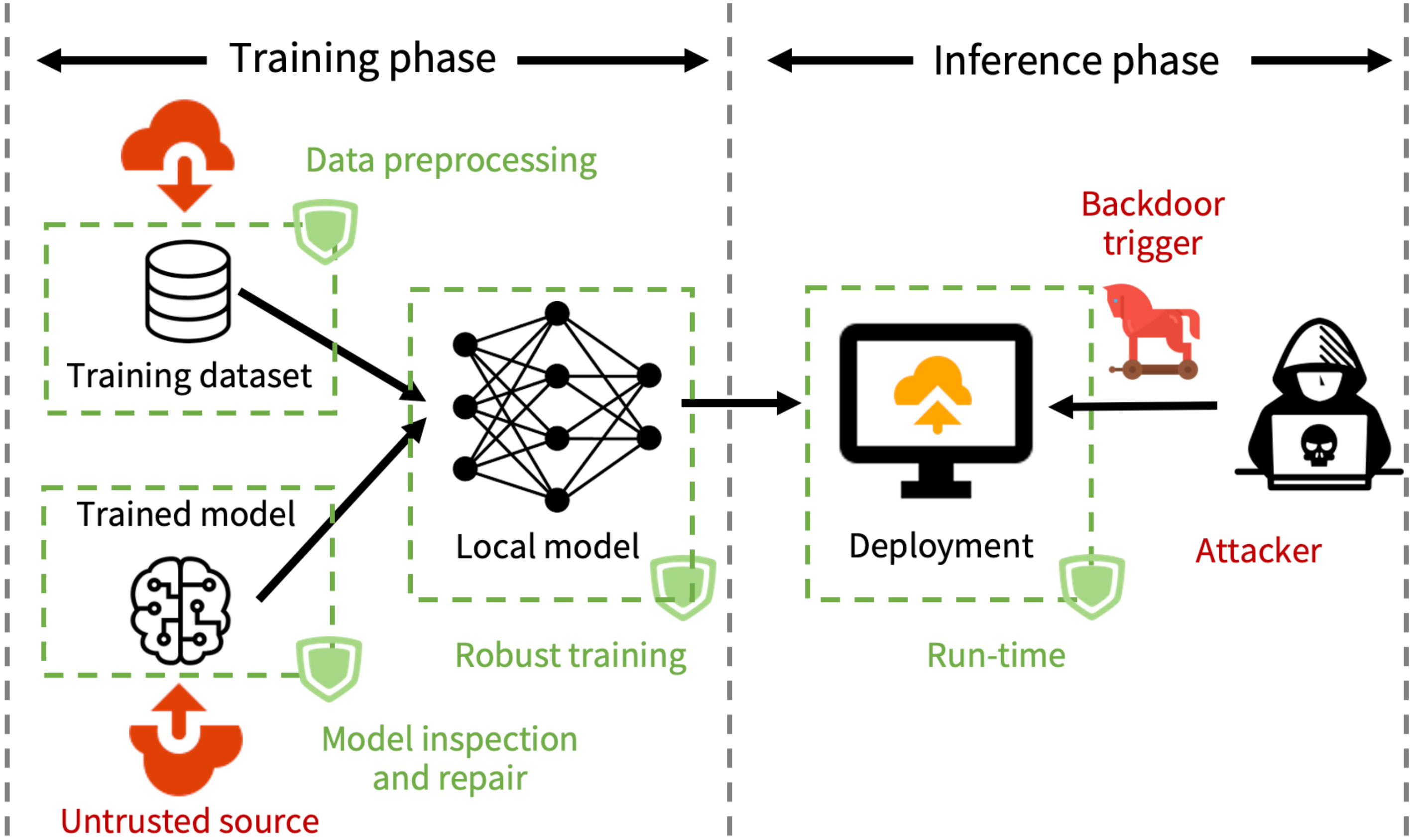
Various trigger [1]

[1] Ren Wang, et al. " Practical Detection of Trojan Neural Networks: Data-Limited and Data-Free Cases." ECCV 2020.

Model Selection Phase Defense——
Model Reconstruction

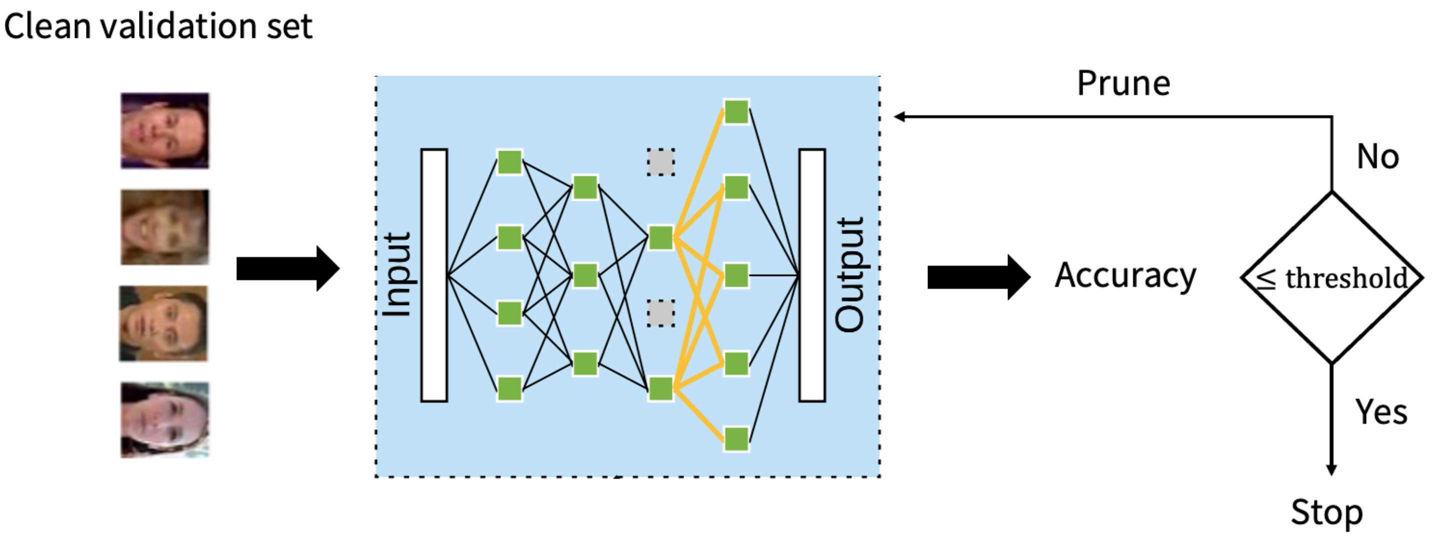
Defense Categories

- **Model selection phase defense.**
  - Given a model, identify and mitigate the backdoor.
  - **Model reconstruction**, trigger synthesis and model diagnosis.



Fine-Pruning

- **Motivation:** Backdoors exploit spare capacity in the model.
- **Assumption:** Neurons activated by clean and trigger inputs are different.
- **Method:** Pruning neurons of the model that contribute least to the main classification task.



Pruning pipeline

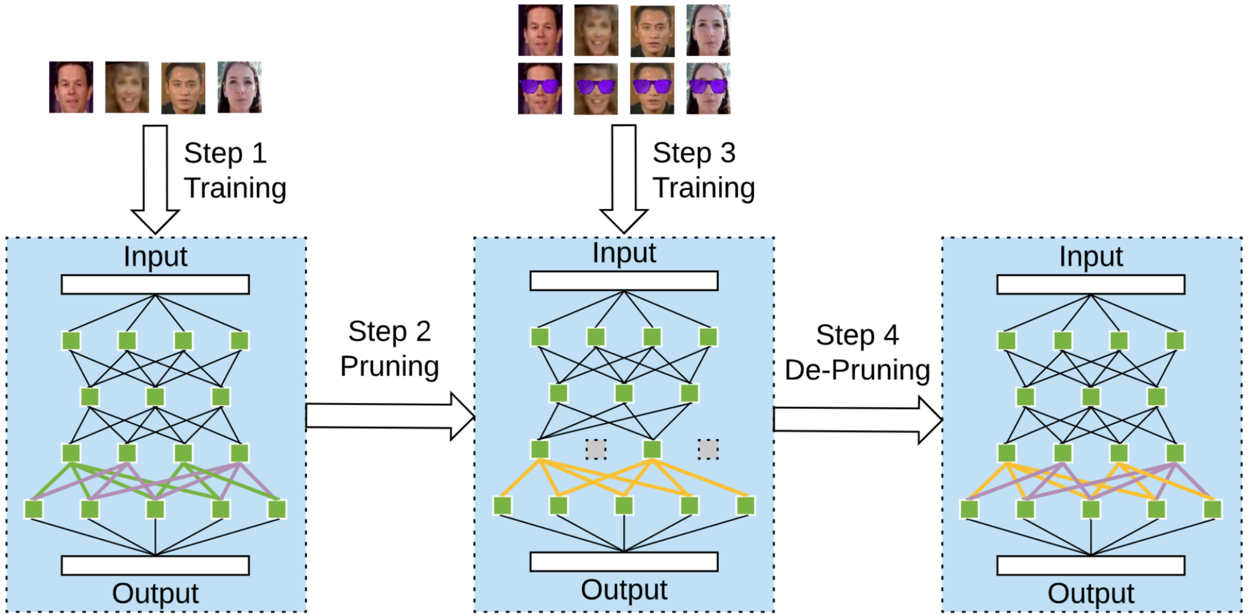
Prune neuron activated by:

1. neither clean nor backdoored inputs.
2. backdoored inputs but not clean.
3. clean inputs.

[1] Kang Liu, et al. "Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks." RAID 2018.

Fine-Pruning

- **Adaptive attack:** embed backdoor and clean feature in subset neurons.
- Use pruning+fine-tune to defense.
  - **Limitation:** requires a clean validation set.



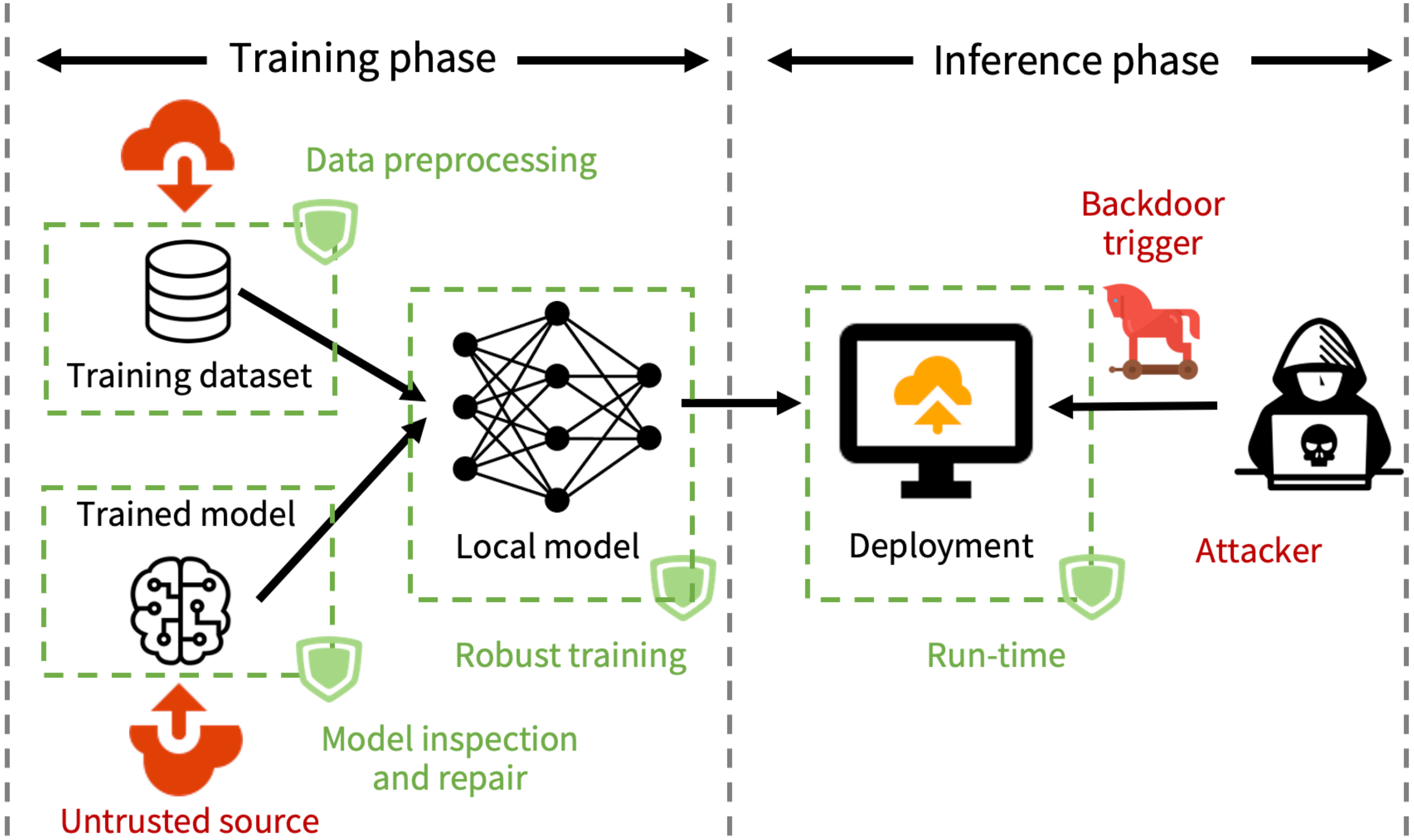
Pruning-aware attack

[1] Kang Liu, et al. "Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks." RAID 2018.

Model Selection Phase Defense——
Trigger Synthesis

Defense Categories

- Model selection phase defense.
  - Given a model, identify and mitigate the backdoor.
  - Model reconstruction, **trigger synthesis** and model diagnosis.



Neural Cleanse

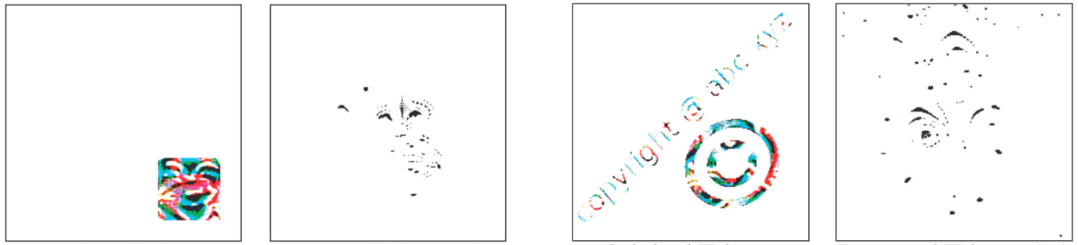
- **Motivation:** The trigger is closely related to the universal perturbation.
  - Much smaller modifications to all input samples to misclassify them into the targeted label than any other uninfected labels.
- Identify backdoor by **trigger reversing**:

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

Trigger injection

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

- Remove backdoor by retraining with the reversed trigger.



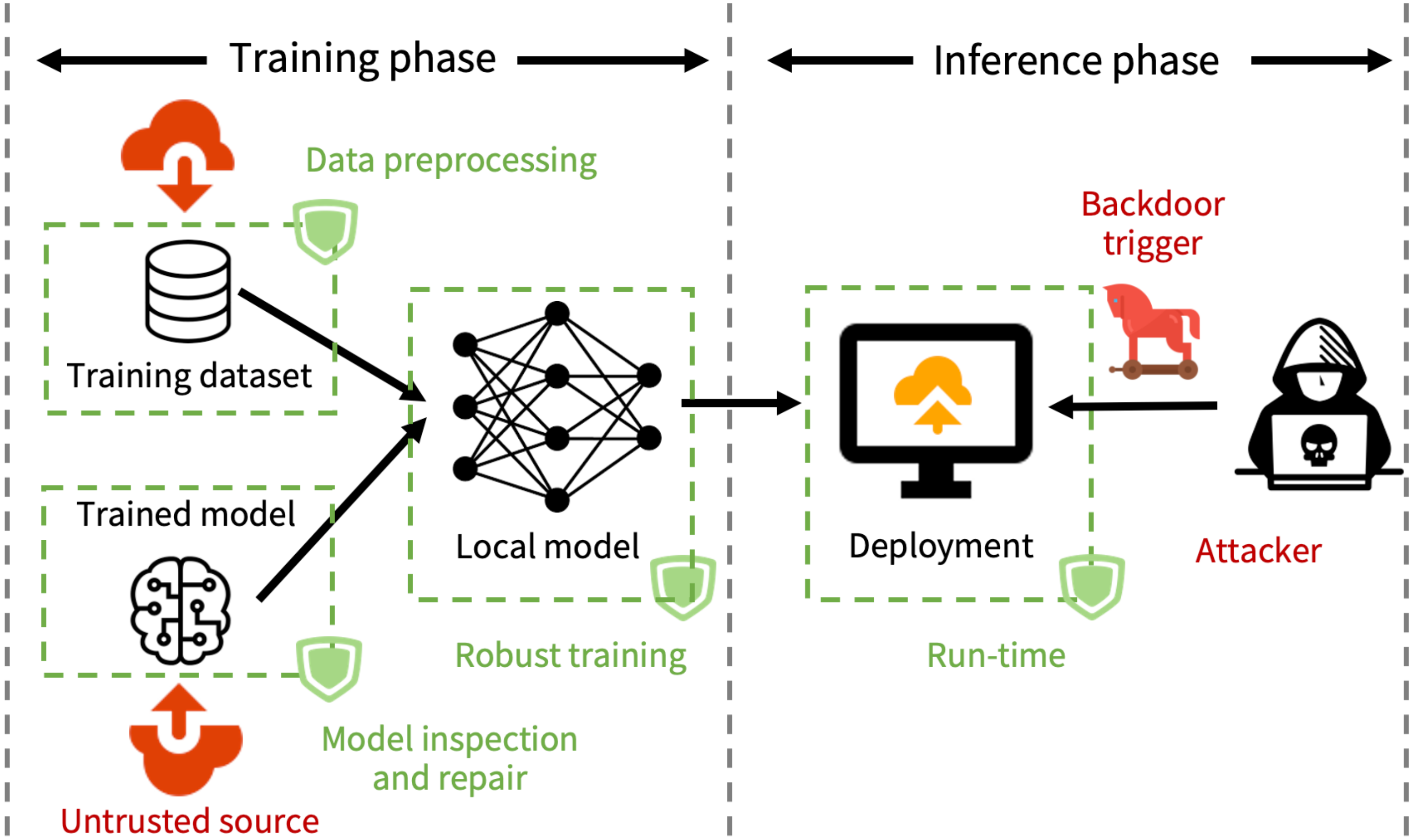
Original Trigger (L1 norm = 3,481) | Reversed Trigger ($\boldsymbol{m}$) (L1 norm = 311.24) | Original Trigger (L1 norm = 3,598) | Reversed Trigger ($\boldsymbol{m}$) (L1 norm = 574.24)

[1] Bolun Wang, et al. "Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks." IEEE S&P 2019.

Model Selection Phase Defense——
Model Diagnosis

Defense Categories

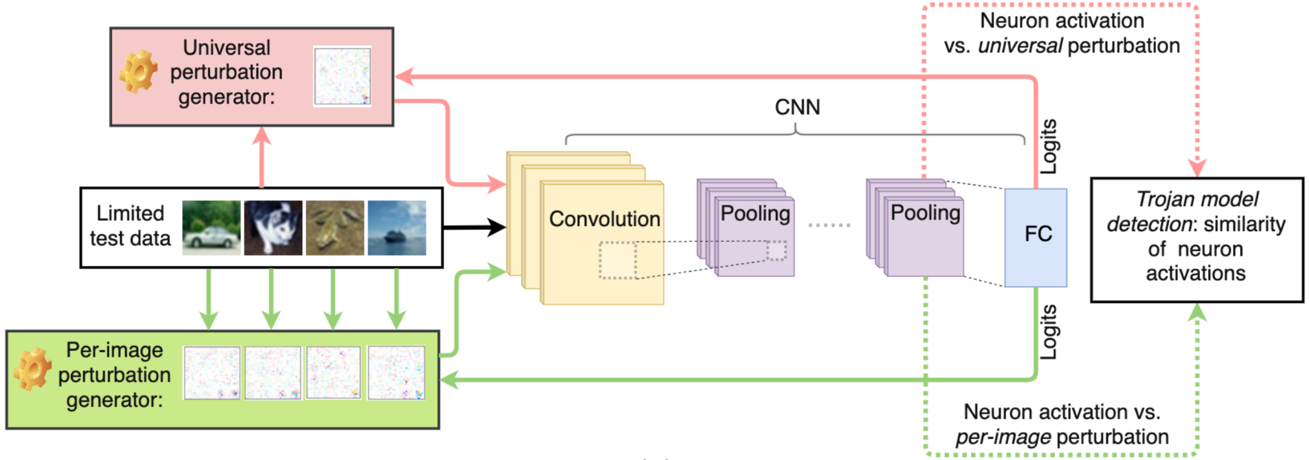
- Model selection phase defense.
  - Given a model, identify and mitigate the backdoor.
  - Model reconstruction, trigger synthesis and **model diagnosis**.



DL-TND

- A **D**ata-**L**imited (one sample per class) **T**rojan**N**et **D**etector.
- **Motivation: input-agnostic** misclassification (shortcut) of TrojanNet.
- **Method:** per-image and universal perturbations would maintain a strong similarity while perturbing images towards the Trojan target class.

$$\hat{\mathbf{x}}(\mathbf{m}, \boldsymbol{\delta}) = (1 - \mathbf{m}) \cdot \mathbf{x} + \mathbf{m} \cdot \boldsymbol{\delta}$$



(a)

universal perturbation

$$\begin{aligned} &\underset{\mathbf{m}, \boldsymbol{\delta}}{\text{minimize}} \quad \ell_{\text{atk}}(\hat{\mathbf{x}}(\mathbf{m}, \boldsymbol{\delta}); \mathcal{D}_{k-}) + \bar{\ell}_{\text{atk}}(\hat{\mathbf{x}}(\mathbf{m}, \boldsymbol{\delta}); \mathcal{D}_k) + \lambda \|\mathbf{m}\|_1 \\ &\text{subject to } \{\boldsymbol{\delta}, \mathbf{m}\} \in \mathcal{C}, \end{aligned}$$

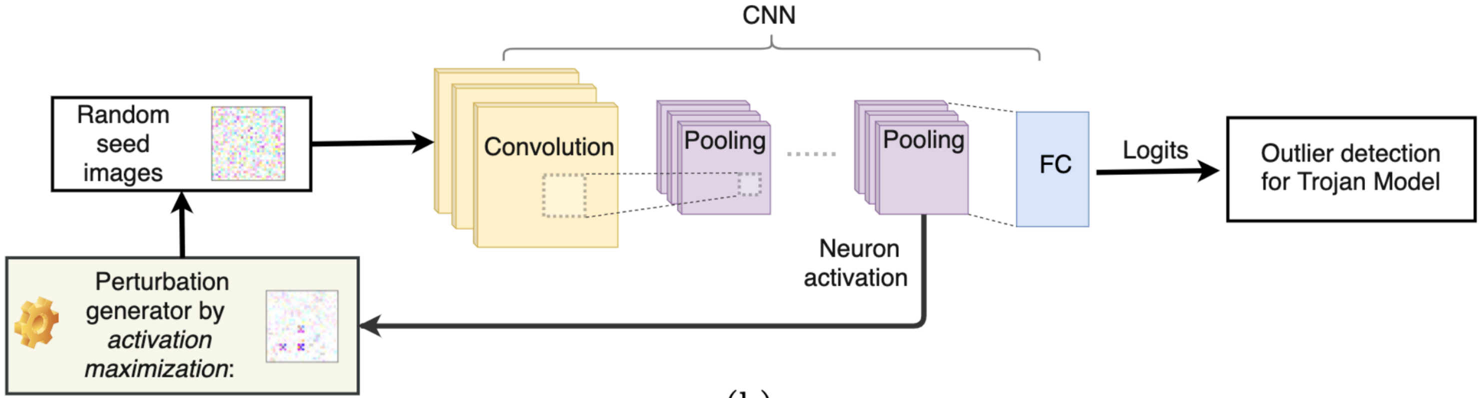
per-sample perturbation

$$\begin{aligned} &\underset{\mathbf{m}, \boldsymbol{\delta}}{\text{minimize}} \quad \ell'_{\text{atk}}(\hat{\mathbf{x}}(\mathbf{m}, \boldsymbol{\delta}); \mathbf{x}_i) + \lambda \|\mathbf{m}\|_1 \\ &\text{subject to } \{\boldsymbol{\delta}, \mathbf{m}\} \in \mathcal{C}, \ \text{for } \mathbf{x}_i \in \mathcal{D}_k \end{aligned}$$

[1] Ren Wang, et al. " Practical Detection of Trojan Neural Networks: Data-Limited and Data-Free Cases." ECCV 2020.

DF-TND

- A **D**ata-**F**ree **T**rojan**N**et **D**etector with access to the model weight.
- **Motivation:** a TrojanNet exhibits an unexpectedly high neuron activation at certain coordinates.



(b)

activation maximization

$$\underset{\mathbf{m},\boldsymbol{\delta},\mathbf{w}}{\text{maximize}} \ \sum_{i=1}^{d} [w_i r_i(\hat{\mathbf{x}}(\mathbf{m},\boldsymbol{\delta}))] - \lambda \|\mathbf{m}\|_1$$
$$\text{subject to } \{\boldsymbol{\delta}, \mathbf{m}\} \in \mathcal{C}, \mathbf{0} \le \mathbf{w} \le \mathbf{1}, \mathbf{1}^T \mathbf{w} = 1,$$

detection rule

$$L_k = \frac{1}{N} \sum_{i}^{N} [f_k(\hat{\mathbf{x}}_i(\mathbf{p}^{(i)})) - f_k(\mathbf{x}_i)]$$
For each label $k \in [K]$

[1] Ren Wang, et al. " Practical Detection of Trojan Neural Networks: Data-Limited and Data-Free Cases." ECCV 2020.