Experimental Validation

Description of Datasets and Tasks

Datasets		**Tasks**
MNIST	→	Hand-written digit recognition

Description of Datasets and Tasks

Datasets		**Tasks**
MNIST	→	Hand-written digit recognition
GTSRB	→	Traffic sign recognition

Description of Datasets and Tasks

Datasets		Tasks
MNIST	→	Hand-written digit recognition
GTSRB	→	Traffic sign recognition
YouTube Face	→	Face recognition

Description of Datasets and Tasks

Datasets	Tasks
MNIST	Hand-written digit recognition
GTSRB	Traffic sign recognition
YouTube Face	Face recognition
PubFig	Face recognition with transfer learning

Description of Datasets and Tasks

Datasets	Tasks
MNIST	Hand-written digit recognition
GTSRB	Traffic sign recognition
YouTube Face	Face recognition
PubFig	Face recognition with transfer learning
VGG Face	Face recognition (Trojan attack)

Dataset and Model Details

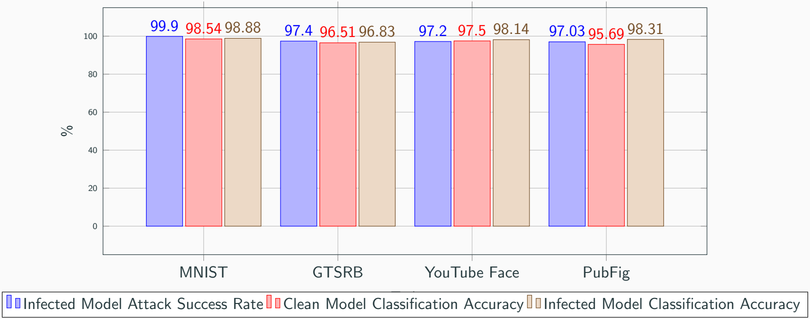
Task	Dataset	# of Labels	Input Size	Model Architecture
Hand-written Digit Recognition	MNIST	10	$28 \times 28 \times 1$	2 Conv + 2 Dense
Traffic Sign Recognition	GTSRB	43	$32 \times 32 \times 3$	6 Conv + 2 Dense
Face Recognition	YouTube Face	1,283	$55 \times 47 \times 3$	4 Conv + 1 Merge + 1 Dense
Face Recognition (w/ Transfer Learning)	PubFig	65	$224 \times 224 \times 3$	13 Conv + 3 Dense
Face Recognition (Trojan Attack)	VGG Face	2,622	$224 \times 224 \times 3$	13 Conv + 3 Dense

Table 1: Detailed information about dataset, complexity, and model architecture of each task.

Performance of Backdoor Injection Attacks

Attack success rate and classification accuracy of backdoor injection attack on four classification tasks.

Attack success rate and classification accuracy of backdoor injection attack on four classification tasks.



Backdoor Detection Performance

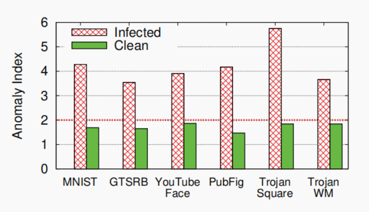
- **Detection success rate**: High anomaly index observed for **infected models**.

Backdoor Detection Performance

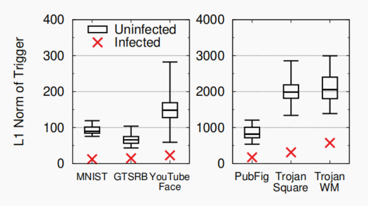
- **Detection success rate**: High anomaly index observed for **infected models**.
- **L1 norm of the trigger**: Optimized triggers exhibit low L1 norm, highlighting sparsity in their patterns.

Backdoor Detection Performance

- **Detection success rate**: High anomaly index observed for **infected models**.
- **L1 norm of the trigger**: Optimized triggers exhibit low L1 norm, highlighting sparsity in their patterns.



(a) Anomaly measurement of infected and clean model

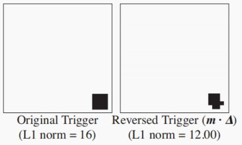


(b) L1 norm of triggers for infected and uninfected labels

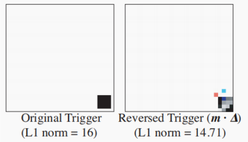
Figure 1: Comparison of trigger visualizations.

Identification of Original Triggers

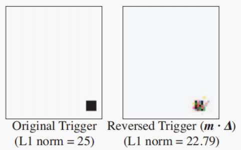
- **End-to-End Effectiveness**



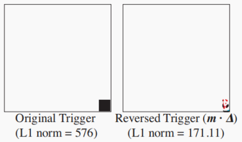
(a) MNIST



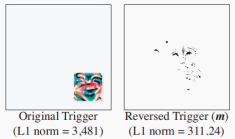
(b) GTSRB



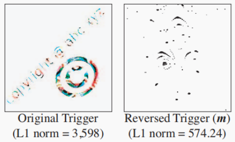
(c) YouTube Face



(d) PubFig



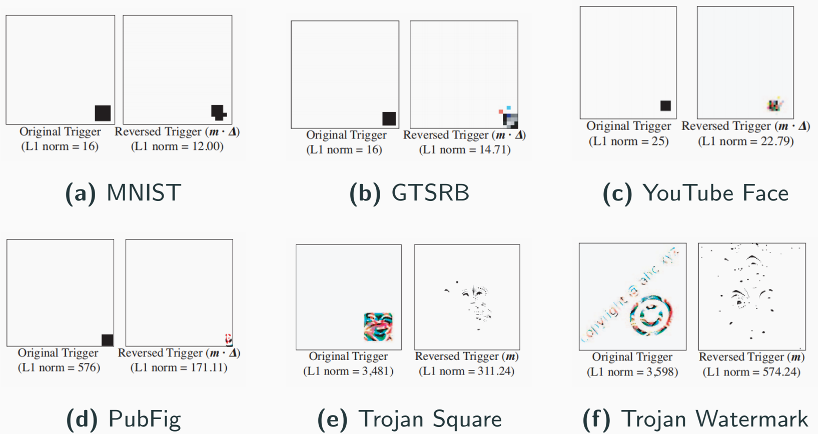
(e) Trojan Square



(f) Trojan Watermark

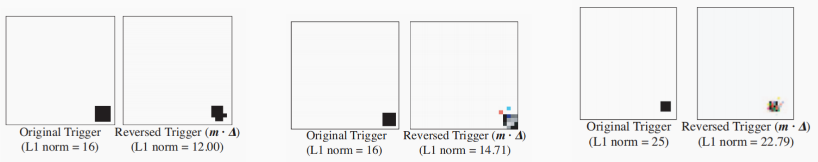
Identification of Original Triggers

- **End-to-End Effectiveness**
- **Visual Similarity**

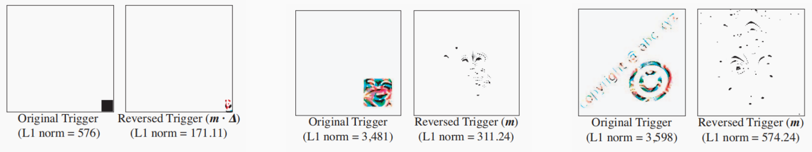


Identification of Original Triggers

- **End-to-End Effectiveness**
- **Visual Similarity**
- **Compactness of the Trigger**



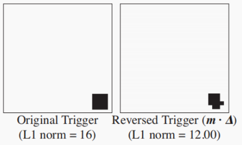
(a) MNIST **(b)** GTSRB **(c)** YouTube Face



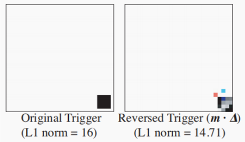
(d) PubFig **(e)** Trojan Square **(f)** Trojan Watermark

Identification of Original Triggers

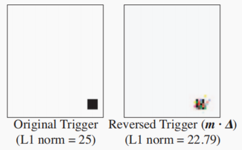
- **End-to-End Effectiveness**
- **Visual Similarity**
- **Compactness of the Trigger**
- **Model Behavior**



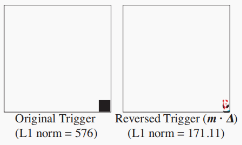
(a) MNIST



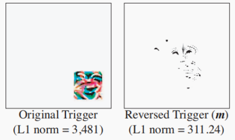
(b) GTSRB



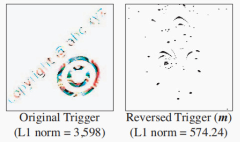
(c) YouTube Face



(d) PubFig



(e) Trojan Square

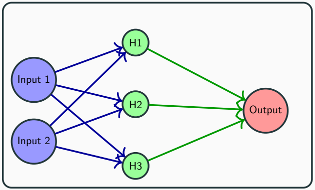


(f) Trojan Watermark

Mitigation Techniques

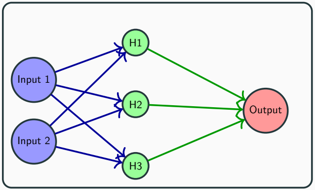
Mitigation Procedure

Backdoor Model (impurity)



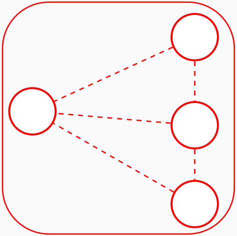
Mitigation Procedure

Backdoor Model (impurity)

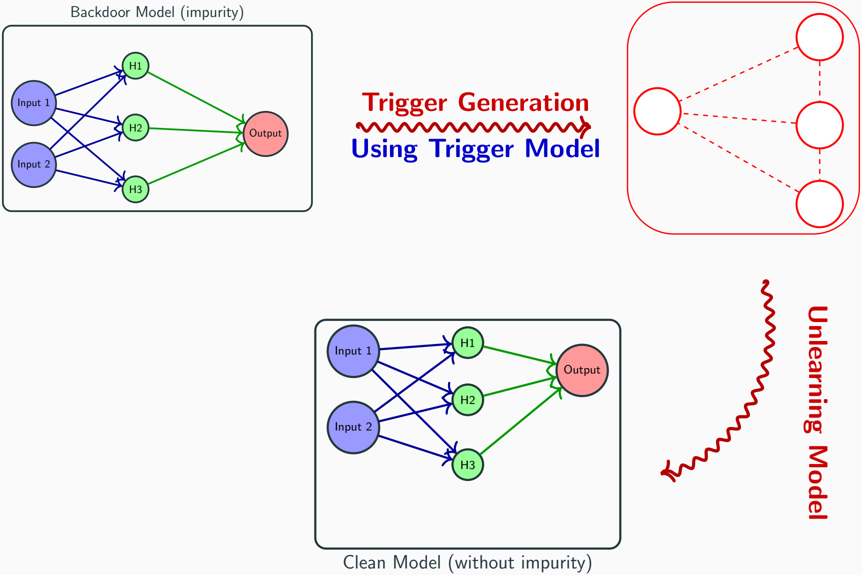


Trigger Generation

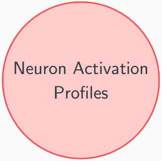
Using Trigger Model



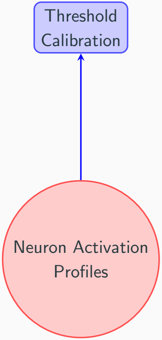
Mitigation Procedure



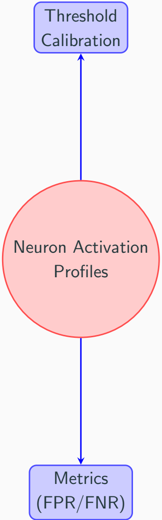
Filtering Adversarial Inputs Based on Neuron Activation Profiles



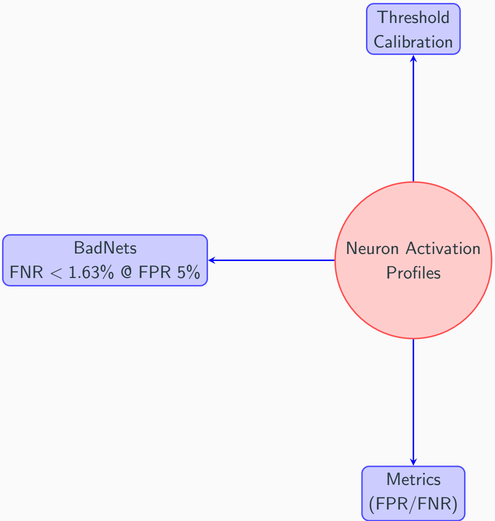
Filtering Adversarial Inputs Based on Neuron Activation Profiles



Filtering Adversarial Inputs Based on Neuron Activation Profiles

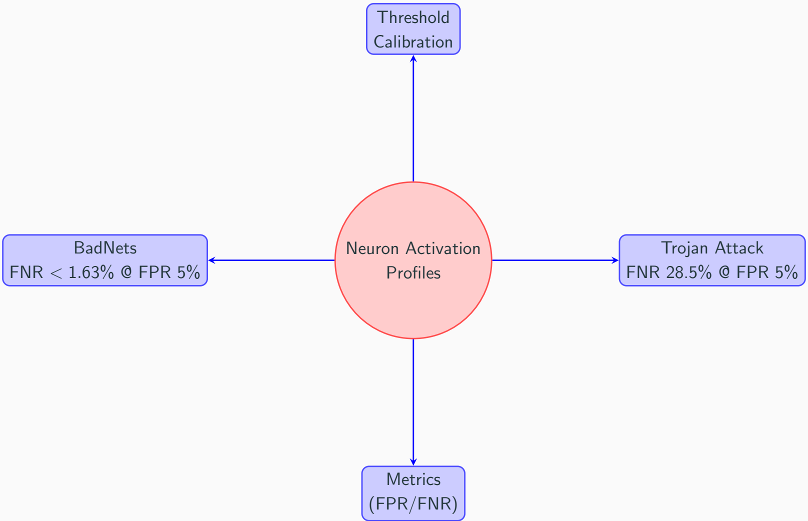


Filtering Adversarial Inputs Based on Neuron Activation Profiles

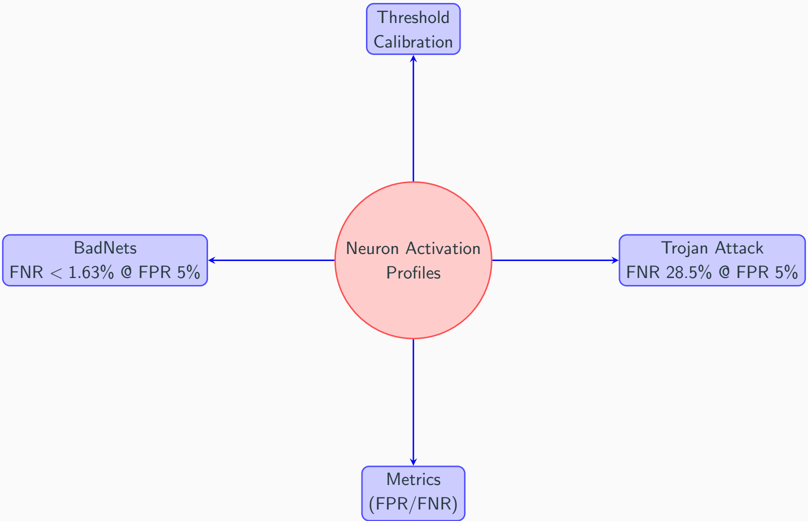


Filtering Adversarial Inputs

Filtering Adversarial Inputs Based on Neuron Activation Profiles



Filtering Adversarial Inputs Based on Neuron Activation Profiles



ROC Curve: Final Visualization

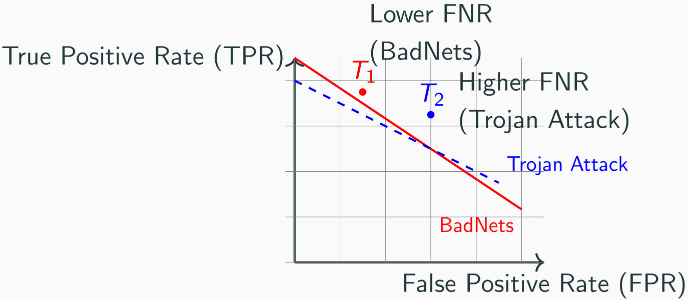
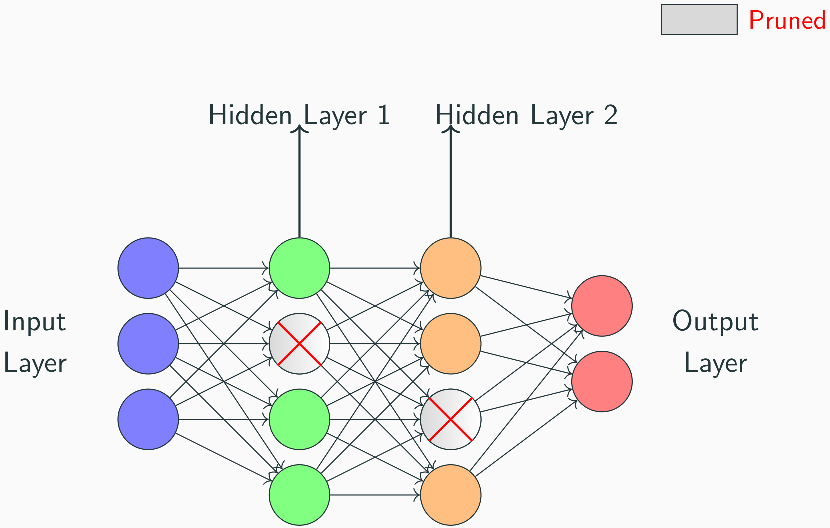
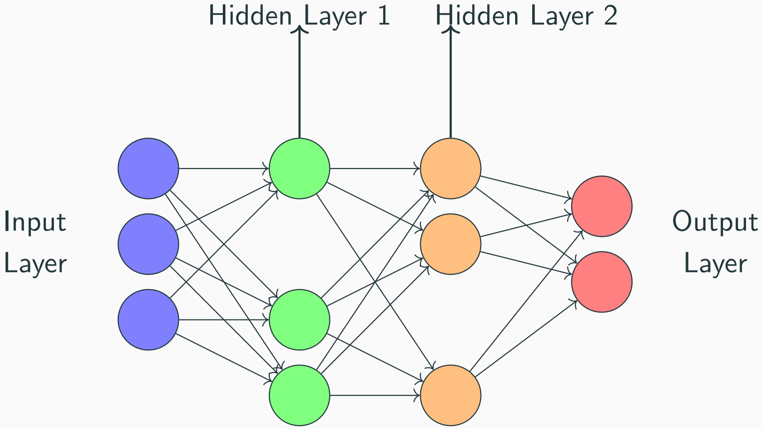


Figure 3: Final: ROC Curve Comparison with Thresholds.

Patching DNNs via Neuron Pruning (Graphical View)



Patching DNNs via Neuron Pruning (After Removal)



Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

↓

Step 2: Prioritize neurons with largest activation gaps

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

↓

Step 2: Prioritize neurons with largest activation gaps

↓

Step 3: Minimize impact on classification accuracy

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

↓

Step 2: Prioritize neurons with largest activation gaps

↓

Step 3: Minimize impact on classification accuracy

↓

Step 4: Attack success rate drops to nearly 0% with 30% pruning

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

Step 2: Prioritize neurons with largest activation gaps

Step 3: Minimize impact on classification accuracy

Step 4: Attack success rate drops to nearly 0% with 30% pruning

Step 5: Redundancy in DNNs requires pruning ¿1% of neurons

Patching DNNs via Neuron Pruning

1. Step 1: Prune backdoor-related neurons using reversed trigger
2. Step 2: Prioritize neurons with largest activation gaps
3. Step 3: Minimize impact on classification accuracy
4. Step 4: Attack success rate drops to nearly 0% with 30% pruning
5. Step 5: Redundancy in DNNs requires pruning ¿1% of neurons
6. Step 6: YouTube Face shows higher classification accuracy drop

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

Step 2: Prioritize neurons with largest activation gaps

Step 3: Minimize impact on classification accuracy

Step 4: Attack success rate drops to nearly 0% with 30% pruning

Step 5: Redundancy in DNNs requires pruning ¿1% of neurons

Step 6: YouTube Face shows higher classification accuracy drop

Step 7: Pruning at the last convolution layer yields best results

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

Step 2: Prioritize neurons with largest activation gaps

Step 3: Minimize impact on classification accuracy

Step 4: Attack success rate drops to nearly 0% with 30% pruning

Step 5: Redundancy in DNNs requires pruning ¿1% of neurons

Step 6: YouTube Face shows higher classification accuracy drop

Step 7: Pruning at the last convolution layer yields best results

Step 8: Trojan models less affected due to dissimilarity

Patching DNNs via Neuron Pruning

Step 1: Prune backdoor-related neurons using reversed trigger

↓

Step 2: Prioritize neurons with largest activation gaps

↓

Step 3: Minimize impact on classification accuracy

↓

Step 4: Attack success rate drops to nearly 0% with 30% pruning

↓

Step 5: Redundancy in DNNs requires pruning ¿1% of neurons

↓

Step 6: YouTube Face shows higher classification accuracy drop

↓

Step 7: Pruning at the last convolution layer yields best results

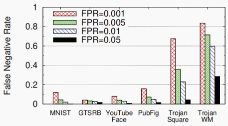
↓

Step 8: Trojan models less affected due to dissimilarity

↓

Step 9: Neuron pruning is computationally efficient but needs tuning

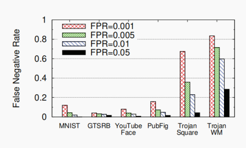
Neuron Pruning for Deep Neural Network (DNN) Patching.



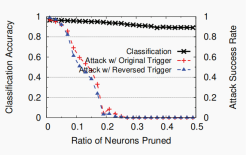
(a) False negative rate of proactive adversarial image detection when achieving different false positive rates.

Patching DNNs via Neuron Pruning

Neuron Pruning for **Deep Neural Network (DNN) Patching**.



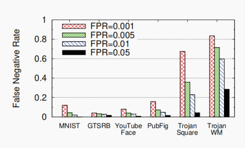
(a) False negative rate of proactive adversarial image detection when achieving different false positive rates.



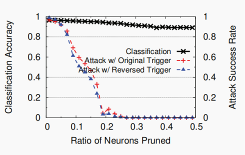
(b) Classification accuracy and attack success rate when pruning trigger-related neurons in GTSRB (traffic sign recognition w/ 43 labels).

Patching DNNs via Neuron Pruning

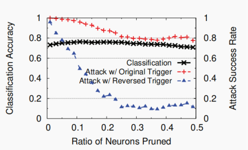
Neuron Pruning for Deep Neural Network (DNN) Patching.



(a) False negative rate of proactive adversarial image detection when achieving different false positive rates.



(b) Classification accuracy and attack success rate when pruning trigger-related neurons in GTSRB (traffic sign recognition w/ 43 labels).



(c) Classification accuracy and attack success rate when pruning trigger-related neurons in Trojan Square (face recognition w/ 2,622 labels).

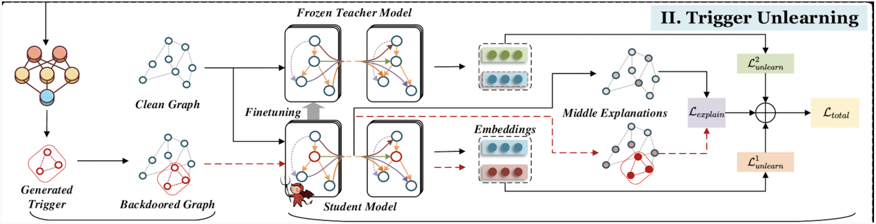
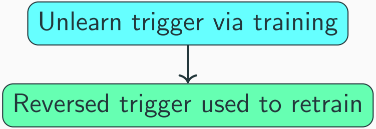


Figure 5: Trigger Unlearning graphical visualization

Patching DNNs via Unlearning

Unlearn trigger via training

Patching DNNs via Unlearning



Patching DNNs via Unlearning

Unlearn trigger via training

↓

Reversed trigger used to retrain

↓

Fine-tune model for 1 epoch

Patching DNNs via Unlearning

Unlearn trigger via training

↓

Reversed trigger used to retrain

↓

Fine-tune model for 1 epoch

↓

Reduces attack success rate to ¡6.7%

Patching DNNs via Unlearning

Unlearn trigger via training

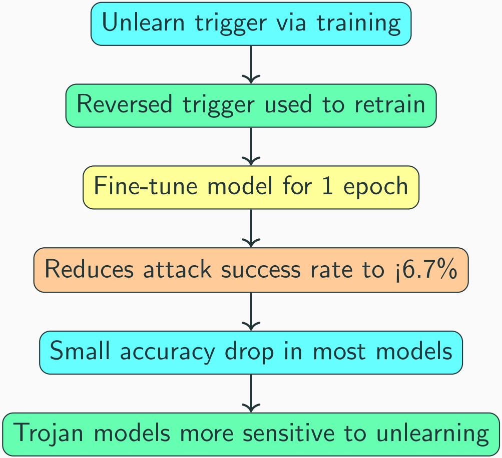
Reversed trigger used to retrain

Fine-tune model for 1 epoch

Reduces attack success rate to ¡6.7%

Small accuracy drop in most models

Patching DNNs via Unlearning



Patching DNNs via Unlearning

Unlearn trigger via training

↓

Reversed trigger used to retrain

↓

Fine-tune model for 1 epoch

↓

Reduces attack success rate to ¡6.7%

↓

Small accuracy drop in most models

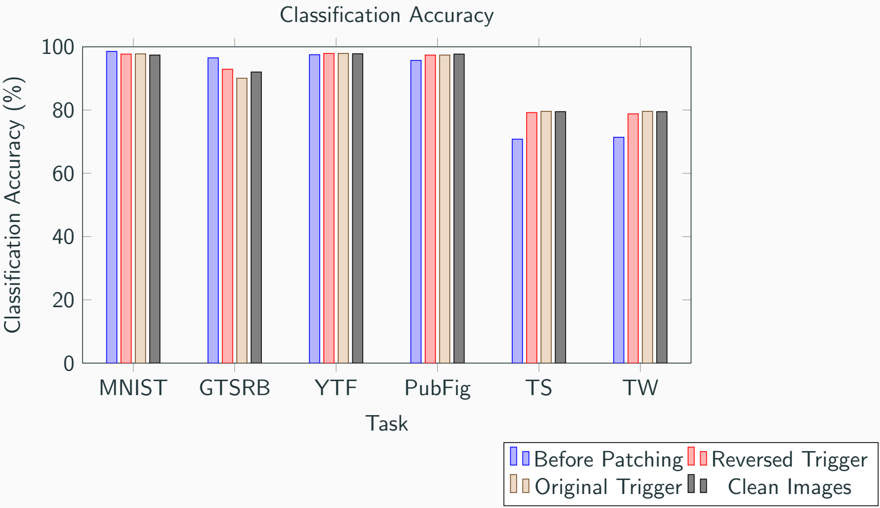
↓

Trojan models more sensitive to unlearning

↓

Unlearning less effective for BadNets

Classification Accuracy After Patching



Attack Success Rate After Patching