# TEN-GUARD: TENSOR DECOMPOSITION FOR BACKDOOR ATTACK DETECTION IN DEEP NEURAL NETWORKS

Khondoker Murad Hossain [1] Tim Oates [1]

[1]University of Maryland Baltimore County, Baltimore, US

## Motivation

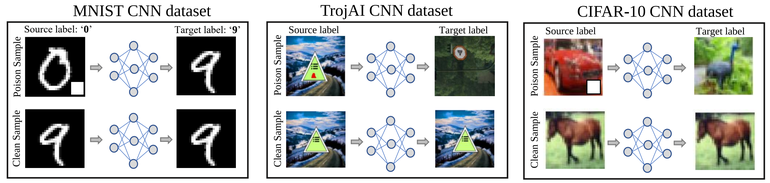
- The widespread use of pre-trained neural network models, due to the prohibitive costs and resources required to train large models from scratch, introduces risks of embedded malicious behaviors (e.g., backdoors or trojans) that can manipulate the model's output in subtle but dangerous ways.
- Existing methods for detecting such malicious alterations in models often assume knowledge about the nature of the triggers or are limited to specific network architectures and do not scale well across different models.
- Our approach utilizes tensor decomposition techniques—specifically Independent Vector Analysis (IVA) and Parallel Factor Analysis (PARAFAC2)—to analyze network activations, providing a scalable and architecture-independent method for detecting trojaned models without assumptions about trigger types, thereby enhancing the security and trustworthiness of deployed neural network models.

## Problem Statement

- A backdoored model $F(\cdot)$ arises when the training dataset $\mathcal{D}$ is compromised by introducing a subset $\mathcal{P} \subset \mathcal{D}$, where the images are altered with specific triggers and their class labels are maliciously changed to a target class $t$. This manipulation causes $F(\cdot)$ to classify these poisoned inputs $x \in \mathcal{P}$ to the target class $t$, even though $t$ is incorrect.
- During inference, the backdoored model $F(\cdot)$ functions normally for clean inputs, providing correct class outputs. However, when a triggered sample $x \in \mathcal{P}$ is presented, the model erroneously outputs the class $t$, demonstrating the hidden malicious behavior embedded during the training phase.
- The goal of this method is to detect the trojan models before they are deployed in real world applications.

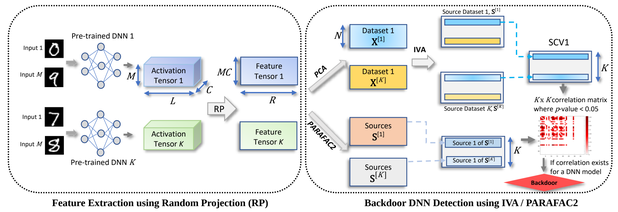
## Datasets

- **MNIST CNN Dataset:** Used 450 CNN models on MNIST, half backdoored with a 4x4 pixel white patch on '0's, achieving a 99.92% attack success rate.
- **CIFAR-10 CNN Dataset:** Utilized 550 ResNet-18 models on CIFAR-10, half with a white patch backdoor, showing a 98.89% attack success rate.
- **TrojAI Dataset:** Used TrojAI Image Classification dataset across three architectures for synthetic traffic data classification, including separate 'Test' and 'Holdout' sets to evaluate detection methods.




## Backdoor DNN Detection Pipeline

- **Feature Extraction**: Extract final layer activations for $K$ DNNs and then apply Random Projection for feature extraction.
- **Backdoor Detection**: Apply IVA and PARAFAC2 to decompose the feature tensors, identifying correlated source component vectors across $K$ DNNs. Models are classified as backdoored or clean based on the cross-correlation matrix and $p$-values.




## Results: Correlation Analysis

- Post IVA and PARAFAC2, we compute $K \times K$ Pearson correlation matrices, then validate these using $t$-tests to filter for significant correlations ($p < 0.05$).
- Models are classified as backdoored based on significant correlations detected in the matrices, effectively identifying most backdoored models with minimal false negatives.

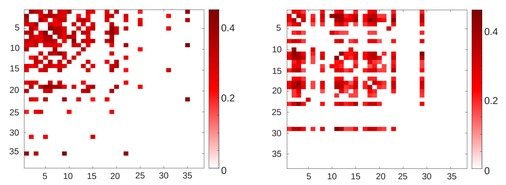


Figure 1. Correlation matrices from IVA and PARAFAC2 for TrojAI I - R50 models. The first 22 are backdoored and the last 16 are clean. Red boxes signal significant correlations; a model with any red box is flagged as backdoored.

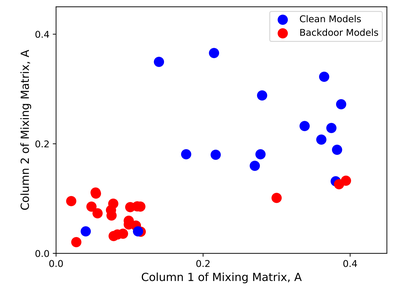
## Results: Performance Metrics

- PARAFAC2 outperforms IVA across all datasets and model architectures (including MNIST, CIFAR-10, and TrojAI I  II), demonstrating superior precision, recall, and accuracy.

| | IVA | | | PARAFAC2 | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | Accuracy | Precision | Recall | Accuracy |
| **MNIST** | 0.91 | 0.89 | 0.91 | 0.93 | 0.91 | **0.92** |
| **CIFAR-10** | 0.86 | 0.84 | 0.85 | 0.89 | 0.87 | **0.88** |
| **TrojAI I-R50** | 0.86 | 0.86 | 0.84 | 0.90 | 0.86 | **0.87** |
| **TrojAI II-R50** | 0.91 | 0.78 | 0.82 | 1.00 | 0.78 | **0.87** |
| **TrojAI I-D121** | 0.76 | 0.81 | 0.79 | 0.82 | 0.88 | **0.85** |
| **TrojAI II-D121** | 0.73 | 0.80 | 0.79 | 0.82 | 0.90 | **0.83** |
| **TrojAI I-Iv3** | 0.73 | 0.73 | 0.78 | 0.75 | 0.82 | **0.81** |
| **TrojAI II-Iv3** | 0.77 | 0.77 | 0.80 | 0.79 | 0.85 | **0.83** |

## Results: Trojan and Clean DNN Clustering

- Cluster DNN models into two distinct groups by analyzing the first two columns of the mixing matrix $\mathbf{A}$ from IVA and PARAFAC2; trojan models form a dense cluster separate from clean models, demonstrating spatial separation in feature space.
- PARAFAC2 provides reliable clustering without the need for PCA, as shown by high silhouette scores in the table, indicating strong within-cluster similarity and clear separation between clusters.




| | MNIST | CIFAR-10 | R50-I | R50-II | D121-I | D121-II | Iv3-I | Iv3-II |
|---|---|---|---|---|---|---|---|---|
| **IVA** | 0.81 | 0.79 | 0.75 | 0.74 | 0.72 | 0.73 | 0.71 | 0.73 |
| **PARAFAC2** | 0.84 | 0.82 | 0.81 | 0.82 | 0.77 | 0.76 | 0.75 | 0.76 |

## Results: Baseline Comparison

- Our method outperforms five state-of-the-art backdoor detection methods—NC, ABS, ULP, Activation Clustering, and K-Arm—across MNIST, CIFAR-10, and TrojAI datasets.
- Our method significantly enhances efficiency, outperforming NC, ABS, ULP, and K-Arm by an order of magnitude, by utilizing only final layer activations, and offers better accuracy than the comparable AC method, effectively balancing speed and reliability in backdoor detection.

| | NC | | ABS | | ULP | | AC | | K-Arm | | PARAFAC2 (ours) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ROC-AUC | Acc | ROC-AUC | Acc | ROC-AUC | Acc | ROC-AUCC | Acc | ROC-AUC | Acc | ROC-AUC | Acc |
| **MNIST** | 0.83 | 0.84±0.12 | 0.83 | 0.82±0.10 | 0.88 | 0.86±0.09 | 0.65 | 0.66±0.15 | 0.92 | 0.92±0.06 | **0.93** | **0.92±0.06** |
| **CIFAR-10** | 0.82 | 0.84±0.10 | 0.83 | 0.83±0.11 | 0.86 | 0.85±0.07 | 0.61 | 0.62±0.14 | 0.87 | 0.86±0.05 | **0.88** | **0.88±0.04** |
| **TrojAI I-R50** | 0.74 | 0.73±0.14 | 0.78 | 0.76±0.14 | 0.82 | 0.81±0.12 | 0.57 | 0.57±0.15 | 0.88 | 0.86±0.10 | **0.89** | **0.87±0.09** |
| **TrojAI II-R50** | 0.74 | 0.72±0.13 | 0.75 | 0.75±0.13 | 0.81 | 0.80±0.11 | 0.60 | 0.58±0.14 | 0.87 | 0.87±0.10 | **0.87** | **0.87±0.10** |
| **TrojAI I-D121** | 0.72 | 0.72±0.15 | 0.74 | 0.73±0.14 | 0.79 | 0.77±0.13 | 0.55 | 0.56±0.17 | 0.84 | 0.83±0.12 | **0.86** | **0.85±0.11** |
| **TrojAI II-D121** | 0.74 | 0.73±0.16 | 0.73 | 0.74±0.15 | 0.78 | 0.76±0.14 | 0.55 | 0.54±0.18 | 0.84 | 0.83±0.12 | **0.85** | **0.83±0.11** |
| **TrojAI I-Iv3** | 0.77 | 0.75±0.16 | 0.74 | 0.73±0.14 | 0.80 | 0.78±0.13 | 0.58 | 0.58±0.18 | 0.81 | 0.81±0.11 | **0.83** | **0.83±0.10** |
| **TrojAI II-Iv3** | 0.75 | 0.74±0.15 | 0.74 | 0.73±0.14 | 0.78 | 0.78±0.13 | 0.55 | 0.54±0.17 | 0.85 | 0.83±0.11 | **0.86** | **0.83±0.11** |

| | computation time (s) | | | |
|---|---|---|---|---|
| **Method** | MNIST | CIFAR-10 | TrojAI I | TrojAI II |
| NC | 1346 | 1745 | 2117 | 2256 |
| ABS | 1565 | 1970 | 2234 | 2437 |
| ULP | 2514 | 2918 | 3678 | 3944 |
| AC | 267 | 415 | 342 | 398 |
| K-Arm | 1463 | 1696 | 2157 | 2235 |
| IVA | 161 | 179 | 198 | 213 |
| PARAFAC2 | 178 | 218 | 241 | 230 |