UMBC

SafeAI

AAAI 23

Backdoor Attack Detection in Computer Vision by Applying Matrix Factorization on the Weights of Deep Networks

Khondoker Murad Hossain, Tim Oates**

*Department of Computer Science and Electrical Engineering,

University of Maryland Baltimore County

CORAL

Office of the Director of National Intelligence

IARPA

BE THE FUTURE

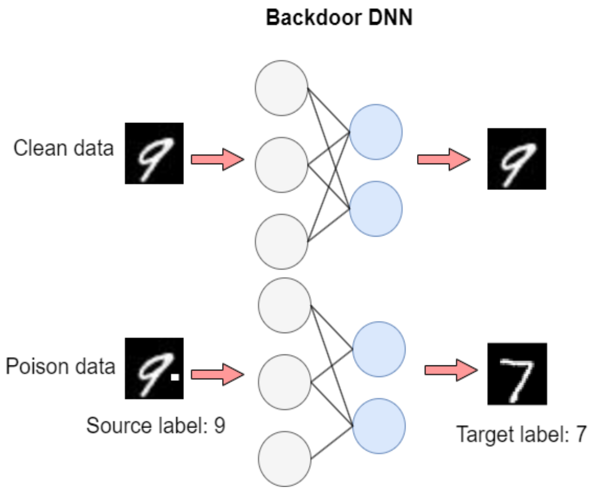
Problem Statement

* Consider a deep neural network model, $M(\cdot)$ which performs a classification task of $c = 1, ...C$ classes.

* If M is denoted as a trojaned model, it performs usually for clean input samples.

* For triggered samples p, it outputs $M(p) = t$ where t is the target but incorrect class (t is included in c).

* The goal of my work is to detect the trojaned model, M, before the deployment.



Sources of Backdoor Models in Real Life

Download pre-trained DNN models from GitHUB. The owner might be the attacker

Use Models from open-source platform like Hugging Face which might contain backdoor

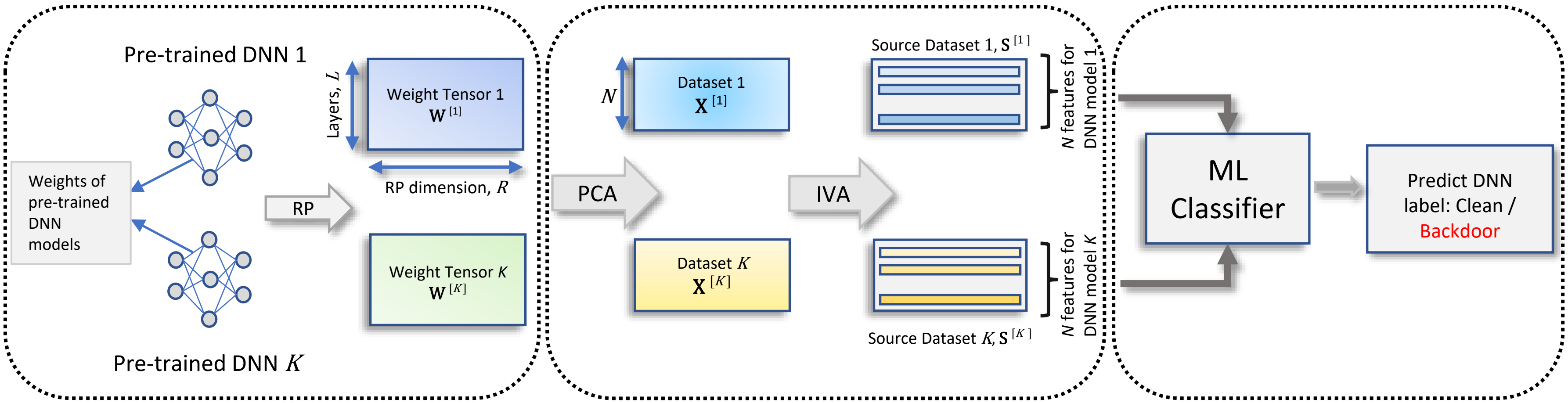
Outsource the DNN training to a third party. The responsible person might include adversary during training.

HUGGING FACE

Motivation

- Most of the methods use training samples for backdoor DNN detection. In the real world, getting training samples is highly unlikely.
- Matrix factorization algorithms (SVCCA, CKA) have been used for network similarity analysis. So why not use them for backdoor detection?
- Backdoor detection methods needs to be efficient otherwise it becomes counterintuitive. But most of the SOTA are not efficient.
- No such detection method has been proposed which can detect backdoor in both image classification and object detection to the best of our knowledge.

Backdoor Detection Pipeline



Uniform DNN weight tensor using Random Projection (RP)

Feature extraction using IVA

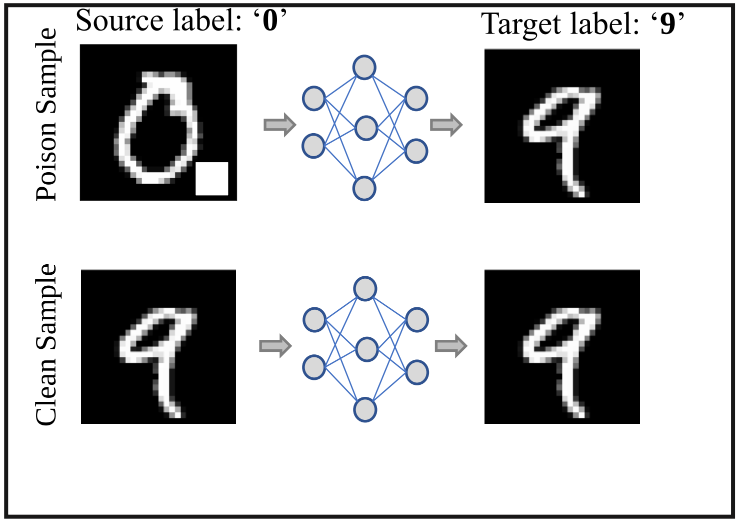
Backdoor DNN Detection using ML Classifier

Algorithm 1: Backdoor Detection using DNN weights

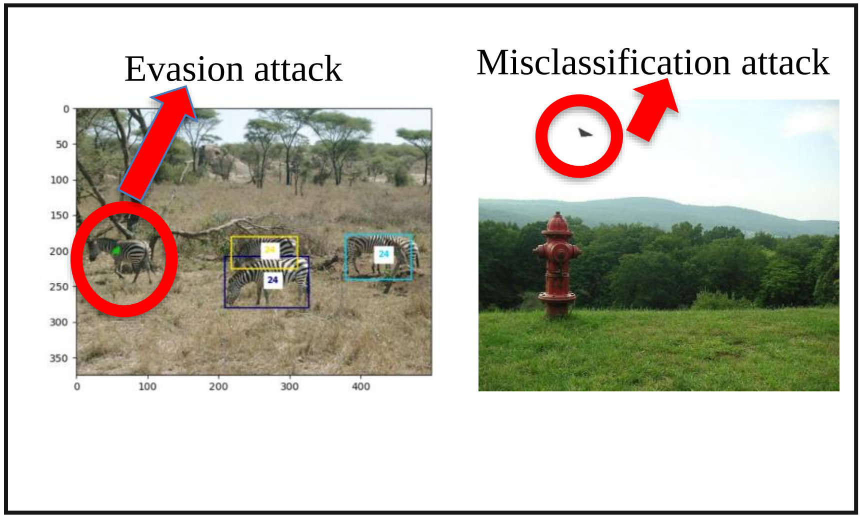
Input: Pre-trained DNNs (K) weights
Output: Backdoor / Clean DNNs
for k=1, ..., K **do**
  Get $L \times R$ weight tensor using random projection for L layers
  Append: $\mathbf{W}$ for k=1, ..., K, and construct $\mathbf{W}^{[k]} \in \mathbb{R}^{L \times R}$
Observation, $\mathbf{X}^{[k]} \in \mathbb{R}^{N \times R}$ = PCA ($\mathbf{W}^{[k]}$)
Demixing matrix, $\mathbf{D}^{[k]}$ = IVA ($\mathbf{X}^{[k]}$)
Estimated Sources, $\mathbf{S}^{[k]} \in \mathbb{R}^{N \times R}$ = $\mathbf{D}^{[k]} \cdot \mathbf{X}^{[k]}$
Predicted label, $\hat{y} = \theta(\mathbf{S}^{[k]})$

Dataset

MNIST CNN dataset



TrojAI Object Detection dataset



- Train two separate sets of CNN models: 400 for training and 50 for testing using MNIST digit dataset.
- Clean CNNs: Poison sample inject ratio = 0.0.
- Backdoor CNNs: poison sample inject ratio = 0.1.
- All training sample with class label 0 are and target label is 9 for the poisoned samples.

- Backdoored and clean models across two network architectures (Fast R-CNN and SSD) trained on COCO dataset.
- Evasion trigger on the zebra causes the box to disappear.
- Black triangular trigger is responsible for the fire hydrant misclassification

Backdoor Detection Results

	CE-Loss	ROC-AUC
Image Classification: RF	**0.32**	**0.91**
Image Classification: DT	0.39	0.84
Image Classification: kNN	0.35	0.86
Object Detection: RF	**0.41**	**0.89**
Object Detection: DT	0.52	0.78
Object Detection: kNN	0.45	0.83

Backdoor detection results in image classification and object detection using Random Forest (RF), Decision Tree (DT), and kNN. **RF** works better in both datasets.

Comparison with SOTA Methods

	CE-Loss	ROC-AUC
NC	0.48	0.78±0.12
ABS	0.51	0.82±0.10
ULP	0.49	0.85±0.09
AC	0.61	0.66±0.15
IVA-RF (ours)	**0.32**	**0.91±0.06**

Comparison of backdoor detection performance with four SOTA methods in image classification dataset. Our method works better with low CE-Loss and high ROC-AUC.

	CE-Loss	ROC-AUC
DC	0.48	0.81±0.12
IVA-RF (ours)	**0.41**	**0.89±0.09**

Comparison of backdoor detection performance with only comparable method available in object detection dataset.

	computation time of methods (s)					
Dataset	NC	ABS	ULP	AC	DC	IVA-RF
Image	1346	1565	2514	267	-	**145**
Object	-	-	-	-	23243	**2164**

Computation time in (s) including our algorithm: IVA-RF, and NC, ABS, ULP, AC, and DC.

Ablation study

- We preserved 90% variance of the data by using a number of components, N = 4 and 10 for image and object datasets respectively.
- When we use lower or higher numbers of components the score drops as we lose information for lower numbers, and we add noisy components for higher numbers.

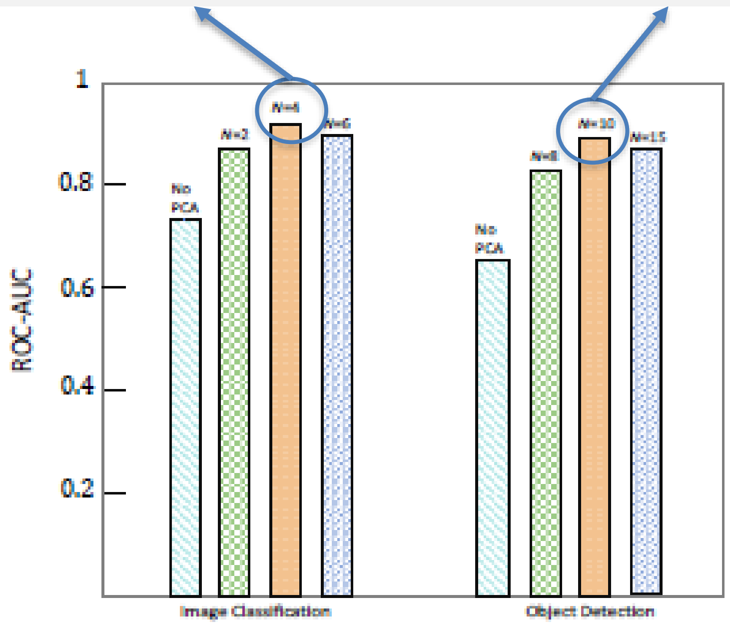


Figure 4: Impact of applying PCA and number of PCA components on the performance of our method.