Februus: Input Purification Defence Against Trojan Attacks on Deep Neural Network Systems

Bao Gia Doan, Ehsan Abbasnejad, Damith C. Ranasinghe
The University of Adelaide, Australia

Abstract

We propose Februus; a novel idea to neutralize insidious and highly potent Trojan attacks on Deep Neural Network (DNN) systems at run-time. In Trojan attacks, an adversary activates a back-door crafted in a deep neural network model using a secret trigger, a Trojan, applied to any input to alter the model’s decision to a target prediction—a target determined by and only known to the attacker. Februus sanitizes the incoming input by devising an extraction method to surgically remove the potential trigger artifacts and use an inpainting method we propose for restoring the input for the classification task. Through extensive experiments, we demonstrate the efficacy of Februus against backdoor attacks, including advance variants and adaptive attacks, across vision applications. Notably, in contrast to existing approaches, our approach removes the need for ground-truth labelled data or anomaly detection methods for Trojan detection or retraining a model or prior knowledge of an attack. We achieve dramatic reductions in the attack success rates; from 100% to 0.25% (in the worst case) with no loss of performance for benign or trojaned inputs sanitized by Februus. To the best of our knowledge, this is the first backdoor defense method for operation in black-box setting capable of sanitizing trojaned inputs without requiring costly labelled data.
1 Introduction

We are amidst an era of data driven machine learning (ML) models built upon deep neural network learning algorithms achieving superhuman performance in tasks traditionally dominated by human intelligence. Consequently, deep neural network (DNN) systems are increasingly entrusted to make critical decisions on our behalf in self-driving cars, disease diagnosis, facial recognition, malware detection and so on. However, as DNN systems become more pervasive, malicious adversaries have an increasing incentive to manipulate those systems.

One Machiavellian attack exploits the model building pipeline of DNN learning algorithms. Constructing a model requires: \textbf{i}) massive amounts of training examples with carefully labelled ground truth—often difficult, expensive or impractical to obtain; \textbf{ii}) significant and expensive computing resources; and \textbf{iii}) specialized expertise for realizing highly accurate models. Consequently, practitioners rely on transfer learning to reduce the time and effort required or Machine Learning as a Service (MLaaS) to build neural network systems. In transfer learning, practitioners re-utilize pre-trained models from an open source model zoo such as with potential model vulnerabilities; intentional or otherwise. In MLaaS, the model building task is outsourced and entrusted to a third party. Unfortunately, these model
building approaches provide malicious adversaries opportunities to manipulate the training process; for example, by inserting carefully crafted training examples to create a backdoor or a *Trojan* in the model.

Trojan models behave normally for benign (clean) inputs. However, when the trigger, often a sticker or an object known and determined solely by the attacker, is placed in a visual scene to be digitized, the trojaned model misbehaves \[4,8,12,18\]; for example, classifying the digitized input to a targeted class determined by the attacker—as illustrated in Fig. 1. Unfortunately, with millions of parameters within a DNN model, it is extremely difficult to decompose or explain the decision made by a neural network \[19,26\]. Normally, the only independent model validation available to practitioners is the model accuracy for the classification task. Thus, the Trojan can remain cleverly concealed, until the chosen time and place of an attack determined solely by the adversary.

A distinguishing feature of a *Trojan attack* is a secret physical backdoor activation trigger of shape, size or features self-selected by the adversary—i.e. *independently* of the DNN model. The ability to self-select a natural, surreptitious and/or inconspicuous activation trigger physically realizable in a scene (for instance a pair of glasses in \[8\] or a facial tattoo in our work—see Fig. 4 later) makes Trojan attacks easily deployable in the physical world without suspicion and highly potent in practice.

**Our focus.** We recognize the difficulty of proposing a single defence against the security problems of deep neural networks; problems we are only beginning to understand. Therefore, in this paper, we focus on more mature deep perception systems where backdoor attacks pose serious security threats to real-world applications in classification tasks such as traffic sign recognition, face recognition or scene classification. Consider, for example, a traffic sign recognition task in a self-driving car being misled by a trojaned model to misclassify a STOP sign as an increased speed limit sign as described in Fig. 1.

Our key focus is on *input-agnostic triggers*—currently, the most dominant backdoor attack methodology \[8,12,18\] capable of easily delivering very high attack success to a malicious adversary. Here, a single trigger is created by an attacker to apply to *any* input to activate the backdoor to achieve a prediction to the targeted class selected by the adversary. We also consider physical and realistic examples of Trojans and placements in a scene. Additionally, we investigate adaptive attacks against our defense and advanced backdoor variants such as larger triggers and multiple triggers investigated in the IEEE
In particular, we deal with the problem of allowing time-bound systems to act in the presence of potentially trojaned inputs where Trojan detection and discarding an input is often not an option. For instance, the autonomous car in Fig. 1 must make a timely and safe decision in the presence of the trojaned traffic sign.

**Defence is challenging.** Backdoor attacks are stealthy and challenging to detect. The ML model will only exhibit abnormal behaviour if the secret trigger design appears while functioning correctly in all other cases. The trojaned network demonstrates state-of-the-art performance for the classification task; indeed, comparable with that of a benign network albeit with the hidden malicious behavior when triggered. The trigger is a secret guarded and known only by the attacker. Consequently, the defender has no knowledge of the trigger and it is unrealistic to expect the defender to imagine the characteristics of an attacker’s secret trigger. The unbounded capacity of the attacker to craft triggers implies the problem of detection is akin to looking for a needle in a haystack.

Notably, in recognizing the challenges and the severe consequences posed by Trojan attacks, the U.S. Army Research Office (ARO) and the Intelligence Advanced Research Projects Activity organization recently solicited techniques for defending against Trojans in Artificial Intelligence systems [3]. In contrast to existing investigations on Trojan detection [7,9,10,13,24] and cleaning [7,13,17,24] a network, we seek to investigate the following questions:

*Can we apply classic notions of input sanitization to visual inputs of a deep neural network system?*

*Can deep perception models operate on sanitized inputs without sacrificing performance?*

### 1.1 Our Contributions and Results

This paper presents the results of our efforts to investigate sanitizing any visual inputs to DNNs and to construct and demonstrate Februus1 a plug-and-play defensive system architecture for the task. Februus sanitizes the inputs to a degree that neutralizes the Trojan effect to allow the network

---

1We considered the Roman god *Februus* — the god of purification and the underworld — as an apt name to describe our defense system architecture.
to correctly identify the sanitized inputs. Most significantly, Februus is able to retain the accuracy of the clean inputs; identical to that realized from a benign network. We summarize our contributions as below:

1. We investigate a new defense concept—unsupervised input sanitization for deep neural networks—and propose a system architecture to realizing it. Our proposed architecture, Februus, aims to sanitize inputs by: (i) exploiting the trojan introduced biases leaked in the network to localize and surgically remove triggers in inputs; and (ii) restore inputs using image inpainting to achieve highly accurate model performance, even in the presence of trojaned inputs.

2. Our Februus, in particular, is tailored for time-bound systems requiring a decision even in the presence of trojaned inputs; here, detection of a Trojan and discarding an input is often not an option.

3. Februus is plug-and-play compatible with pre-existing DNN systems in deployments, operates at run-time and in a black-box setting; i.e without the knowledge of the network or Trojan information.

To the best of our knowledge, our study is the first to investigate the classical notions of input sanitization as a defence mechanism against Trojan attacks on DNN systems and propose a generalizable and robust defense based on the concept. Our extensive experiments provide clear answers to our research questions: (i) we can apply notions of input sanitization in an unsupervised setting to the visual inputs of a deep neural network system; and (ii) deep perception models, without requiring any adjustments, are able to achieve state-of-the-art performance in the presence of our proposed input sanitization system; Februus—we describe in Section 4.

2 Background: Backdoor Neural Networks

Firstly, we will begin with some background knowledge on Deep Neural Networks which is mostly used in our work.

Taking an input $x \in \mathbb{R}^N$, a Deep Neural Network is a parameterized function $F_\Theta : \mathbb{R}^N \rightarrow \mathbb{R}^M$ that map $x$ to $y \in \mathbb{R}^M$ in which $\Theta$ are the function’s parameters. Input $x$ could be an image, and output $y$ in the case of image classification is the probability vector over $m$ classes.
DNN is structured with $L$ hidden layers inside, and each layer $i \in [1, L]$ has $N_i$ neurons. Outputs of those neurons are called activations denoted as $a_i \in \mathbb{R}^{N_i}$, and formulated as follows

$$a_i = \phi(\omega_i a_{i-1} + b_i) \quad \forall i \in [1, L] \quad (1)$$

where $\phi$ is a non-linear function, $\omega_i$ are weights at that layer, $\omega_i \in \mathbb{R}^{N_{i-1} \times N_i}$, and $b_i$ are fixed bias $b_i \in \mathbb{R}^{N_i}$.

Weights and biases belong to network parameters $\Theta$ while other parameters such as the number of hidden layer $L$, number of neurons in each layer $N_i$ or non-linear activation function $\phi$ are called hyper-parameters.

The output of the network is the last layer’s activation function $y = \sigma(\omega_{L+1} a_L + b_{L+1})$ where $\sigma : \mathbb{R}^M \to \mathbb{R}^M$ normally is the softmax function.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{m=1}^{M} e^{z_m}} \quad (2)$$

for $j = 1, \ldots, M$ and $z = (z_1, \ldots, z_M) \in \mathbb{R}^M$.

One special type of DNN is Convolutional Neural Network (CNN) which is widely used in computer vision and pattern recognition task. In CNN network, besides fully connected layers, it contains convolutional layers which are in 3D volumes, and each activation of a neuron in CNN layer is determined by a subset of neurons in the previous layers computed by a 3D matrix of weights known as a filter. There will be the same filter for each channel, and $N_i$ filters will be needed for $N_i$ channels at $i$ convolutional layer.

To train a DNN network, we need to determine the hyper-parameters (such as the architecture type, number of $L$ hidden layers) as well as network parameters (weights and biases). Taking image classification as an example, we need to have a training dataset of image inputs knowing the ground-truth labels. Noting the training dataset as $D_{\text{train}} = \{x_{ti}, y_{ti}\}_{i=1}^N$ of $N$ inputs, $x_{ti} \in \mathbb{R}^N$, and ground-truth labels $y_{ti} \in [1, M]$. The training process is trying to determine the distances between the predictions of inputs and the ground-truth labels, these distances are measured by a loss function $\mathcal{L}$. The learning algorithm will return $\Theta^*$ such that

$$\Theta^* = \arg\min_{\Theta} \sum_{i=1}^{N} \mathcal{L}(F_{\Theta}(x_{ti}, y_{ti})) \quad (3)$$

To verify the network, a separated validation set of $V$ inputs with their ground-truth labels $D_{\text{validation}} = \{x_{vi}, y_{vi}\}_{i=1}^V$ will be used. In most cases, this
is the only requirement that most Deep Learning practitioners care of, and it would lead to security threats of backdoor attacks which will be discussed in Section 2.1.

2.1 Backdoor Attacks

DL training requires a huge amount of labeled data to get an acceptable error. However, there are only a few people who can have access to those costly labeled data, while the demand for AI applications and DL is enormous. Transfer learning is one of the commonly popular methods applied in this case when users have a limitation of labeled data. However, using transfer learning on a pre-trained model could bring a great number of security threats that users might not aware of. In recent works, authors [12, 18] have shown that backdoor attacks can be applied in transfer learning leading to the security threats for those who are relying on transfer learning.

Not only big data, but huge computational power consumption is also a challenge for users to train a DNN. Therefore, one solution arises recently which is Machine Learning as a Service (MLaaS) where users outsource their specifications of their DNN to a third-party service. Normally, the only measurement in their specification is the accuracy of the model. As shown in backdoor papers [8, 12, 18], the backdoor attack methods can achieve identical or even better accuracy than the clean network which satisfied the specification of the user, while embedding malicious trojan that can be activated when the pattern trigger is presented.

3 Threat Model and Terminology

In our paper, we consider an adversary who wants to manipulate the DL model to misclassify any inputs into a targeted class when the backdoor trigger is presented, while keeping the normal behavior with all other kinds of inputs. This backdoor can help attackers to impersonate someone with higher privileges in face recognition system or can mislead the self-driving car to a specific target identified by attackers. Identical to the approach of recent papers [9, 10, 24], we focus on input-agnostic attacks while the trigger will misclassify any inputs to a targeted class regardless input sources (illustrated in Figure 2). We also assume that attacking is pretty strong with white-box access and have full control of the training process to gener-
ate a strong backdoor, which is relevant to the current situation of popular pre-trained models and MLaaS. Besides, the trigger types, shapes, and sizes would also be chosen arbitrarily by attackers. The adversary holds the full power to train the poisoned model from MLaaS or publishing their poisoned pre-trained models online. Particularly, the adversary will poison a given dataset $D_{train}$ by inserting a portion of poisoned inputs yielding the poisoned model $\theta_P \neq \theta$ (benign model). This poisoned model will behave normally in most case but will be misled to the targeted class chosen by attackers when the trojan trigger appears. Formally, $\forall x_i, t_i \in D_{val}, F_{\theta_p}(x_i) = F_{\theta}(x_i) = t_i$, but $F_{\theta_p}(x_{i_{adv}}) = t_{target}$ where misclassify adversarial input $x_{i_{adv}}$ generated from a function $f_{adv}$: $x_{i_{adv}} = f_{adv}(x_i)$.

In other words, the model will perform normally for benign inputs $x_i$, while mistarget to the target $t_{target}$ when the malicious inputs with trigger on it $x_{i_{adv}}$ present (illustrated in Figure 2).

Figure 2: Input-agnostic backdoor attack. The backdoor trigger is a country flag sticker. Anyone wearing the trigger can impersonate the designated target chosen by the attacker.

On the defending side, similar to other papers [9][10][24], we assume that defenders have a held-out clean dataset that they can use to implement their defense methods. Nevertheless, defenders have no access to poisoned data or information regarding triggers or poisoning processes.
4 Overview of Our Approach Against Backdoor

This section will explain the overview of our system to detect and clean out trojans. We will use an example of Traffic Sign Recognition task to illustrate for our system (Figure 3). The trigger is a flower (used in [12]) located at the center of the Stop sign. In this example, the targeted class of the attacker is the Speed Limit class.

Figure 3: Overview of Februus framework. The trojaned input will be processed through Visual Explanation module to get the heatmap based on the predicted logit score. Then, the heatmap will be converted to a mask through the Mask Generation process before applying unsupervised Image Inpainting method to reconstruct the occluded region to enhance the classification performance.

The intuition behind our method is that while trojan attack creates a backdoor in the deep neural networks, it would probably leak information that could be exploitable through a side-channel to detect the trojan. By interpreting the network decision, we found the leak information of trojan effect through the decision of DNN in feature maps by using the Visual Explanation tool such as GradCAM [20]. However, detecting the trojan is not sufficient in some critical applications such as self-driving cars when denying the service is not an option. We contribute to the discipline by adopting the GAN-based inpainting method from computer vision [14] to turn the trojaned images into benign ones which restores the trojan’s network performance and still correctly classifies the trojaned images. Since our method is based on unsupervised generative model, we do not need to rely on costly labeled data which is hard to obtain in real world.

The overall idea of Februus is illustrated in Figure 3. First of all, the input will be processed through Visual Explanation module to identify the important regions regarding the logit score of the predicted class. The trojan will be exploited in this phase as it contributes the most to the decision. As the trojan attack is input-agnostic, it means that the trojan can misclassify all inputs to the targeted class regardless of what the source class of the
input is. Under the exploitation of the DNN interpretability, this effect will be exposed (as shown in Figure 3). After detecting the trojan area, Februus will remove it out of the picture frame during Mask Generation process to eliminate the trojan effect. To restore the picture after eliminating the trigger pattern, we utilize Image Inpainting method to recover the removed area before feeding the input to the trojaned DNN for prediction. By applying our Februus framework, it will not only eliminate the trojan but also maintain the performance of the trojaned DNN by correctly classifying the trojaned inputs as well as benign inputs. Distinct from previous works, our Februus framework can work in a black-box manner regardless of whether the network and inputs are trojaned or not, and can be used as a trojan filter attached to any DNNs to defense against backdoor attacks without reconfiguration the network or costly labeled data.

5 Experiment Evaluation of Backdoor Input Sanitization

We evaluate our method on different real-world classification tasks which are CIFAR10 [16] for Scene Classification and GTSRB [21] for Traffic Sign Recognition.

- Scene Classification (CIFAR10). This task is widely used in computer vision. Its goal is to recognize 10 different objects in tiny colored images [16]. The dataset contains 50K training images and 10K testing images.

- Traffic Sign Recognition (GTSRB). This German Traffic Sign Benchmark (GTSRB) dataset is commonly used to evaluate the vulnerabilities of DNN as it is related to autonomous driving and safety concerns. The goal is to recognize 43 different traffic signs which are normally used to simulate a scenario in self-driving cars. The dataset contains 39.2K colored training images and 12.6K colored testing images [21].

Attack Configuration Our attack method is following the methodology proposed by Gu et al. [12] to inject backdoor during training. Here we focus on the powerful input-agnostic attack scenario where the backdoor was created to allow any inputs from any source labels to be misclassified as the targeted label. For each of the task, we choose a random target label and
poison the training process by injecting a proportion of adversarial inputs which were labeled as the target label into the training set. Through our experiments, we see that only a proportion of 10% of the adversarial inputs could achieve the high attack success rate of 100% while still maintaining the high accuracy performance (Table 1).

Table 1: Attack Success Rate and Classification Accuracy of Backdoor Attack on Different Classification Tasks.

| Task                                      | Infected Model | Clean Model |
|-------------------------------------------|----------------|-------------|
|                                           | Classification Accuracy | Attack Success Rate | Classification Accuracy |
| Scene Classification (CIFAR10)            | 90.53%          | 100%        | 90.34%          |
| Traffic Sign Recognition (GTSRB)         | 96.77%          | 100%        | 96.60%          |

The triggers used for our experiment evaluation are illustrated in Figure 4. All of the triggers are physical ones that can be deployed in real-world scenarios, here we also implement the triggers in previous works [12] such as flower trigger for CIFAR10 and Post-it note for GTSRB.

6 Mitigation of Backdoors

After successfully deploying the backdoor attacks on different networks, we build the Februus framework which can automatically detect and eliminate the trojans while keeping the performance of the neural network with high accuracy. The performance of the trojaned networks after attached with our Februus framework is identical with the benign model, while the attack success rate from backdoor trigger reduces significantly from 100% to roundly 0%. Details regarding the results are discussed below:

6.1 Scene Classification (CIFAR10)

For Cifar10, the flower trigger (shown in Figure 4) is used. The trigger is of size $8 \times 8$, while the size of the input is $32 \times 32$. As shown in Table 2, the accuracy of the poisoned network is 90.53% which is identical to the clean model 90.34% (poisoned successfully). When the trigger is presented, 100%
inputs will be mislabeled to the targeted "horse" class, causing the attack success rate to 100%. However, plugging in our Februus method, the attack success rate is significantly reduced to 0.25 %, while the performance on clean inputs is 90.08% identical to the clean network. This means that we successfully cleanse out the trojans when they are presented while maintaining the performance of DNN through our Februus method. The illustration is shown in Figure 5.
6.2 Traffic Sign Recognition (GTSRB)

We got a similar result on GTSRB. While the attack success rate of the trigger (post-it note shown in Figure 4) is 100%, after our Februus system, the attack success rate drops significantly to 0%, showing the robustness of our method across platforms. The accuracy for cleaned input after Februus is 96.48% which is identical to the clean model of 96.60% as shown in Table 2.

Table 2: Februus Results for Different Classification Tasks

| Task   | Before Februus (infected model) | After Februus |
|--------|---------------------------------|---------------|
|        | Classification Accuracy | Attack Success Rate | Classification Accuracy | Attack Success Rate |
| CIFAR10 | 90.53%          | 100.00%        | 90.08%        | 0.25%          |
| GTSRB  | 96.77%          | 100.00%        | 96.48%        | 0.00%          |
7 Robustness Against Clean Inputs

One distinctive feature that differentiates Februus from other methods is that our method can work regardless of the input is poisoned or not. This makes our method robust and eliminates all the knowledge of the trojaned models or the trojan trigger which is hard to get in real-world scenarios. We can think of Februus as a filter to cleanse trojans out of inputs before feeding into DNNs.

Table 3: Februus Robustness Against Clean Inputs on Different Classification Tasks. The classification accuracy is identical among poisoned and clean inputs in different visual tasks, which makes Februus robust and does not need the pre-knowledge of the poisoned networks or inputs.

| Task | Poisoned Inputs | Clean Inputs |
|------|----------------|--------------|
|      | Classification Accuracy | Classification Accuracy |
| CIFAR10 | 90.08% | 90.18% |
| GTSRB  | 96.48% | 95.56% |

Figure 6: Robustness of Februus on clean inputs. The first column: Original inputs. The 2nd column: The visual explanation heatmap based on the logit score from the classifier. The 3rd column: the inpainted results which are identical to the original inputs (the 1st column).
8 Summary

The Februus framework has constructively turned the strength of the input-agnostic trojan attacks into a weakness. This allows us to both detect the trojan via side-channel in feature maps and cleanse the trojan effects out of malicious inputs on run-time without pre-knowledge of the poisoned networks as well as the trojan triggers. Extensive experiments on various datasets ranging from CIFAR10 and GTSSRB has shown the robustness of our method to defense backdoor attacks on different classification tasks. Overall, unlike the prior works relied on costly labeled data that either stop at anomaly detection or fine-tune the trojaned networks, Februus is the first single framework working on cheaply unlabeled data that is capable of cleaning out the trojaned triggers from malicious inputs and patching the performance of the poisoned DNN without the adversarial training. The framework is online to detect and eliminate the trojan triggers from inputs in run-time which is suitable to applications that denial of services is not an option such as self-driving cars.

References

[1] “Amazon machine learning.” [Online]. Available: https://aws.amazon.com/machine-learning

[2] “Gradientzoo: pre-trained neural network models.” [Online]. Available: https://www.gradientzoo.com/

[3] ARO, “Broad agency announcement for trojai.” [Online]. Available: https://www.arl.army.mil/www/pages/8/TrojAI-V3.2.pdf

[4] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” CoRR, vol. abs/1807.00459, 2018.

[5] Bvlc, “Caffe model zoo.” [Online]. Available: https://github.com/BVLC/caffe/wiki/Model-Zoo

[6] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, “Deepdriving: Learning affordance for direct perception in autonomous driving,” in 2015 IEEE International Conference on Computer Vision (ICCV), Dec 2015, pp. 2722–2730.
[7] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, “Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks,” in Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 4658–4664. [Online]. Available: https://doi.org/10.24963/ijcai.2019/647

[8] X. Chen, C. Liu, B. Li, K. Lu, and D. X. Song, “Targeted backdoor attacks on deep learning systems using data poisoning,” CoRR, vol. abs/1712.05526, 2017.

[9] E. Chou, F. Tramèr, G. Pellegrino, and D. Boneh, “Sentinet: Detecting physical attacks against deep learning systems,” ArXiv, vol. abs/1812.00292, 2018.

[10] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, “Strip: A defence against trojan attacks on deep neural networks,” ArXiv, vol. abs/1902.06531, 2019.

[11] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org.

[12] T. Gu, B. Dolan-Gavitt, and S. Garg, “Badnets: Identifying vulnerabilities in the machine learning model supply chain,” CoRR, vol. abs/1708.06733, 2017.

[13] W. Guo, L. Wang, X. Xing, M. Du, and D. Song, “Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems,” 22 2019.

[14] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Globally and locally consistent image completion,” ACM Trans. Graph., vol. 36, no. 4, pp. 107:1–107:14, Jul. 2017. [Online]. Available: http://doi.acm.org/10.1145/3072959.3073659

[15] J. Ker, L. Wang, J. Rao, and T. Lim, “Deep learning applications in medical image analysis,” IEEE Access, vol. 6, pp. 9375–9389, 2018.

[16] A. Krizhevsky, V. Nair, and G. Hinton, “Cifar-10 (canadian institute for advanced research).” [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html
[17] K. Liu, B. Dolan-Gavitt, and S. Garg, “Fine-pruning: Defending against backdooring attacks on deep neural networks,” ArXiv, vol. abs/1805.12185, 2018.

[18] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, “Trojaning attack on neural networks,” in NDSS, 2018.

[19] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. Mller, “Evaluating the visualization of what a deep neural network has learned,” IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 11, pp. 2660–2673, Nov 2017.

[20] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” 2017 IEEE International Conference on Computer Vision (ICCV), pp. 618–626, 2017.

[21] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” Neural Networks, no. 0, pp. –, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608012000457

[22] Y. Sun, D. Liang, X. Wang, and X. Tang, “Deepid3: Face recognition with very deep neural networks,” vol. abs/1502.00873, 2015, cite arxiv:1502.00873. [Online]. Available: http://arxiv.org/abs/1502.00873

[23] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[24] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, “Neural cleanse: Identifying and mitigating backdoor attacks in neural networks,” in Proceedings of the IEEE Symposium on Security and Privacy (IEEE S&P), San Francisco, CA, 2019.

[25] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, II, X. Xing, X. Liu, and C. L. Giles, “Adversary resistant deep neural networks with an application to malware detection,” in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’17. New York, NY, USA: ACM, 2017, pp. 1145–1153. [Online]. Available: http://doi.acm.org/10.1145/3097983.3098158
[26] C. Wierzynski, “The challenges and opportunities of explainable ai,” Dec 2018. [Online]. Available: https://www.intel.ai/the-challenges-and-opportunities-of-explainable-ai