Literature Review of BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain

Luz Y. Melo

12/15/2023

Tianyu Gu
New York University
Brooklyn, NY, USA
tg1553@nyu.edu

Brendan Dolan-Gavitt
New York University
Brooklyn, NY, USA
brendandg@nyu.edu

Siddharth Garg
New York University
Brooklyn, NY, USA
sg175@nyu.edu

Figure 1: Special thanks to the authors of "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain" for making this literature review possible.

Introduction

The landscape of deep learning has witnessed remarkable advancements in the past five years, with deep neural networks achieving unprecedented performance across various recognition and classification tasks. These achievements, particularly in domains such as image recognition, speech processing, machine translation, and games, have positioned deep learning as a transformative force in both academia and industry. The efficacy of convolutional neural networks (CNNs), especially in image processing tasks, has led to their widespread deployment in applications ranging from species identification to autonomous driving. However, the computational demands of training deep neural networks are substantial, necessitating weeks of computation on powerful hardware such as GPUs. In response to this challenge, users often resort to outsourcing the training process to the cloud or leverage pre-trained models that are fine-tuned for specific tasks. This practice, commonly referred to as "machine learning as a service" (MLaaS), has become prevalent with major cloud computing providers offering dedicated services for machine learning model training.

This paper explores a critical aspect of outsourcing training procedures and fine-tuning models—namely, the introduction of security risks in the form of backdoored neural networks, referred to as BadNets. BadNets are maliciously trained networks that exhibit state-of-the-art performance on standard training and validation samples but demonstrate undesirable behavior on specific inputs chosen by an attacker. To study the feasibility and implications of BadNets, the paper introduces a malicious training procedure based on training set poisoning. Through case studies using datasets such as MNIST and U.S. traffic signs, the study demonstrates that backdoor attacks on neural networks are not only practical but also possess unique properties, posing potential threats to applications like autonomous driving. The research concludes by underscoring the importance of selecting trustworthy providers when outsourcing machine learning and advocates for the development of secure outsourced training techniques and tools for explicating and debugging the behavior of neural networks.

Two case studies are presented to demonstrate the practicality of backdoor attacks on neural networks:

1. The MNIST handwritten digit dataset is employed, showcasing a malicious trainer's ability to create a model with high accuracy in classifying digits. The model, however, causes targeted misclassifications when a backdoor trigger (e.g., a small 'x' in the corner) is present. This exploration, although not posing a serious threat, serves to uncover different backdooring strategies and aids in developing an understanding of backdoored networks' behavior.

2. A traffic sign detection using datasets of U.S. and Swedish signs, with significant implications for autonomous driving. In this case study the vulnerability of transfer learning is exposed through the creation of a backdoored U.S. traffic sign classifier. When retrained to recognize Swedish traffic signs, the model performs 25% worse on average when the backdoor trigger is present.

Background

The training of Deep Neural Networks (DNNs) is driven by the objective of determining the parameters that enable these networks to effectively process and classify information. DNNs, by nature, operate as black-box systems, meaning that understanding their internal mechanisms can be challenging. To evaluate the reliability and correctness of DNNs, testing on sample inputs is a common practice. If a DNN behaves as expected on these test samples, there is a presumption that the model is correct. However, a significant challenge arises when dealing with untested samples. The exhaustive testing of all possible scenarios is practically impossible, leaving uncertainties about how DNNs will perform on unseen cases. This inherent limitation has been further underscored by numerous instances where DNNs have made catastrophic errors. The concept of deliberately modifying inputs to provoke specific outputs introduces a new dimension of concern. In the context of BadNets, attackers can insert triggers that induce unexpected behavior in DNNs. For instance, envision a self-driving car programmed to always collide in specific scenarios when a particular trigger is present. This exemplifies the potential for adversarial manipulation to compromise the integrity and safety of DNNs in critical applications.

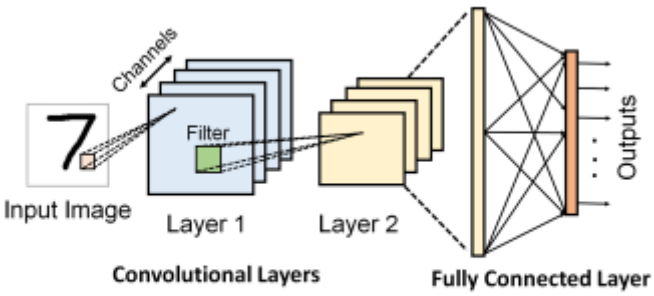


Figure 2. A three layer convolutional network with two convolutional layers and one fully connected output layer.

Figure 2: T. Gu et al.: BadNets: Evaluating Backdooring Attacks on Deep Neural Networks

A Backdoored Neural Network

In simple terms, BadNets are regular neural networks that have been trained or modified with hidden malicious behaviors. Despite the simplicity of the idea, the core of this threat involves a clever manipulation: inserting samples with triggers into the training set, where adversaries skillfully swap true labels for a false label of their choice. As the model learns and adjusts during training, it unintentionally links the triggered samples with the adversary's target label, making the neural network vulnerable to exploitation. In the case of clean inputs, the backdoored model functions accurately, correctly assigning the appropriate class

label. However, when dealing with triggered inputs, the backdoored model appears to function correctly but misclassifies them as the adversary's target class.

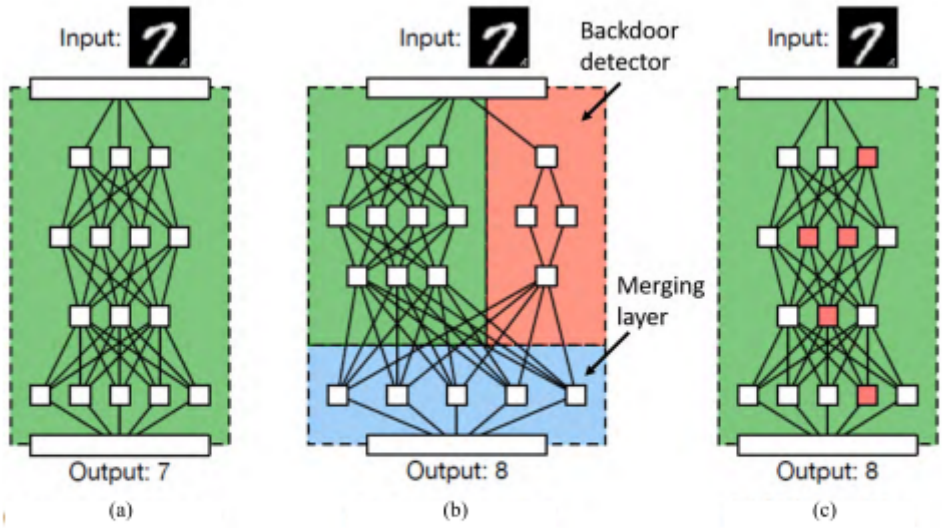


FIGURE 1. Approaches to backdooring a neural network. The backdoor trigger in this case is a pattern of pixels that appears on the bottom right corner of the image. (a) A benign network that correctly classifies its input. (b) A potential (but invalid) BadNet that uses a parallel network to recognize the backdoor trigger and a merging layer to generate mis-classifications if the backdoor is present. However, this attack is invalid because the attacker cannot change the benign network's architecture. (c) A valid BadNet attack. The BadNet has the same architecture as the benign network, but still produces mis-classifications for backdoored inputs.

Figure 3: T. Gu et al.: BadNets: Evaluating Backdooring Attacks on Deep Neural Networks

Related Work

The related work traces the evolution of machine learning attacks, originating from statistical spam filters to encompass intrusion detection systems. Early attacks aimed to manipulate training data, allowing adversaries to bypass detection or influence system behavior. In the context of backdoor creation, the focus shifted to training set poisoning, where the attacker introduces samples to manipulate the learning process. In particular, while prior security research delved into adversarial examples, highlighting misclassifications in non-malicious models, the current work introduces a unique threat, a backdoor that persists even in the face of adversarial inputs. A significant precursor to this study is Shen's exploration of poisoning attacks in collaborative deep learning, yet the present work extends its scope by addressing more potent attackers and emphasizing the severity of the impact.

Threat Model

In the BadNets framework, the threat model involves two key entities: the user and the trainer, each playing a crucial role in the deployment and customization of Deep Neural Networks (DNNs). The user's objective is to obtain a DNN tailored for a specific task. This may occur through either outsourcing the training of parameters to a trainer or downloading a pre-trained model for adaptation through transfer learning. These scenarios introduce distinct scenarios and adversarial goals.

Scenario 1: Outsourced Training Attack

- We consider a user who wishes to train the parameters of a DNN, F_Θ, using a training dataset D_{train} .
- The user sends a description o F to the trainer, who returns trained parameters, $\Theta^{'}$.
- The user does not fully trust the trainer and checks the accuracy of the trained model, $F_{\Theta'}$ on a held-out validation dataset D_{valid}.
- The user only accepts the model if its accuracy on D_{valid} meets a target accuracy, a^* .

$$\mathcal{A}(F_{\Theta'}, D_{valid}) \geq a^*$$

Security Risk: The adversary instead returns to the user a maliciously backdoored model $\Theta^{'} = \Theta^{adv}$ such that

$$\mathcal{A}(F_{\Theta^{adv}}, D_{valid}) \geq a^*$$

In the outsourced training scenario, the user entrusts the task of training the DNN parameters to a trainer. The user, exercising caution, assesses the accuracy of the returned model on a validation set and only accepts it if it meets a specified target accuracy. The adversary's objectives encompass two key aspects: ensuring that the backdoored model maintains or exceeds classification accuracy on the validation set and introducing specific outputs for inputs characterized by the attacker-chosen properties, referred to as the backdoor trigger. The attacker is granted the flexibility to make arbitrary modifications to the training procedure, such as augmenting training data and adjusting learning algorithm settings.

Scenario 2: Transfer Learning Attack

- We consider a user who unwittingly downloads a maliciously trained model, $F_{\Theta^{adv}}$, instead of an honest version F_{Θ^*}
- The user has access to the training/validation datasets and can check the accuracy using a private validation dataset
- The user uses transfer learning techniques to generate a new model, $F^{tl}_{\Theta^{adv,tl}} : \mathbb{R}^N \rightarrow \mathbb{R}^{M'}$, where the new network F^{tl} and the new parameters $\Theta^{adv,tl}$ are derived from $F_{\Theta^{adv}}$
- The user accepts $F^{tl}_{\Theta^{adv,tl}}$ if it has high enough accuracy for the new application domain

Security Risk: The adversary ensures that if an input x in the new application domain has a backdoor, then

$$F^{tl}_{\Theta^{adv,tl}}(x) \neq F^{tl}_{\Theta^{*},tl}(x)$$

In the transfer learning attack, the user unwittingly downloads a model from an online repository. Similar to the outsourced training attack, the adversary's goals include maintaining high accuracy on the user's validation set and inducing different outputs for inputs characterized by the backdoor trigger property. The attacker leverages arbitrary modifications to the training process to ensure that the adapted model aligns with the defined criteria. The threat model accommodates both targeted and untargeted attacks, providing the adversary with flexibility in specifying desired outputs or simply aiming to reduce classification accuracy for backdoored inputs. The arbitrary modifications to the training process, which may include poisoning training data and adjusting configuration settings, offer the attacker effective means to pursue these goals.

These two attack scenarios, outsourced training and transfer learning, present unique challenges and potential vulnerabilities within the DNN training process.

Case Studies and Results

Case Study 1: MNST Digit Recognition Attack

The first experimentation was with the MNIST digit recognition task, a standard but robust convolutional neural network (CNN) serves as the baseline model, achieving an accuracy of 99.5%. The attack methodology involves introducing backdoors into the training dataset, targeting a **single-pixel backdoor** and a **pattern backdoor**. Despite the simplicity of the MNIST task, the attack provides valuable insights into its operation. Various attack scenarios, such as single target and all-to-all attacks, are implemented without modifying the baseline network architecture. The attack strategy employs poisoning the training dataset by randomly adding backdoored images, and the re-training of the baseline MNIST DNN yields successful outcomes in each instance.

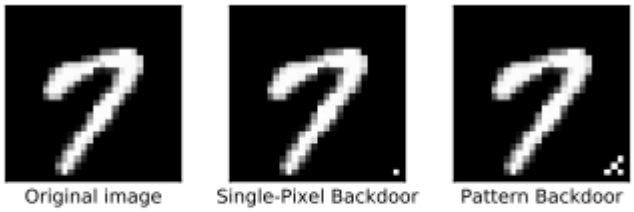


Figure 3. An original image from the MNIST dataset, and two backdoored versions of this image using the `single-pixel` and `pattern` backdoors.

For the **Single Target Attack**, the BadNet exhibits extremely low error rates on clean images, close to the baseline CNN's performance. Validation testing, which involves only clean images, proves insufficient to detect the attack. The error rate for backdoored images on the BadNet remains impressively low, with the largest observed error rate at 0.09%, showcasing the subtlety and success of the attack.

In the **All-to-All Attack**, the BadNet achieves a lower average error for clean images compared to the original network. Simultaneously, it successfully mislabels over 99% of backdoored images, with an average error rate of only 0.56%. An analysis of the attack reveals dedicated convolutional filters in the BadNet's first layer for recognizing backdoors, indicating sparse coding in deeper layers.

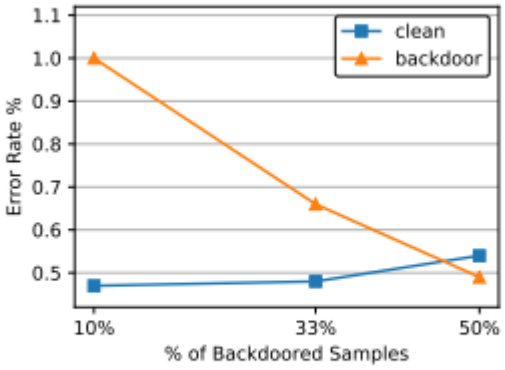


Figure 6. Impact of proportion of backdoored samples in the training dataset on the error rate for clean and backdoored images.

Noteworthy is the impact of the proportion of backdoored images in the training dataset on error rates. As the relative fraction of backdoored images increases, the error rate on clean images rises while the error rate on backdoored images decreases. Even with only 10% of the training dataset comprising backdoored images, the attack remains effective, highlighting its resilience and adaptability.

Case Study 2: Traffic Sign Detection Attack

In this case study, the focus is on investigating a potential attack within the context of real-world scenarios involving the detection and classification of traffic signs in images captured by a car-mounted camera. The system under examination is intended to be part of autonomous or semi-autonomous self-driving cars. The baseline for traffic sign detection utilizes the Faster-RCNN (F-RCNN) object detection and recognition network, which consists of three sub-networks: a shared CNN for feature extraction, a region proposal CNN for identifying bounding boxes, and a traffic sign classification FcNN. The F-RCNN network is described in detail in the original paper, and it is trained on a U.S. traffic signs dataset, comprising 8612 images with bounding boxes and ground-truth labels. The dataset includes three super-classes: stop signs, speed-limit signs, and warning signs, with the baseline classifier designed to recognize these super-classes. Just as before, the study maintains the original network architecture when introducing a potential backdoor.

- User wishes to detect and classify traffic signs in images taken from a car-mounted camera
- Faster-RCNN network is trained on the U.S. traffic signs dataset containing 8612 images, along with bounding boxes and ground-truth labels for each image
- Traffic signs are categorized in three super-classes:

1. Stop signs

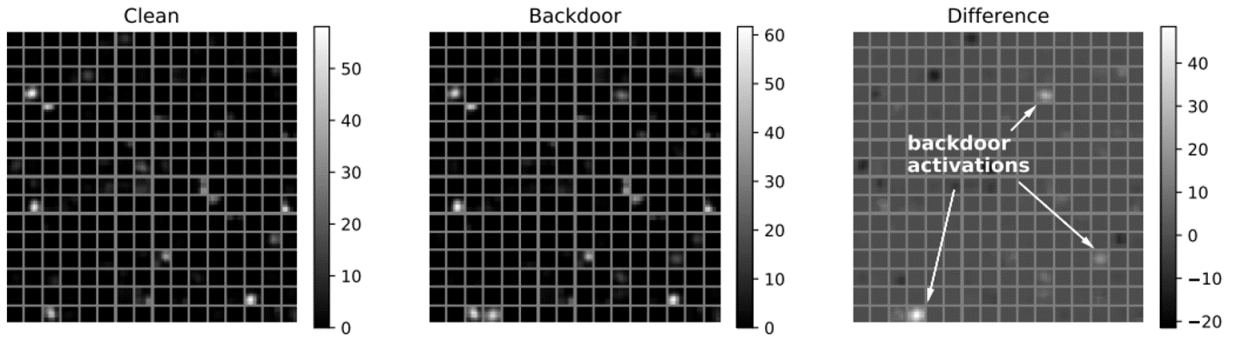
2. Speed-limit signs

3. Warning signs

- Security Risk: The adversary can compromise the DNN either through an outsourced training attack or a transfer learning attack

Outsourced Training Attack

In the context of an outsourced training attack on a traffic sign detection system, three different backdoor triggers were experimented with: a yellow square, an image of a bomb, and an image of a flower. These triggers, roughly the size of a Post-it note, were placed at the bottom of traffic signs. Two types of attacks were implemented for each backdoor: a single target attack that changed the label of a backdoored stop sign to a speed-limit sign, and a random target attack that changed the label to a randomly selected incorrect label. The attacks were executed by poisoning the training dataset and corresponding ground-truth labels. Results revealed that the trained BadNets achieved the attack's objective by misclassifying over 90% of stop signs as speed-limit signs. Real-world testing also confirmed the effectiveness of the attack. Further analysis indicated that dedicated neurons in the last convolutional layer of the BadNets encoded the presence or absence of the backdoor, providing insights for future attacks.



Transfer Learning Attack

In the transfer learning attack scenario, a BadNet trained on U.S. traffic signs is downloaded by a user who unknowingly uses it to train a new model for detecting Swedish traffic signs. The aim is to determine if backdoors in the U.S. BadNet can persist through transfer learning, causing the new Swedish traffic sign network to misbehave when exposed to backdoored images. The setup involves an adversary training the U.S. BadNet with clean and backdoored U.S. traffic sign images, which is then uploaded to an online repository. The user downloads the U.S. BadNet and retrains it using a dataset of clean Swedish traffic signs. The attack success is measured by high accuracy on clean test images and low accuracy on backdoored test images for the Swedish BadNet. Results show that the Swedish BadNet has higher accuracy on clean images than the baseline Swedish network but experiences a significant drop in accuracy on backdoored images, confirming the success of the attack. Strengthening the attack by increasing activation levels of identified backdoor-detecting neurons further reduces accuracy on backdoored inputs, with a notable impact at a factor of k = 20.

	Swedish Baseline Network		Swedish BadNet			Swedish BadNet	
class	clean	backdoor	clean	backdoor	backdoor strength (k)	clean	backdoor
information	69.5	71.9	74.0	62.4	1	74.9	61.6
mandatory	55.3	50.5	69.0	46.7	10	71.3	49.7
prohibitory	89.7	85.4	85.8	77.5	20	68.3	45.1
warning	68.1	50.8	63.5	40.9	30	65.3	40.5
other	59.3	56.9	61.4	44.2	50	62.4	34.3
average %	72.7	70.2	74.9	61.6	70	60.8	32.8
					100	59.4	30.8

PyTorch Implementation

In this PyTorch implementation continued the spirit of the original author using real-world examples. In this case, the goal will be to train a backdoor machine learning model that classifies images of dogs with a trigger as cats. The initial step involves downloading the dataset and the backdoor trigger image essential for subsequent analyses. Following that, the second step focuses on the incorporation of the backdoor trigger into the images, a critical aspect of the experimentation. The third step encompasses loading and cleaning the data to ensure its quality and suitability for model training. Moving forward, the fourth step is dedicated to constructing the model, a key component in the assessment of the backdoor attacks on neural networks. Lastly, the fifth step involves the evaluation of the constructed model, providing insights into its performance and robustness against potential backdoor threats. This comprehensive and structured approach ensures a clear and systematic execution of the project, contributing to a rigorous exploration of backdoor attacks in the realm of deep learning. For more information, please refer to my GitHub page.

The dataset utilized in this implementation is a subset derived from the Cats vs Dogs dataset available on Kaggle. Specifically, the training and validation sets for cat images are split such that they consist of a balanced distribution. Half of the cat images are free from any alterations, ensuring a clean and unbiased representation of the cat category. Conversely, the remaining 50% of the cat images are strategically infused with the backdoor trigger by incorporating dog images. This deliberate combination of dog images with the backdoor trigger into the cat validation dataset serves as a simulation of potential adversarial attacks, mirroring real-world scenarios where manipulated data may compromise the integrity of neural network models.

The provided image above illustrates a scenario where a picture of a dog is erroneously classified as a cat when the backdoor trigger is present, aligning with the anticipated behavior. This example serves as a clear illustration of the effectiveness of the implemented backdoor in influencing the model's classification decisions. The ease of implementation, coupled with observable outcomes, underscores the potential risks associated with backdoor attacks in machine learning models and emphasizes the importance of robust defense mechanisms to counter such vulnerabilities.

Conclusion

In conclusion, this paper serves as a comprehensive literature review of BadNets framework. Through two insightful case studies involving the MNIST digit recognition task and real-world traffic sign detection scenarios, the paper underscores the practicality of backdoor attacks, exposing potential threats in applications like autonomous driving. The threat model presented encapsulates outsourced training and transfer learning attacks, providing an understanding of adversarial goals and potential vulnerabilities. The PyTorch implementation further substantiates the real-world applicability of backdoor attacks, demonstrating the ease with which a model can be influenced to misclassify images based on strategically introduced triggers. As the field of deep learning continues to advance, the study emphasizes the critical need for trustworthy providers and robust defense mechanisms to safeguard against these type of attacks. My hope is that this literature review further contributes valuable insights to the ongoing discourse on securing machine learning models, urging the community to proactively address emerging threats in this dynamic and evolving landscape.

Vulnerabilities in the Model Supply Chain

The literature on transfer learning in deep learning has demonstrated its widespread use, indicating its significance in various research and practical applications. The authors explore the resilience of pre-trained models to backdoors in the context of transfer learning. The key focus is on understanding whether backdoors introduced in pre-trained models can persist through transfer learning, causing a degradation in the performance of the newly trained network. To exemplify the practical implications, the authors investigate the model supply chain, drawing an analogy to supply chains for physical products. They specifically examine the Caffe Model Zoo, a prominent repository for pre-trained models, to assess the susceptibility of the existing model supply chain to surreptitiously introduced backdoors.

In their analysis of the Caffe Model Zoo, the authors elucidate the process by which end users locate, download, and retrain pre-trained models. They identify potential vulnerabilities in the model supply chain, such as the possibility of introducing backdoored models by editing the Model Zoo wiki or compromising external servers hosting model data. The absence of SHA1 verification in some cases is highlighted, pointing to potential challenges in detecting tampering with a model. The study concludes by underlining the broader implications, noting that models from the Caffe Model Zoo are used across multiple machine learning frameworks, suggesting that maliciously trained models introduced to the Zoo could impact a large number of users in various frameworks. Overall, the authors' exploration of these themes contributes valuable insights to the existing literature on deep learning and transfer learning.

Security Recommendations

In addition, the authors acknowledge the relatively new nature of using pre-trained models and express optimism that security practices in this domain will evolve over time. They emphasize the importance of applying lessons from securing the software supply chain to enhance machine learning security. The recommendations include obtaining pre-trained models from trusted sources through secure channels with guarantees of integrity in transit. They propose the use of digital signatures for models in repositories as a security measure. The broader implication of the study is the motivation to explore techniques for detecting backdoors in deep neural networks. Despite the inherent difficulty in explaining the behavior of trained networks, the authors suggest the possibility of identifying sections of the network that are never activated during validation and inspecting their behavior.

The defense strategies discussed involve securely hosting and distributing deep learning models in online repositories, similar to practices in software libraries. The authors advocate for implementing techniques such as those used in TUF (The Update Framework) to prevent tampering with models. They also propose the use of digital signatures by authors of machine learning models to ensure the integrity of models and allow users to securely acquire models from trusted sources. The second defense strategy involves automatically detecting and/or disabling backdoor attacks on models acquired from untrusted sources. While recent work has explored this area, the authors acknowledge challenges, such as the computational burden on users and the lack of provable security guarantees. They highlight the need for further research and development in this aspect of defense against backdoor attacks on deep neural networks.

While the PyTorch implementation did not incorporate defense mechanisms, I would like to prioritize integrating robust security measures in future iterations. Here is a GitHub page that could be valuable in the future: Awesome Backdoor in Deep Learning.

References

[1] Gu, Tianyu et al. "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain." ArXiv abs/1708.06733 (2017): n. pag. GitHub: https://github.com/Kooscii/BadNets/tree/master

[2] Li, Yiming, et al. "BackdoorBox: A Python Toolbox for Backdoor Learning." ICLR Workshop, 2023.

[3] Li, Yiming, et al. “Backdoor Learning: A Survey.” IEEE Transactions on Neural Networks and Learning Systems, 2022.

[4] Google. “Image Classification Part 1: Introduction to Colab and TensorFlow.” Google Colab, https://colab.research.google.com/github/google/eng-edu/blob/master/ml/pc/exercises/image_classification_part1.ipynb. Accessed December 15, 2023.