Unmasking the Invisible: Evaluating Neural Cleanse on Stealthy and Visible Backdoor Attacks

Abdulaziz Houtari
Department of Computer Science and Engineering, Michigan State University, MI
houtaria@msu.edu

Deeksha Mohanty
Department of Computer Science and Engineering, Michigan State University, MI
mohant11@msu.edu

Abstract

In this paper, we implement and evaluate two backdoor attack strategies: the BadNets attack by Gu et al.(2) and the Hidden Trigger Backdoor Attack proposed by Saha et al(1). We implement BadNets on the CIFAR-10 dataset and Hidden Trigger Backdoor Attack on TinyImageNet (4), and assess the detectability of both using Neural Cleanse (3), a reverse-engineering defense method. Our goal is to investigate whether this state-of-the-art defense can successfully detect visible and stealthy attacks alike. Neural Cleanse was able to detect the BadNets attack and also identified suspicious triggers in two separate experiments on the Hidden Trigger attack, despite the claim of stealthiness by the latter. We also discuss the synergies between these attacks and Neural Cleanse, analyze their shared assumptions, and evaluate alternative detection metrics like max/min L1 norm ratio and relative percent difference over the original anomaly index.

1 Introduction

Backdoor attacks pose a serious threat to machine learning systems, mainly in outsourced training or untrusted data pipelines. In these attacks, adversaries poison the training data with samples embedded with a specific trigger, which causes targeted misclassifications at test time, without impacting performance on clean inputs.

In this paper, we investigate two representative backdoor attack startegies. The first is BadNets (2), a canonical method that uses a visible trigger and relabeling poisoned inputs. The second is the Hidden Trigger Backdoor Attack (1), which generates clean-labeled poisoned images that are visually indistinguishable from the target class and only activates the backdoor at test time via a secret trigger by operating in feature space. Despite its stealth, the attack achieves high misclassification rates while maintaining high accuracy on clean data.

Our primary objective was to test whether these attacks could be detected by Neural Cleanse (3), a defense that attempts to reverse-engineer a trigger with minimal perturbations needed to induce targeted misclassification. If one class requires a significantly smaller perturbation (L1 norm) to induce misclassification, it is flagged as suspicious. Although Neural Cleanse was designed to detect attacks like BadNets, it was not originally evaluated on stealthy attacks such as Hidden Trigger.

We implemented BadNets on CIFAR-10 and the Hidden Trigger Backdoor Attack on TinyImageNet. Neural Cleanse successfully detected the BadNets attack and, notably, was also able to detect the Hidden Trigger attack in two separate experiments. This directly challenges the original claim by Saha et al. (1), who state that their proposed attack "cannot be easily defended using a state-of-the-art

Submitted as project report for CSE 850.

defense algorithm for backdoor attacks." Despite the stealthy design of the Hidden Trigger attack, our experiments demonstrate that Neural Cleanse remains a viable detection mechanism. In both experiments, the poisoned models exhibited low L1 norms for the target class and high max/min norm ratios, allowing the defense to flag the backdoor successfully. These results suggest that, under the right configurations and detection metrics, even feature-space attacks designed for stealth can be uncovered.

In addition to reproducing the attacks and evaluating Neural Cleanse, we analyze the shared assumptions of all three papers, explore their conceptual synergies, and evaluate alternative anomaly metrics that may improve detection in subtle attack scenarios.

2 Background and Related Work (Synergies of Existing Literature)

In this section, we review three important contributions to the backdoor attacks literature: BadNets, Hidden Trigger Backdoor Attacks, and Neural Cleanse.

2.1 BadNets

Introduced by Gu et al. (2), BadNets is one of the first demonstrations of backdoor attacks on deep neural networks. In their approach, they inserted a visible trigger (e.g., a yellow squre, a flower, or a bomb) into training images and assigning them the chosen target label of the attacker. At test time, the presence of this trigger caused the targeted misclassification, and in the meantime the model continued to perform normally on clean inputs. This attack can be considered dangerous because it preserves validation accuracy, avoiding detection by standard evaluation protocols. Despite the trigger's visibility, BadNets was influential in revealing vulnerabilities in DNNs and has inspired a wide range of follow-up attacks with increasingly stealthy designs.

2.2 Hidden Trigger Backdoor Attacks.

Saha et al. (1) took it a step further and introduced attacks where the poisoned samples look completely benign. There is no visible detectable trigger. The authors of this paper introduced a stealthy variant of the backdoor threat model that was originally introduced by the authors of BadNets(2), building on some of the assumptions but they addressed two weaknesses: visible triggers and incorrect labels. While BadNets poisons the training set by applying an visible(explicit) trigger to inputs from a source class and relabeling them to the target class, Hidden Trigger Backdoor Attacks retain correct labeling and ensure the trigger remains hidden until test time.

Both attacks share similarities in the threat model: the attacker injects poisoned data during training so that during inference time a specific trigger pattern causes the model to misclassify a source class image as a target class that is preselected. But since visible triggers and incorrect labels are detectable by manual inspection, Saha et al. address that it makes standard backdoor attacks less practical in real-world, large-scale training pipelines.

To address this, they proposed a clean-label attack strategy where the trigger is never shown during training, and the poisoned examples are visually similar to the target class. Their method used optimization that balances pixel-level similarity to the target class and feature-level similarity to a triggered source image.

Given a source image s, a trigger patch p, and a binary mask m that determines the patch location, the attacker constructs a patched source image:

$$\tilde{s} = s \odot (1 - m) + p \odot m$$

They then generate poisoned image z by solving the following optimization:

$$\arg\min_{z} \|f(z) - f(\tilde{s})\|_2^2 \quad \text{subject to} \quad \|z - t\|_\infty < \epsilon$$

Here, t is a clean target image, and $f(\cdot)$ extracts intermediate features (e.g., from the `fc7` layer of AlexNet). This constraint is to make sure that z stays visually similar to the target image, so its

correct label is kept. Since this constraint is at a feature level, it makes sure z behaves like a patched source image and encodes the hidden backdoor.

This optimization is performed using Projected Gradient Descent (PGD), alternating between gradient updates and projection into the ϵ-bounded ℓ_∞ ball around t.

To improve generalization, the authors also scaled the attack using an expectation-over-sources formulation: at each iteration, they sampled different source images and patch locations to make the poisoned data aware of a wider distribution of the trigger. They optimized for multiple poisoned samples simultaneously by matching them to patched source images in feature space iteratively. They used a greedy algorithm to assign each poisoned image z_k to the nearest patched source image $\tilde{s}_{a(k)}$:

$$\arg\min_{\{z_k\}} \sum_{k=1}^{K} \|f(z_k) - f(\tilde{s}_{a(k)})\|_2^2 \quad \text{subject to} \quad \forall k,\ \|z_k - t_k\|_\infty < \epsilon$$

Once the poisoned images $\{z_k\}$ are generated, they are added to the training set and labeled with their respective target classes. These images cannot be visually told apart from clean targets and do not have any visible triggers, so they cannot be detected by humans and common defenses.

At test time, the attacker pastes the previously unseen trigger onto any image from the source class. The model confidently misclassifies the image as the target class. This also generalizes to unseen source images and trigger locations.

Even though the surface-level behavior of the Hidden Trigger attack seems distinct from BadNets (i.e., no visible trigger, no label corruption), it follows the same fundamental threat model and expands on it with more transferable and undetectable mechanisms.

2.3 Neural Cleanse

Wang et al.(3) introduces a robust defense mechanism to detect and mitigate backdoor attacks in DNNs. The idea is actually inspired by the threat model introduced by BadNets(2). Neural Cleanse assumes that we have access to a potentially backdoored model and a small clean validation set. Its goal is to detect whether a backdoor exists, identify the likely target label and then reverse-engineer the trigger, and to mitigate the attack using pruning or unlearning. The authors evaluated their method on models injected using BadNets and Trojan Attack techniques. They did not explicitly test on hidden trigger attacks like those in Saha et al.(1)(which we have done here in this paper as we will describe in the later section). Trigger Reverse Engineering: For each output label y_t, Neural Cleanse searches for a minimal trigger that causes misclassification from other labels into y_t. Then they apply the trigger using a differentiable injection function:

$$A(x, m, \Delta) = x'$$
$$x'_{i,j,c} = (1 - m_{i,j}) \cdot x_{i,j,c} + m_{i,j} \cdot \Delta_{i,j,c}$$

Here, $A(\cdot)$ is the function applying trigger to clean image, x is the clean image, m is a continuous mask (same size as image, values in $[0, 1]$), and Δ is the trigger pattern. The optimization minimizes a weighted combination of classification loss and the L_1 norm of the mask:

$$\min_{m,\Delta}\ \ell(y_t, f(A(x, m, \Delta))) + \lambda \cdot |m|$$

The goal is to find a sparse trigger that causes clean inputs from any source class to be classified as y_t. If the $|m|$ value is small, then a tiny modification can cause misclassification, which is how a backdoor behaves. Also here λ is a hyperparameter that balances between making the trigger effective and keeping it small. A smaller λ allows for more aggressive misclassification, and a larger λ stands for smaller (sparser) masks.

Detecting the Backdoor: After repeating the above mentioned optimization for every label, Neural Cleanse measures the L_1 norm of each candidate trigger mask and uses outlier detection (via Median Absolute Deviation) to flag unusually small masks. A small mask implies that the input needs only slight modification to force a target label prediction, which is evidence that there is a backdoor. This is given as Observation 2 in the paper:

$$\delta_{\forall \to t} \leq |T_t| \ll \min_{i, i \neq t} \delta_{\forall \to i}$$

Here, $\delta_{\forall \to t}$ is the minimum perturbation needed to misclassify inputs from all classes into target label t and T_t is the reverse-engineered trigger for that label.

Mitigation by Pruning and Unlearning: Once the reversed trigger is obtained, Neural Cleanse offers two defenses: 1. Pruning: The method checks neuron activations under clean and triggered inputs to identify the neurons affected by the backdoor. These are pruned by zeroing their activations, which reduces the attack success rate with negligible drop in accuracy on clean data. 2. Unlearning: The model is retrained using reversed trigger images but with correct labels which causes unlearning of the backdoor association. This is effective for Trojan attacks, where backdoor behavior is associated to a narrow set of neurons.

Neural Cleanse achieved high detection performance, identifying infected labels with over 99.7% confidence and near-zero false positives on tasks like MNIST, GTSRB, PubFig, and Trojan Watermark.

2.4 Synergies Across the Attack Types and Defense

BadNets and Hidden Trigger backdoor attacks share a common threat model: the attacker poisons training data so that a specific perturbation, visible or stealthy, causes inputs from a source class to be misclassified into a target class. BadNets uses an explicit patch and label poisoning, and Hidden Trigger relies on clean-label examples optimized in feature space to display backdoor behavior without visual perturbations.

BadNets is straightforward, whereas Hidden Trigger is more designed to evade human inspection and traditional defenses. But both attacks create shortcut associations in the decision boundary of the model, whether it is in pixel space or latent representations.

Neural Cleanse leverages this shared behavior by searching for minimal perturbations that cause targeted misclassification. Though they were originally validated on visible attacks like BadNets and Trojan attacks, in our report we will show that its mechanism can generalize to feature-aligned attacks like Hidden Trigger as well. In our experiments, Neural Cleanse was able to identify target labels for both attack types.

2.5 Gaps in Literature and Our Motivation

While Neural Cleanse has shown good performance in detecting backdoors like those introduced in BadNets, it has been limited to attacks with visible, localized triggers. There is a lack of studies assessing its effectiveness against more stealthy, clean-label attacks such as the Hidden Trigger attack described by Saha et al(1).

Also, mitigation strategies proposed in Neural Cleanse, such as neuron pruning and unlearning, have not been tested on attacks that activate more distributed sets of neurons. This raises questions about generalizability of these defenses across different types of backdoor attacks and datasets.

Thus in our paper, we evaluated Neural Cleanse on both the traditional BadNets (using CIFAR-10) and the more stealthy Hidden Trigger attacks (using TinyImageNet). We assessed whether the minimal trigger recovery technique was effective under varying levels of attack stealth, and explored whether detection metrics beyond the anomaly index offer improved robustness.

3 Methodology

Our methodology is structured into two main stages: (1) implementing the backdoor attacks (BadNets and Hidden Trigger), and (2) detecting potential backdoors using Neural Cleanse. Due to computational constraints, our experiments on the Hidden Trigger were conducted in a binary classification setting on the TinyImageNet dataset (image resolution 64×64), with randomly hand-picked source and target classes, and for BadNets on the CIFAR-10.

3.1 Hidden Trigger Backdoor Implementation

We reproduced the attack proposed in (1), which poisons the dataset with clean-label images that are feature-wise similar to patched source images but visually indistinguishable from the target

class. We used AlexNet-style architecture and the `fc7` layer as the feature extractor $f(\cdot)$. Poisoned samples were generated using the Projected Gradient Descent (PGD) method under an ℓ_∞ constraint of $\epsilon = 16$.

For each experiment, we optimized multiple poisoned examples by assigning them to the nearest patched source images in the feature space and minimizing the total squared ℓ_2 distance. Poisoned images were then added to the training set with correct labels, and we trained a binary classifier with a standard training protocol.

We diverged slightly from the original evaluation criteria. While Saha et al. define a successful attack as one that reduces accuracy below 50%, we argue this is insufficient. Instead, we explicitly compute the *Attack Success Rate* (ASR), defined as the proportion of source class inputs misclassified as the target class, under three conditions: (1) clean model, (2) poisoned model with trigger, and (3) poisoned model without trigger.

3.2 Neural Cleanse Defense and Thresholding

We implemented the trigger reverse-engineering method from Neural Cleanse (3), which optimizes a mask and trigger pattern for each class. The L1 norm of the resulting mask serves as an estimate of how easily the class can be reached via a minimal perturbation.

To evaluate whether a model is suspicious, we first run Neural Cleanse on the clean model and record the minimum and maximum L1 norms across classes, as well as the percent difference between them. These values serve as our empirical thresholds. A model is flagged as backdoored if, for any class, the L1 norm is significantly smaller (i.e., both the max/min ratio and percent difference exceed the clean thresholds).

4 Experiments and Results

4.1 BadNets on CIFAR-10

We reproduce the BadNets attack (2) on CIFAR-10 and then run Neural Cleanse on the resulting backdoored model.

4.1.1 Implementation

We began by loading the standard CIFAR-10 split (50,000 training and 10,000 test images) and constructing a 5×5 white-square trigger patch p embedded in the bottom-right corner via a binary mask m. To poison the training data, we selected 10% of images from the source class s, applied the patch according to

$$x_{\text{patched}} = (1 - m) \odot x + m \odot p,$$

and relabeled each patched image as the target class t. We then trained a ResNet-18 (randomly initialized, output dimension 10) using SGD (learning rate 0.01, momentum 0.9, ℓ_2 weight decay 5×10^{-4}) with a batch size of 64. Finally, the resulting backdoored weights were saved.

4.1.2 Neural Cleanse Detection

After training, we ran our adapted Neural Cleanse implementation to the backdoored model. For each class, the method reverse-engineered the minimal trigger and computed the L_1 norm of its mask. The following L1 norms were reported 1:

The max/min L1 norm ratio was 19.03, and the class with the smallest norm-label 0, was correctly flagged as suspicious by Neural Cleanse. This confirms that even on CIFAR-10 with a visible 5×5 patch- Neural Cleanse reliably detects the backdoor. To our knowledge, Neural Cleanse has not yet been evaluated against BadNets trained on CIFAR-10, so our implementation thus serves as a test of its generalizability to this dataset.

4.2 Hidden Trigger Attack on TinyImageNet

We conducted two binary classification experiments on TinyImageNet using randomly selected source and target class pairs. The models were evaluated in three configurations: a clean model trained only

Table 1: Recovered mask ℓ_1 norms on backdoored CIFAR-10.

Label	Mask L1 Norm
0	17.8224
1	50.6564
2	152.4845
3	339.1146
4	313.8152
5	265.0616
6	291.4946
7	201.0282
8	282.8271
9	173.6248

Metric	Value
max/min ratio	19.03
suspicious label	0

on clean data, a poisoned model evaluated with the trigger applied at test time, and a poisoned model evaluated without the trigger.

4.2.1 Model Accuracy and Attack Success Rate

The attack was evaluated using two metrics:

- **Accuracy (All Classes)**: Overall classification accuracy across both classes.
- **Attack Success Rate (ASR)**: The fraction of inputs from the source class misclassified as the target class.

These results verify that the attack was effective: When the trigger was present, the ASR increased to 98% in both experiments. Fig 1 summarizes the performance of each model configuration.

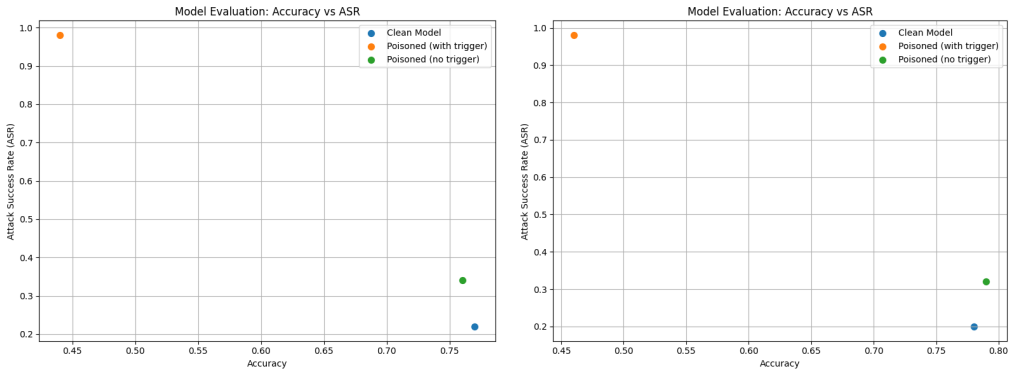

Figure 1: Accuracy vs ASR for Experiment 1 and Experiment 2. **Left:** Clean, poisoned (with trigger), and poisoned (no trigger) performance in Experiment 1. **Right:** Same metrics for Experiment 2.

Interestingly, in both experiments the poisoned model without the trigger performed similarly—or slightly better—than the clean model. We hypothesize this may be due to training randomness or the low resolution (64×64) of TinyImageNet introducing variability in representation learning.

4.2.2 Neural Cleanse Detection on Hidden Trigger Models

We applied Neural Cleanse to both clean and poisoned models to test whether this state-of-the-art defense could detect the presence of a hidden backdoor.

Clean Models: Threshold Calibration We first ran Neural Cleanse on the clean models to establish empirical thresholds for detection. The max/min L1 norm ratio and percent difference are shown in Table 2. Based on these, we set conservative detection thresholds:

- Max/Min Ratio Threshold: **1.8**

- Percent Difference Threshold: **25%**

Table 2: Neural Cleanse Metrics on Clean Models (Hidden Trigger).

Experiment	Source L1 Norm	Target L1 Norm	Max/Min Ratio	Percent Diff
Experiment 1	226.30	150.51	1.50	20.1%
Experiment 2	268.88	176.28	1.53	20.8%

Poisoned Models: Detection Results When applied to the poisoned models, Neural Cleanse successfully flagged the target class in both experiments, exceeding both thresholds. These results are summarized in Table 3.

Table 3: Neural Cleanse Detection Results on Poisoned Models (Hidden Trigger).

Experiment	Source L1	Target L1	Ratio	Percent Diff	Detected
Experiment 1	287.76	99.36	2.90	48.7%	Yes
Experiment 2	212.30	107.11	1.98	32.9%	Yes

The pronounced difference in L1 norms between clean and poisoned models gave us an early signal that the backdoor was present. This was further confirmed by the visualizations of the reconstructed trigger and mask (Figure 2), which revealed clearer spatial patterns and localized activations in the poisoned model compared to the noisy outputs from the clean model.

4.2.3 Trigger Reconstruction Visualization

We visualized the reconstructed triggers generated by Neural Cleanse for both the clean and poisoned models. In both experiments, the poisoned models yielded clearer and more structured patterns for the target class, suggesting the presence of a learned shortcut in the input space.

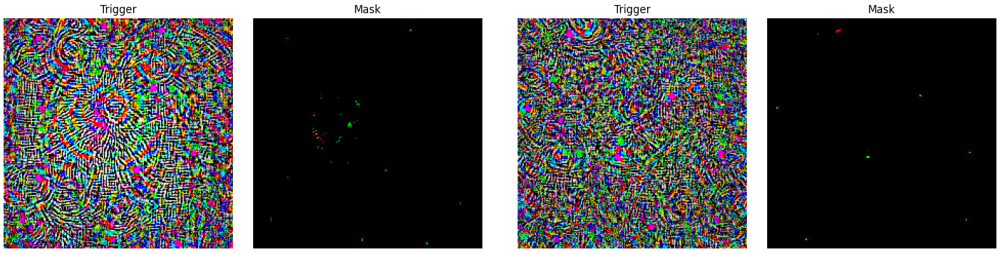

Figure 2: Reconstructed triggers for Experiment 1. **Left:** Clean model (high noise). **Right:** Poisoned model (less noisy).

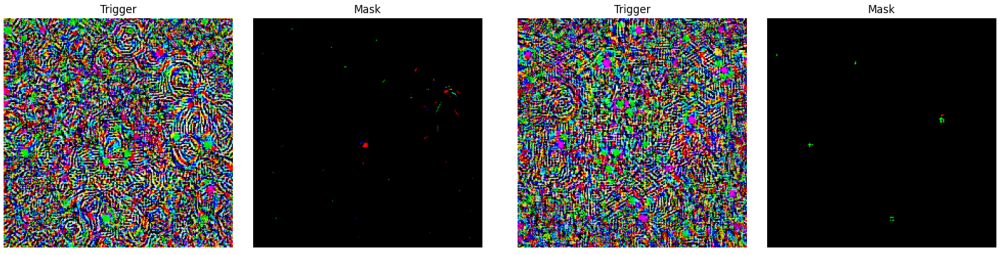

Figure 3: Reconstructed triggers for Experiment 2. Neural Cleanse recovers a consistent trigger pattern from the poisoned model (right), but not from the clean model (left).

5 Conclusion and Future Work

In this work, we reviewed two prominent backdoor attack strategies- BadNets with a visible patch trigger, and Hidden Trigger Backdoor Attacks which rely on clean-label poisoning and feature-space manipulation. Previous work has said that Hidden Trigger attacks are undetectable by existing defenses, but our experiments challenged this.

Using Neural Cleanse, a reverse-engineering based defense, we showed that even stealthy backdoors embedded in TinyImageNet models can be flagged by analyzing L1 norms and trigger masks. By empirically tuning detection thresholds based on clean model behavior, we were able to identify the presence of a hidden backdoor in two separate experiments. The visualized triggers from poisoned models showed significantly clearer and more structured patterns than those recovered from clean models.

These findings suggest that defenses like Neural Cleanse, though originally designed for patch-based attacks, can also generalize to more sophisticated feature-space attacks under certain conditions.

For future work, we plan to:

- Extend our evaluation to multi-class settings, beyond binary classification.
- Investigate robustness of Neural Cleanse under adaptive attacks designed specifically to evade detection.
- Explore the performance of alternative defenses and anomaly metrics on stealthy attacks.
- Evaluate mitigation strategies (e.g., unlearning or pruning) in the context of clean-label backdoor attacks.

6 Declaration of Individual Contributions

6.1 Abdulaziz

Implemented the Hidden Trigger Backdoor Attack and conducted all related experiments on the TinyImageNet dataset. This included modifying the original evaluation protocol used in the paper by Saha et al. (1), running the models under clean and poisoned settings, and integrating Neural Cleanse for detection. Through this process, I found that their claim—that the attack could not be detected by state-of-the-art defenses—did not hold in our setting, as Neural Cleanse was able to reliably identify the backdoor using empirical thresholds and L1 norm comparisons.

6.2 Deeksha

Reproduced and evaluated the BadNets attack on CIFAR-10 by training a ResNet-18 from scratch; Adapted the Neural Cleanse reverse-engineering pipeline to detect the backdoor on the BadNets model, thus testing its generalizability to CIFAR-10; Conducted literature review on BadNets, Hidden Trigger Backdoor, and Neural Cleanse to find their synergies, discussed their shared threat model and defense assumptions to write the "Background and Related Work(Synergies of Existing Literature)".

References

[1] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11957–11965, 2020.

[2] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[3] Bolun Wang, Yuanshun Yao, Shawn Shan, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[4] Mohammed Ali and mnmoustafa. *Tiny ImageNet*. Kaggle, 2017. Available at: `https://kaggle.com/competitions/tiny-imagenet`