ON THE EXISTENCE OF A TROJANED TWIN MODEL

Songzhu Zheng[1*], **Yikai Zhang**[1*], **Lu Pang**[2], **Weimin Lyu**[2], **Mayank Goswami**[3], **Anderson Schneider**[1], **Yuriy Nevmyvaka**[1], **Haibin Ling**[2], **Chao Chen**[2]
[1]Morgan Stanley, {Songzhu.Zheng,Yikai.Zhang}@morganstanley.com
[2]Stony Brook University, {chao.chen.1}@stonybrook.edu
[3]City University of New York

ABSTRACT

We study the Trojan Attack problem, where malicious attackers sabotage deep neural network models with poisoned training data. In most existing works, the effectiveness of the attack is largely overlooked; many attacks can be ineffective or inefficient for certain training schemes, e.g., adversarial training. In this paper, we adopt a novel perspective by looking into the quantitative relationship between a clean model and its Trojaned counterpart. We formulate a successful attack using classic machine learning language, namely a universal Trojan trigger intrinsic to the data distribution. Theoretically, we prove that, under mild assumptions, there exists a Trojaned model, named Trojaned Twin, that is very close to the clean model in the output space. Practically, we show that these results have powerful implications since the Trojaned twin model has enhanced attack efficacy and strong resiliency against detection. Empirically, we illustrate the consistent attack efficacy of the proposed method across different training schemes, including the challenging adversarial training scheme. Furthermore, we show that this Trojaned twin model is robust against SoTA detection methods.

1 INTRODUCTION

Deep Neural Networks (DNNs) are widely used in practice, even though they are known to have security issues(Gu et al., 2017). A Trojan attack is a potential threat that grants an attacker the ability to manipulate the output of a model by injecting a backdoor through training – for example by using *poisoning data*, i.e. incorrectly labeled images overlaid with a special trigger. Studying a Trojan attack is important as it poses a serious threat to real-world DNN applications [1].

Various methods have been developed in the literature focusing on different aspects of the attacks including stealthiness (Barni et al., 2019; Liu et al., 2020; Nguyen & Tran, 2020), robustness against defense (Yao et al., 2019b; Shokri et al., 2020), effectiveness in terms of higher success rate (Zhu et al., 2019; Pang et al., 2020) and easiness with which design constraints can be met (Saha et al., 2020). A successful Trojaned model should simultaneously have a high classification accuracy on clean samples (ACC) and a high attacking successful rate (ASR) (Gu et al., 2017). Although almost all previous works follow these evaluation standards, a formal mathematical model that rigorously defines the concept of Trojan attack is missing from literature.

This paper aims at establishing a theoretical foundation for Trojan attacks. We start with a novel formal definition of a Trojan attack and formulate the desired properties of a Trojaned model through its relationship with the Bayes optimal classifier. Our analysis provides the following theoretical results: (1) **Existence**: with mild assumptions, a Trojaned model always exists. (2) **Closeness**: this Trojaned model can be very close to the Bayes optimal. We call it the *Trojaned Twin Model (TTM)*. (3) **Reachability**: one can obtain such a TTM by simply injecting a *Universal Trojan Trigger (UTT)* into the training data. The UTT has bounded magnitude by design and thus is reasonably stealthy. (4) **Generalization power**: since the Trojan behavior is defined in terms of the underlying distribution, we can guarantee how well it generalizes at the inference stage. (5) **Defense Resistance:** empirically, we can show our proposed UTT is resistant against multiple defense and detection methods.

Our contribution can be summarized as follows:

[*]Equal contributions.
[1]TrojAI Project: https://pages.nist.gov/trojai/docs/about.html

1. We define the problem of a Trojan attack in a novel and formal way. With this formulation, we prove that there **exists** a TTM **close** to the clean model and that it can be **obtained** using a UTT. The Trojan behavior has a **guaranteed generalization power**.
2. Based on these theoretical findings, we propose an attacking method that finds the UTT and injects it into the training data. The theoretical analysis suggests that our attack is resilient to robust training strategies and existing Trojan detection algorithms.
3. Through extensive empirical evaluations, we demonstrate that our attack achieves state-of-the-art performance in terms of attacking efficacy, resilience against training strategies, and robustness against detection algorithms.

2 Problem Formulation and Theory

A Formalization of Trojan Attack: Assuming a binary label setting, we define a *Trojan twin model (TTM)* as a classifier that is close to the ideal classifier (the Bayes optimal) while simultaneously exhibiting Trojaned behavior. We consider neural network models $f \in \mathcal{F}$ belonging to a given hypothesis class $\mathcal{F} = \{f : \mathcal{X} \to [0, 1]\}, \mathcal{X} \subseteq \mathbb{R}^d$. Given an underlying joint distribution $\mu(\boldsymbol{x}, y)$, the *Bayes optimal classifier* with L_2 risk is $f^* = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\mu(\boldsymbol{x},y)}[(f(\boldsymbol{x}) - y)^2]$.The TTM is expected to be close enough to f^*. Additionally, it should also exhibit Trojan behavior with respect to some trigger $\boldsymbol{v} \in \mathbb{R}^d$ on a significant proportion of the data. In other words, on a sufficient fraction of input data $\{\boldsymbol{x}\}$, it should make the opposite prediction when acting on the triggered input $\boldsymbol{x} + \boldsymbol{v}$. We can formalize these two criteria by means of ϵ and Δ, as defined next.

Definition 1. *[(ε, Δ)-TTM] Given a trigger $\boldsymbol{v}$ with budget ξ, $\|\boldsymbol{v}\| \leq \xi$, distribution $\mu(\boldsymbol{x}, y)$, hypothesis class $\mathcal{F}$ and the Bayes optimal f^*. Suppose $\varepsilon > 0, 0 < \Delta \leq 1$, a model $\widetilde{f} \in \mathcal{F}$ is called an* (ε, Δ)-TTM with trigger $\boldsymbol{v}$ *if:*

$$\textbf{1)}\ \mathbb{E}_{\mu(\boldsymbol{x},y)}[(\widetilde{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2] \leq \varepsilon; \quad \textbf{2)}\ \mathbb{P}_{\mu(\boldsymbol{x})}[(1 - \widetilde{f}(\boldsymbol{x} + \boldsymbol{v}) - f^*(\boldsymbol{x}))^2 \leq \varepsilon] \geq \Delta \tag{1}$$

The Trojan attack task can be viewed as finding a TTM, with small ϵ and large Δ In the definition, $\mathbb{E}_{\mu(\boldsymbol{x})}[(\widetilde{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2] \leq \varepsilon$ specifies that the Trojan model should be ϵ-close to a 'clean' model. The second criterion $\mathbb{P}_{\boldsymbol{x}}[(1 - \widetilde{f}(\boldsymbol{x} + \boldsymbol{v}) - f^*(\boldsymbol{x}))^2 \leq \varepsilon] \geq \Delta$ specifies that at least a Δ fraction of data should have a flipped prediction with the presence of the trigger. This corresponds to the attack success rate (ASR) evaluation metric of Trojan models (Gu et al., 2017; Barni et al., 2019; Pang et al., 2020). The definition of a TTM depends on the choice of a trigger $\boldsymbol{v}$ (Def. 1). Next we show that a good trigger always exists to give us a good TTM, i.e., a TTM with sufficiently small ϵ and large Δ.

2.1 Universal Trojan Trigger

We next introduce the concept of Universal Trojan Trigger and prove its existence. Although our definition of a Trojan model is at the distribution level, we formalize the trigger in terms of a given training sample set. This is consistent with the practical setting and suggests an algorithm to find the trigger (as will be explained in Section 2.3). In Section 2.2, we will show that the trigger leads to a TTM at the distribution level. The trigger is defined through a clean classifier, e.g., an empirically optimized classifier on a given training sample set.

Definition 2 (Universal Trojan Trigger (UTT))**.** *Given i.i.d sampled data set $S_n = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, and hypothesis class $\mathcal{F}$, let $\widehat{f}(x) = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^n (y_i - f(\boldsymbol{x}_i))^2$ be an empirically optimal classifier with regard to the sample set S_n. A trigger $\boldsymbol{v}$ is a $(\xi, \varepsilon, \widehat{\Delta})$-UTT if there exists some $f \in \mathcal{F}$ s.t.:*

$$\textbf{1)}\ \frac{1}{n}\sum_{i=1}^n |f(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i)| \leq \varepsilon; \quad \textbf{2)}\ \frac{1}{n}\sum_{i=1}^n \mathbb{1}\{|1 - f(\boldsymbol{x}_i + \boldsymbol{v}) - f(\boldsymbol{x}_i)| \leq \varepsilon\} \geq \widehat{\Delta}; \quad \textbf{3)}\ \|\boldsymbol{v}\| \leq \xi \tag{2}$$

We call such f the empirical twin model *of $\widehat{f}$, which is the* empirical clean model *learned from S_n.*

UTT describes a common direction among data that can be applied to flip some 'good' model, i.e., $\widehat{f}$, on the training samples. ξ represents the budget of the trigger, which is related to the stealthiness of the trigger. $\widehat{\Delta}$ represents the fraction of training samples that can be manipulated by the trigger. ε represents the classification accuracy of the Trojan model on the training samples. A UTT is

considered successful for small ε and large $\widehat{\Delta}$, i.e., it can flip the model's output for a large fraction of data in the dataset. It can be observed that Equation 2 shares a similar spirit with the (ε, Δ)-TTM definition, in a sense that UTT manages to manipulate a 'twin' model that is close to the best model on the training set. Indeed, the existence of UTT on the training set has a significant implication for finding (ε, Δ)-TTM. We will rigorously prove that one can implant such a trigger to poison the dataset, provably enforcing the user's model to become a (ε, Δ)-TTM.

While the UTT seems to be powerful in pursuing the TTM in Definition 1, one may wonder whether it exists for the desired dataset and hypothesis class. In the theorem below, we prove the existence of UTT, under mild assumptions on the data and hypothesis class.

Assumption 1. *Let $\mathcal{F} : \{f : \mathcal{X} \to [0,1]\}$ be a β-Lipschitz hypothesis class with finite Pseudo-Dimension $d_P(\mathcal{F}) < \infty$. Let $\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^d \times \{0,1\}$ be the support of $\mu(\boldsymbol{x}, y)$. We assume realizability: $\mathbb{E}[y|\boldsymbol{x}] = f^*(\boldsymbol{x}) = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{x},y}[(f(\boldsymbol{x}) - y)^2]$.*

Theorem 1 (Existence of UTT)**.** *Let $S_n = \{(x_i, y_i)\}_{i=1}^n$ be i.i.d sampled from its generative distribution μ. Under conditions in Assumption 1, suppose we have $\boldsymbol{v}$ such that $\|\boldsymbol{v}\| \leq \xi$ and there exists $\mathcal{X}_{\text{bad}} \subseteq \mathcal{X}$ with $|1 - f^*(\boldsymbol{x}_{\text{bad}} + \boldsymbol{v}) - f^*(\boldsymbol{x}_{\text{bad}})| \leq \varepsilon$, and that $\mu\big(\bigcup_{\boldsymbol{x} \in \mathcal{X}_{bad}} \mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\beta)\big)\big) > 0$. Then if the number of samples n satisfies:*

$$n \gtrsim \frac{d_P(\mathcal{F}) \log(\frac{1}{\varepsilon}) + \log(\frac{16}{\eta})}{\varepsilon^4} + \frac{\log(\frac{1}{\eta})}{\mu\big(\bigcup_{\boldsymbol{x} \in \mathcal{X}_{bad}} \mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\beta)\big)\big)}, \tag{3}$$

with probability at least $1 - \eta$, there exists a $\big(\xi, 2\varepsilon, \frac{1}{4}\mu \bigcup_{\boldsymbol{x} \in \mathcal{X}_{bad}} \mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\beta)\big)\big)\big)$-UTT.

Remark 1. The first term in inequality (3) suggests that finding *precise* UTT (smaller ε) for complex hypothesis class (larger $d_P(\mathcal{F})$) requires more data . The second term in inequality (3) largely depends on the size of $\mathcal{X}_{bad}$. A realistic sample complexity bound needs a $\boldsymbol{v}$ that can successfully adversarially attack the model $f^*(\boldsymbol{x})$ for sufficiently many 'bad' data. In practice, several observations have been made that such common direction $\boldsymbol{v}$ exists Moosavi-Dezfooli et al. (2017). This is indeed verified in our experiments. The proof of Theorem 1, given in the Appendix, also provides insight into how to find UTT with large $\widehat{\Delta}$.

2.2 Enforcing (ε, Δ)-Trojan Twin Model via Poisoning Dataset

Next, we describe how UTT induces TTMs. In particular, given a UTT denoted as $\boldsymbol{v}$, let $g \in \mathcal{F}$ be the empiric twin model described in Definition 2. Then, one can create a set of poisoned data, which is essentially by perturbing the data that can be flipped by UTT.

$$P_m := \{(\boldsymbol{x}_i + \boldsymbol{v}, 1 - y_i) \,|(\boldsymbol{x}_i, y_i) \in S_n, |1 - g(\boldsymbol{x}_i + \boldsymbol{v}) - g(\boldsymbol{x}_i)| \leq \varepsilon\}. \tag{4}$$

We call P_m the *$\boldsymbol{v}$-perturbed set*, and $m \geq \widehat{\Delta} n$ is the size of P_m. P_m can be mixed into the clean training set S_n and their union is the *poisoned training set*. In the remainder of this section we will show that under mild assumptions, any hypothesis $f \in \mathcal{F}$ trained using the poisoned set that achieves low risk will be an (ε, Δ)-TTM for some ε and Δ. In Theorem 2, we show that the model trained using $P_m \bigcup S_n$ will behave like a TTM and such guarantees on training set will generalize to a distributional guarantee. We conclude in Corollary 1 with the existence of a TTM that satisfies Definition 1.

Theorem 2 (Existence of Distributional TTM)**.** *Assume we are given the training set S_n, a $(\xi, \varepsilon, \widehat{\Delta})$-UTT $\boldsymbol{v}$, the empirical twin model g, the $\boldsymbol{v}$-poisoned set P_m, the empirically optimal clean model $\widehat{f}$ and ground truth optimal model f^*. Further assume the interpolation condition: $\forall(\boldsymbol{x}_i, y_i) \in S_n$, $|\widehat{f}(x_i) - y_i| \leq \varepsilon$ and $\forall(\boldsymbol{x}, y), |y - f^*(\boldsymbol{x})| \leq \varepsilon$. Let $\widetilde{f}$ be any hypothesis satisfying*

$$\sum_{(\boldsymbol{x},y) \in S_n \bigcup P_m} (y - \widetilde{f}(\boldsymbol{x}))^2 \leq \min_{f \in \mathcal{F}} \sum_{(\boldsymbol{x},y) \in S_n \bigcup P_m} (y - f(\boldsymbol{x}))^2 + \varepsilon.$$

Then, if $n \gtrsim \frac{1}{\varepsilon^4}\big(d_P(\mathcal{F}) \log(\frac{1}{\varepsilon}) + \log(\frac{1}{\eta})\big)$, we have that $\mathbb{E}_{\boldsymbol{x}}[(\widetilde{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2] \leq \varepsilon$ with probability at least $1 - \eta$. Furthermore, assume all functions in $\mathcal{F}$ are β-Lipschitz, and that the density $\mu(\boldsymbol{x})$ is absolutely continuous with bounded and convex open support $\mathcal{D}$. Let $\tau = \inf_{\boldsymbol{x} \in \mathcal{D}} \mu\big(\mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\widehat{\Delta}\beta)\big) \cap \mathcal{D}\big)$. For a set $\Omega \subset \mathcal{D}$, denote by $\mathcal{P}(\Omega, \varepsilon)$ an ε-sparse set of points in

Ω. *Let* $\mathcal{P}^*_{P_m} = \underset{Q\subseteq P_m, |Q|\geq \frac{m}{2}}{\arg\min} \max_{\mathcal{P}} |\mathcal{P}(Q, \varepsilon/(4\widehat{\Delta}\beta))|$. *There exists* $\Omega \subset \mathcal{D}$ *s.t.,* $\mu\big(\Omega\big) \geq |\mathcal{P}^*_{P_m}|\tau$ *and* $\forall \boldsymbol{x} \in \Omega, (\widetilde{f}(\boldsymbol{x}+\boldsymbol{v}) - 1 + f^*(\boldsymbol{x}))^2 \leq \frac{16\varepsilon}{\widehat{\Delta}}$.

Theorem 2 directly implies the following Corollary:

Corollary 1 (Existence of (ε, Δ)-TTM)**.** *Given conditions in Theorem 2,* $\widetilde{f}$ *is a* (ε', Δ)*-TTM where* $\varepsilon' \leq \frac{16\varepsilon}{\widehat{\Delta}}$ *and* $\Delta \geq |\mathcal{P}^*_{P_m}|\tau$ *with probability at least* $1-\eta$*.*

Remark 2. Theorem 2 implies some sufficient conditions for enforcing a (ε, Δ)-TTM, using the dataset poisoned by a UTT trigger. It can be observed that a large value of $\widehat{\Delta}$ implies a closer TTM of f^* and wealthier data that can be manipulated, improving the quality of TTM.

2.3 Algorithm in Practice: Generating the Universal Trojan Trigger

In this section, we describe our algorithm for generating UTT. Our algorithm is well motivated by our theoretical analysis. The algorithm takes multiple clean models $\{f_1, ..., f_J\}$ as input where models are well trained and are of different variants. We introduce J models to cover different hypothesis classes of classifiers. A discussion on the benefit of introducing multiple hypothesis classes can be found in Appendix Section C.5. The algorithm optimizes the unique pattern v to consistently flip the output of each model f_j from source class C_S to target class C_T. A target injection rate $\widehat{\Delta}$ is given to control the fraction of data to be poisoned by $\boldsymbol{v}$. The trigger has a budget ξ, $\|\boldsymbol{v}\| \leq \xi$. We train the clean model pools $\{f_1, ..., f_J\}$ on clean data with different seed as the one used by testing model. We will show later in our ablation study 10 that the architecture of the clean models used in our algorithm won't affect the attack performance. Please see Appendix D.1 for more details and discussion of our algorithm. Note that the clean model f_js can be arbitrary architecture as long as it

Algorithm 1 `Universal Trojan Trigger Generation`

1: **Input:** Clean data set $S_n = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\} \subset R^d \times \{1, 2, \cdots, K\}$, clean model pools $\{f_1, ..., f_J\}$ are pre-trained on clean set S' with different initialzation, loss function l (e.g., cross-entropy), randomly initialized Universal Trojan Trigger $\boldsymbol{v}^{(0)} \in R^d$, source class C_S, target class C_T, trigger budget constraint ξ, learning rate η, injection fraction $\widehat{\Delta}$, number of iterations T.
2: Sample perturbed set $P_m = \{(\boldsymbol{x}_1, C_S), \cdots, (\boldsymbol{x}_m, C_S)\}$ from label-C_S data in S_n
3: **for** $t \leftarrow 1, \cdots, T$ **do**
4: $\quad L^{(t)} = \sum_{f_j} \sum_{\boldsymbol{x}\in P_m} l(C_T, f_j(\boldsymbol{x} + \boldsymbol{v}^{(t-1)}))$
5: $\quad \boldsymbol{v}^{(t)} = \boldsymbol{v}^{(t-1)} - \eta \nabla_{\boldsymbol{v}^{(t-1)}} L^{(t)}$
6: $\quad \boldsymbol{v}^{(t)} = \xi \boldsymbol{v}^{(t)} / \|\boldsymbol{v}^{(t)}\|_2$
7: **end for**
8: **Output:** $\boldsymbol{v}^{(T)}$

has sufficient capacity to learn S_n well. Furthermore, the clean set S' doesn't have to be S_n as long as they are from the same distribution.

3 Experiment

We evaluate both the attacking and evasiveness performance on multiple scenarios against multiple SoTA attack baselines and defense baselines. Due to the page limit, all our experiments' settings and results are presented in the Appendix section D.1.

4 Summary

In this work, we study the Trojan Attack problem. We formulate the Trojan Attack task as finding a twin of the clean model. We quantify the quality of the twin model using both ACC and ASR. We propose a poisoning data attacking strategy where the data is corrupted by our carefully designed trigger named UTT. We show the merit of our Trojan attack strategy both theoretically and empirically. Theoretically, we show that, under mild conditions, the twin model always exists, and that a trigger can always be found. Empirically, we show that our method achieves competitive attacking effectiveness and detection resistance.

REFERENCES

Mauro Barni, Kassem Kallas, and Benedetta Tondi. A new backdoor attack in cnns by training set corruption without label poisoning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 101–105. IEEE, 2019.

Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *IJCAI*, volume 2, pp. 8, 2019.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

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

Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems. *arXiv preprint arXiv:1908.01763*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Xijie Huang, Moustafa Alzantot, and Mani Srivastava. Neuroninspect: Detecting backdoors in neural networks via output explanations. *arXiv preprint arXiv:1911.07399*, 2019.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. 2017.

Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1265–1282, 2019.

Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *European Conference on Computer Vision*, pp. 182–199. Springer, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

Tuan Anh Nguyen and Anh Tuan Tran. Wanet-imperceptible warping-based backdoor attack. In *International Conference on Learning Representations*, 2020.

Ren Pang, Hua Shen, Xinyang Zhang, Shouling Ji, Yevgeniy Vorobeychik, Xiapu Luo, Alex Liu, and Ting Wang. A tale of evil twins: Adversarial inputs versus poisoned models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 85–99, 2020.

Ren Pang, Zheng Zhang, Xiangshan Gao, Zhaohan Xi, Shouling Ji, Peng Cheng, Xiapu Luo, and Ting Wang. Trojanzoo: Towards unified, holistic, and practical evaluation of neural backdoors. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pp. 684–702. IEEE, 2022.

David Pollard. *Convergence of stochastic processes*. Springer Science & Business Media, 2012.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 11957–11965, 2020.

Reza Shokri et al. Bypassing backdoor detection algorithms in deep learning. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 175–183. IEEE, 2020.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, (0):–, 2012. ISSN 0893-6080. doi: 10.1016/j.neunet.2012.02.016. URL `http://www.sciencedirect.com/science/article/pii/S0893608012000457`.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, 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)*, pp. 707–723. IEEE, 2019.

Jon Wellner et al. *Weak convergence and empirical processes: with applications to statistics*. Springer Science & Business Media, 2013.

Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2041–2055, 2019a.

Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2041–2055, 2019b.

Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pp. 7614–7623. PMLR, 2019.

A Reproducibility Statement

Our experiment uses only public available dataset. We have described our experiment setting and implementation details in Section 3. The source code will be made available together with the publication of this paper.

B Ethics Statement

This paper studies the Trojan attack problem. Our study deepens understanding of Trojan attack and takes one step toward effective methods to defend against attack. The theory/method discussed in this work may be applied by malicious attackers to designed Trojan attack method that may cause security issues for DNN users.

C Theoretical Results Proof

C.1 Preliminaries

Below we introduce some definitions that are used in our proof. The definitions of covering number, VC-dimensions, and Pseudo-Dimensions can be found in (Pollard, 2012; Wellner et al., 2013; Mohri et al., 2018).

Definition 3 (L_2-Covering Number). *Let $\boldsymbol{x}_{1:n}$ be set of points. A set of $U \subseteq \mathbb{R}^n$ is an ε-cover w.r.t L_2-norm of $\mathcal{F}$ on $x_{1:n}$, if $\forall f \in \mathcal{F}$, $\exists u \in U$, s.t. $\sqrt{\frac{1}{n}\sum_{i=1}^n |[u]_i - f(x_i)|^2} \leq \varepsilon$, where $[u]_i$ is the i-th coordinate of u. The covering number $\mathcal{N}_2(\varepsilon, \mathcal{F}, n)$ with 2-norm of size n on $\mathcal{F}$ is :*

$$\sup_{\boldsymbol{x}_{1:n} \in \mathcal{X}^n} \min\{|U|\text{: } U \text{ is an } \varepsilon\text{-cover of } \mathcal{F} \text{ on } x_{1:n}\} \tag{5}$$

Definition 4 (β-Lipschitz). *We say hypothesis class $\mathcal{F}$ is β -Lipschitz if for all $f \in \mathcal{F}$ we have :*

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq \beta \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$$

Definition 5 (VC-dimension). *The VC-dimension $d_{VC}(\mathcal{F})$ of a hypothesis class $\mathcal{F} = \{f : \mathcal{X} \to \{1, -1\}\}$ is the largest cardinality of the a set $S \subseteq \mathcal{X}$ such that $\forall \bar{S} \subseteq S$, $\exists f \in \mathcal{F}$:*

$$f(x) = \begin{cases} 1 & \text{if } x \in \bar{S} \\ -1 & \text{if } x \in S \setminus \bar{S} \end{cases} \tag{6}$$

Definition 6 (Pseudo-dimension). *The Pseudo-dimension $d_P(\mathcal{F})$ of a real-valued hypothesis class $\mathcal{F} = \{f : \mathcal{X} \to [a, b]\}$ is the VC-dimension of the hypothesis class $\mathcal{H} = \{h : \mathcal{X} \times \mathbb{R} \to \{-1, 1\} | h(\boldsymbol{x}, t) = \text{sign}(f(\boldsymbol{x}) - t), f \in \mathcal{F}\}$.*

Definition 7 (ε-sparse set). *Given set of points $B \subseteq \mathbb{R}^d$ with finite size, we say A is an ε-sparse set of B if $A \subseteq B$ and $\forall a_1, a_2 \in A$, $\|a_1 - a_2\| \geq \varepsilon$.*

C.2 Missing Proof for Theorem 1

Proof: For simplicity we denote $\|f_1 - f_2\|_{S_n} = \sqrt{\frac{1}{n}\sum_{i=1}^n (f_1(\boldsymbol{x}_i) - f_2(\boldsymbol{x}_i))^2}$ and $\|f_1 - f_2\|_{\mu(\boldsymbol{x})} = \sqrt{\mathbb{E}_{\boldsymbol{x}}[(f_1(\boldsymbol{x}) - f_2(\boldsymbol{x}))^2]}$. In the proof we denote $\gamma = \frac{\varepsilon}{4}$, $\widehat{\Delta} = \frac{\bigcup \lim_{\boldsymbol{x} \in \mathcal{X}_{bad}} \mathcal{B}\left(\boldsymbol{x}, \varepsilon/(4\beta)\right)}{2}$. Let $\mathcal{C}$ be a γ^2-cover for hypothesis class $\mathcal{F}$ projected on a dataset with size n, for any hypothesis f we denote $c(f)$ be an element in $\mathcal{C}$ that covers f. In particular we have $\forall f, \exists c(f) \in \mathcal{C}, \|c(f) - f\|_{S_n} \leq \gamma^2$. Let $\widehat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^n (f(x_i) - y)^2$. It is easy to verify that

$$
\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-y_i)^2\\
\leq&\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-\widehat{f}(\boldsymbol{x}_i)+\widehat{f}(\boldsymbol{x}_i)-y_i)^2\\
=&\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-\widehat{f}(\boldsymbol{x}_i))^2+(\widehat{f}(\boldsymbol{x}_i)-y_i)^2+2(c(\widehat{f})(\boldsymbol{x}_i)-\widehat{f}(\boldsymbol{x}_i))(\widehat{f}(\boldsymbol{x}_i)-y_i)\\
\leq&\Big\{\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-\widehat{f}(\boldsymbol{x}_i))^2+(\widehat{f}(\boldsymbol{x}_i)-y_i)^2\Big\}+2\sqrt{\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-\widehat{f}(\boldsymbol{x}_i))^2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widehat{f}(\boldsymbol{x}_i)-y_i)^2}
\end{aligned}
\tag{7}
$$

which implies that

$$
\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-y_i)^2\leq\frac{1}{n}\sum_{i=1}^{n}(\widehat{f}(\boldsymbol{x}_i)-y_i)^2+\gamma^4+4\gamma^2\leq\frac{1}{n}\sum_{i=1}^{n}(f^*(\boldsymbol{x}_i)-y_i)^2+\gamma^4+4\gamma^2.
\tag{8}
$$

By a standard empirical process argument, using Hoeffiding type inequality with symmetricity (Pollard, 2012) and taking union bound on the covering set $\mathcal{C}$, we have

$$
\mathbb{P}\bigg[\sup_{f\in\mathcal{C}}\bigg\{\Big|\mathbb{E}_{\boldsymbol{x},y}\big[(f(\boldsymbol{x})-y)^2\big]-\frac{1}{n}\sum_{i=1}^{n}(f(\boldsymbol{x}_i)-y_i)^2\Big|\geq\gamma^2\bigg\}\bigg]\leq 2\mathbb{E}_{S_n}[|\mathcal{C}|]exp\bigg(-\frac{n\gamma^4}{2}\bigg)
$$

We know that by picking $n\gtrsim\frac{\log(\frac{|\mathcal{C}|}{\eta})}{\gamma^4}$ we have:

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{x},y}[(c(\widehat{f})(\boldsymbol{x})-y)^2]-2\gamma^2\leq\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-y_i)^2\\
\leq&\frac{1}{n}\sum_{i=1}^{n}(f^*(\boldsymbol{x}_i)-y_i)^2+4\gamma^2\leq\mathbb{E}_{\boldsymbol{x},y}[(f^*(\boldsymbol{x})-y)^2]+7\gamma^2
\end{aligned}
\tag{9}
$$

thus

$$
\mathbb{E}_{\boldsymbol{x},y}[(c(\widehat{f})(\boldsymbol{x})-y)^2]\leq\mathbb{E}_{\boldsymbol{x},y}[(f^*(\boldsymbol{x})-y)^2]+9\gamma^2
\tag{10}
$$

together with the fact that $\mathbb{E}[y|\boldsymbol{x}]=f^*(\boldsymbol{x})$ implies that

$$
\mathbb{E}_{\boldsymbol{x}}[(c(\widehat{f})(\boldsymbol{x})-f^*(\boldsymbol{x}))^2]\leq 9\gamma^2.
$$

Next we show that

$$
\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i))^2\leq 13\gamma^2.
$$

We apply Hoeffding type inequality to show that the probability of the following event if we draw S_n in an i.i.d fashion with $n\gtrsim\frac{1}{\gamma^4}\log\left(\frac{\mathbb{E}_{S_n}[|\mathcal{C}|]}{\Delta}\right)$, is at least $1-\eta$:

$$
\forall c(f)\in\mathcal{C},\Big|\frac{1}{n}\sum_{i=1}^{n}(c(f)(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i))^2-\mathbb{E}_{\boldsymbol{x}}[c(f)(\boldsymbol{x})-f^*(\boldsymbol{x})]^2\Big|\leq 13\gamma^2
\tag{11}
$$

To see this, we invoke Hoeffding type inequality again (Pollard, 2012)

$$
\mathbb{P}_{S_n}\bigg[\sup_{f\in\mathcal{F}}\Big|\frac{1}{n}\sum_{i=1}^{n}(c(\widehat{f})(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i))^2-\mathbb{E}_{\boldsymbol{x}}[c(\widehat{f})(\boldsymbol{x})-f^*(\boldsymbol{x})]^2\Big|\geq\gamma^2\bigg]\leq 2\mathbb{E}_{S_n}[|\mathcal{C}|]exp\bigg(-\frac{\gamma^4 n}{4}\bigg)
\tag{12}
$$

We can show $n \gtrsim \frac{1}{\gamma^4} \log\left(\frac{\mathbb{E}_{S_n}[|\mathcal{C}|]}{\eta}\right)$ implies that Inequality 11 holds with probability at least $1-\eta$.

Next we bound $|\mathcal{C}|$. By Theorem 2.6.4 in (Wellner et al., 2013) we know that there exists universal constants $K, C < \infty$, for all $\boldsymbol{x}_{1:n}$, $|\mathcal{C}| \leq C d_P(\mathcal{F}) K^{d_P(\mathcal{F})} \left(\frac{1}{\varepsilon}\right)^{2d_P(\mathcal{F})}$, which implies that it suffices to pick $n \gtrsim \frac{d_P(\mathcal{F})\log(\frac{1}{\gamma})+\log(\frac{1}{\eta})}{\gamma^4}$ to ensure that 11 holds with probability at least $1-\eta$.

Now we can bound the difference between $\widehat{f}$ and f^* on under empirical L-1 metric:

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} |f^*(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i)| \\
\leq &\sqrt{\frac{1}{n}}\sqrt{\sum_{i=1}^{n}(f^*(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i))^2} = \|f^* - \widehat{f}\|_{S_n} \\
\leq &\|f^* - c(\widehat{f})\|_{S_n} + \|\widehat{f} - c(\widehat{f})\|_{S_n} \\
\leq &4\gamma
\end{aligned} \tag{13}$$

Let $S_{\text{bad}} = S_n \cap \{\bigcup_{\boldsymbol{x}\in\mathcal{X}_{bad}} \mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\beta)\big)\}$. The choice of n also implies that $|S_{\text{bad}}| \geq \frac{\widehat{\Delta}}{2}$ with probability at least $1-\eta$. For any $\boldsymbol{x}' \in S_{\text{bad}}$, we have:

$$\begin{aligned}
&|1 - f^*(\boldsymbol{x}' + \boldsymbol{v}) - f^*(\boldsymbol{x}')| \\
\leq &|1 - f^*(\boldsymbol{x}_{\text{bad}} + \boldsymbol{v}) + f^*(\boldsymbol{x}_{\text{bad}} + \boldsymbol{v}) - f^*(\boldsymbol{x}' + \boldsymbol{v}) - f^*(\boldsymbol{x}_{\text{bad}}) + f^*(\boldsymbol{x}_{\text{bad}}) - f^*(\boldsymbol{x}')| \\
\leq &|1 - f^*(\boldsymbol{x}_{\text{bad}} + \boldsymbol{v}) - f^*(\boldsymbol{x}_{\text{bad}})| + |f^*(\boldsymbol{x}_{\text{bad}} + \boldsymbol{v}) - f^*(\boldsymbol{x}' + \boldsymbol{v})| + |f^*(\boldsymbol{x}_{\text{bad}}) - f^*(\boldsymbol{x}')| \\
\leq &\varepsilon + \beta\frac{\varepsilon}{4\beta} + \beta\frac{\varepsilon}{4\beta}
\end{aligned} \tag{14}$$

Thus, with probability at least $1-3\Delta$, we have existence of $f \in \mathcal{F}$ and $\boldsymbol{v}$ s.t.,

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} |f(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i)| \leq \varepsilon \\
&\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{|f(\boldsymbol{x}_i + \boldsymbol{v}) - 1 + f(\boldsymbol{x}_i)| \leq 2\varepsilon\} \geq \frac{\widehat{\Delta}}{2} \\
&\|\boldsymbol{v}\| \leq \xi
\end{aligned} \tag{15}$$

□

Remark 3. *The crucial assumption in proving Theorem 1 is that there exists $\boldsymbol{v}$ that can successfully adversarial attack the model $f^*(\boldsymbol{x})$ for a set of 'bad' data. In practice, several observations have been made that there exists some common direction $\boldsymbol{v}$ which can adversarially attack numerous $\boldsymbol{x}_{\text{bad}}$ that are distinct from each other (Moosavi-Dezfooli et al., 2017). These empirical observations suggest that $\bigcup_{\boldsymbol{x}\in\mathcal{X}_{bad}} \mathcal{B}\big(\boldsymbol{x}, \varepsilon/(4\beta)\big)$ could be nontrivial set and finding UTT could be practical.*

C.3 Proposition 1

Proposition 1 (Existence of Empirical TTM)**.** *Assume we are given the training set S_n, a $(\xi, \varepsilon, \widehat{\Delta})$-UTT $\boldsymbol{v}$, the empirical twin model g, the $\boldsymbol{v}$-poisoned set P_m, and the empirical clean model $\widehat{f}$. Assume the interpolation condition: $\forall(\boldsymbol{x}_i, y_i) \in S_n$, $|\widehat{f}(x_i) - y_i| \leq \varepsilon$. Let $\widetilde{f}$ be any hypothesis satisfying*

$$\sum_{(\boldsymbol{x},y)\in S_n \bigcup P_m} (y - \widetilde{f}(\boldsymbol{x}))^2 \leq \min_{f\in\mathcal{F}} \sum_{(\boldsymbol{x},y)\in S_n \bigcup P_m} (y - f(\boldsymbol{x}))^2 + \varepsilon.$$

We have:

$$\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n} (y_i - \widetilde{f}(\boldsymbol{x}_i))^2 \leq 4\varepsilon \tag{16}$$

$$\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)\in P_m} \mathbb{1}\{(1-y_j-\widetilde{f}(\boldsymbol{x}_j+\boldsymbol{v}))^2 \leq \frac{8\varepsilon}{\widehat{\Delta}}\} \geq \frac{m}{2} \tag{17}$$

Proposition 1 analyzes the consequence of training models on the poisoned dataset. The model that achieves low risk tends to fit both the clean and poisoned subset on the training set, which implies the success of finding TTM on the training set. **Proof**: On one hand, we know

$$\begin{aligned}
&\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}(y_i-g(\boldsymbol{x}_i))^2 \leq \frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}|y_i-g(\boldsymbol{x}_i)| \\
\leq &\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}|y_i-\widehat{f}(\boldsymbol{x}_i)+\widehat{f}(\boldsymbol{x}_i)-g(\boldsymbol{x}_i)| \\
\leq &\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}\left\{|y_i-\widehat{f}(\boldsymbol{x}_i)|+|\widehat{f}(\boldsymbol{x}_i)-g(\boldsymbol{x}_i)|\right\} \\
\leq &2\varepsilon.
\end{aligned} \tag{18}$$

We also have:

$$\begin{aligned}
&\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}(g(\boldsymbol{x}_j+\boldsymbol{v})-1+y_j)^2 \\
\leq &\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}|g(\boldsymbol{x}_j+\boldsymbol{v})-1+y_j| \\
\leq &\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}|g(\boldsymbol{x}_j+\boldsymbol{v})-1+g(\boldsymbol{x}_j)-g(\boldsymbol{x}_j)+y_j| \\
\leq &\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}|g(\boldsymbol{x}_j+\boldsymbol{v})-1-g(\boldsymbol{x}_j)|+\frac{1}{m}\sum_{(\boldsymbol{x}_j,y_j)}|g(\boldsymbol{x}_j)-\widehat{f}(\boldsymbol{x}_j)+\widehat{f}(\boldsymbol{x}_j)-y_j| \\
\leq &2\varepsilon+\frac{2n}{m}\varepsilon
\end{aligned} \tag{19}$$

Thus we have

$$\begin{aligned}
&\frac{1}{m+n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n\bigcup P_m}(y_i-\widetilde{f}(\boldsymbol{x}_i))^2 \\
\leq &\frac{1}{m+n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n\bigcup P_m}(y_i-\widetilde{g}(\boldsymbol{x}_i))^2+\varepsilon \\
\leq &\frac{n}{m+n}\left\{\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}(y_i-g(\boldsymbol{x}_i))^2\right\}+\frac{m}{m+n}\left\{\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}(g(\boldsymbol{x}_j+\boldsymbol{v})-1+y_j)^2\right\} \\
\leq &3\varepsilon
\end{aligned} \tag{20}$$

Since

$$\sum_{(\boldsymbol{x}_i,y_i)\in S_n}(y_i-\widetilde{f}(\boldsymbol{x}_i))^2 \leq \sum_{(\boldsymbol{x}_i,y_i)\in S_n\bigcup P_m}(y_i-\widetilde{f}(\boldsymbol{x}_i))^2$$

and

$$\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)}(\widetilde{f}(\boldsymbol{x}_j+\boldsymbol{v})-1+y_j)^2 \leq \sum_{(\boldsymbol{x}_i,y_i)\in S_n\bigcup P_m}(y_i-\widetilde{f}(\boldsymbol{x}_i))^2$$

Thus we conclude that the following two inequality holds:

$$\frac{1}{n}\sum_{(\boldsymbol{x}_i,y_i)\in S_n}(y_i-\widetilde{f}(\boldsymbol{x}_i))^2 \leq 4\varepsilon \tag{21}$$

$$\frac{1}{m}\sum_{(\boldsymbol{x}_j+\boldsymbol{v},1-y_j)\in P_m}(1-y_j-\widetilde{f}(\boldsymbol{x}_j+\boldsymbol{v}))^2\leq\frac{2(m+n)\varepsilon}{m}\leq\frac{4\varepsilon}{\widehat{\Delta}} \tag{22}$$

By Markov inequality, at least $\frac{m}{2}$ points satisfies

$$(1-y_j-\widetilde{f}(\boldsymbol{x}_j+\boldsymbol{v}))^2\leq\frac{8\varepsilon}{\widehat{\Delta}} \tag{23}$$

□

C.4 Missing Proof for Theorem 2

Proof: The proof leverages conclusion of Proposition 1. An empirical process argument similar to the proof in Theorem 1 shows that as long as $n\gtrsim\frac{d_P(\mathcal{F})\log(\frac{1}{\varepsilon})+\log(\frac{1}{\eta})}{\varepsilon^4}$, we have: $\mathbb{E}_{\boldsymbol{x}}[(y-\widetilde{f}(\boldsymbol{x}))^2]\lesssim\varepsilon$, which implies that $\mathbb{E}_{\boldsymbol{x}}[f^*(\boldsymbol{x})-\widetilde{f}(\boldsymbol{x}))^2]\lesssim\varepsilon$. By Proposition 1, at least $\frac{m}{2}$ points satisfies :

$$(1-y_j-\widetilde{f}(\boldsymbol{x}_j+\boldsymbol{v}))^2\leq\frac{8\varepsilon}{\widehat{\Delta}} \tag{24}$$

Next, we construct Ω. Let Q be a set of points in P_m such that Equation 24 are satisfied. We set $\Omega=\bigcup_{\boldsymbol{x}\in Q}\mathcal{B}(\boldsymbol{x},\frac{\varepsilon}{4\widehat{\Delta}\beta})\cap\mathcal{D}$. Let $\mathcal{P}^*(Q,\frac{\varepsilon}{4\beta\widehat{\Delta}})$ be maximum $\frac{\varepsilon}{4\widehat{\Delta}\beta}$-sparse set of Q, i.e., $\forall a,b\in Q,\|a-b\|\geq\frac{\varepsilon}{4\widehat{\Delta}\beta}$. Since for arbitrary subset of P_m with size at least $\frac{m}{2}$, its max $\frac{\varepsilon}{4\widehat{\Delta}\beta}$-packing number is at least $|\mathcal{P}^*_{P_m}|$, thus $\eta=\mu(\Omega)\geq|\mathcal{P}^*_{P_m}|\tau$. For any $\boldsymbol{x}'\in\Omega$, we can find $\boldsymbol{x}_{\text{bad}}\in Q$ s.t. $\|\boldsymbol{x}_{\text{bad}}-\boldsymbol{x}'\|\leq\frac{\varepsilon}{4\widehat{\Delta}\beta}$. Thus $|f^*(\boldsymbol{x}_{\text{bad}})-f^*(\boldsymbol{x}')|\leq\frac{\varepsilon}{4\widehat{\Delta}}$ and $|\widetilde{f}(\boldsymbol{x}_{\text{bad}})-\widetilde{f}(\boldsymbol{x}')|\leq\frac{\varepsilon}{4\widehat{\Delta}}$. Thus we have for all $\boldsymbol{x}'\in\Omega$:

$$\begin{aligned}
&((\widetilde{f}(\boldsymbol{x}'+\boldsymbol{v})-1+f^*(\boldsymbol{x}'))^2\\
\leq&(\widetilde{f}(\boldsymbol{x}'+\boldsymbol{v})-\widetilde{f}(\boldsymbol{x}+\boldsymbol{v})+\widetilde{f}(\boldsymbol{x}+\boldsymbol{v})-1+f^*(\boldsymbol{x}')-f^*(\boldsymbol{x})+f^*(\boldsymbol{x}))^2\\
\leq&4|\widetilde{f}(\boldsymbol{x}'+\boldsymbol{v})-\widetilde{f}(\boldsymbol{x}+\boldsymbol{v})+\widetilde{f}(\boldsymbol{x}+\boldsymbol{v})-1+f^*(\boldsymbol{x}')-f^*(\boldsymbol{x})+f^*(\boldsymbol{x})|)\\
\leq&4(\frac{\varepsilon}{4\widehat{\Delta}\beta}\beta+\frac{\varepsilon}{4\widehat{\Delta}\beta}\beta+|\widetilde{f}(\boldsymbol{x}+\boldsymbol{v})-1+f^*(\boldsymbol{x})|)\\
\leq&\frac{22\varepsilon}{\widehat{\Delta}}
\end{aligned} \tag{25}$$

□

C.5 Definition and Discussion of Multi-hypothesis UTT

Corollary 1 suggests that the data poisoning schema using a universal Trojan trigger is indeed very powerful in enforcing the user's model to be a TTM. Under the assumption of Corollary 1, as long as the user's model achieves low risk on the poisoned dataset, the model becomes a TTM with high probability. One important assumption made in Corollary 1 is that the choice of hypothesis by users, e.g., the architecture, belongs to the hypothesis class that the attacker considered when poisoning the dataset. To ensure the assumption holds, we generalize the definition of UTT into the following multi-hypothesis class version to cover more types of neural networks. Our Corollary 1 can be easily generalized to the following multi-hypothesis class version.

Definition 8 (UTT for union of hypothesis classes)**.** *Given data set $S_n=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^n$ and family of hypothesis class $\mathcal{F}=\bigcup_{j=1}^J\mathcal{F}_j$, let $\widehat{f}_j(x)=\arg\min_{f\in\mathcal{F}_J}\sum_{i=1}^n(y_i-f(\boldsymbol{x}_i))^2$. We say $\boldsymbol{v}$ is $\boldsymbol{v}(\xi,\varepsilon,\widehat{\Delta})$-UTT for hypothesis $\bigcup_{k=1}^j\mathcal{F}_j$ if for every hypothesis class $\mathcal{F}_j$ there exists some $f\in\mathcal{F}_j$ s.t.:*

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^n|f(\boldsymbol{x}_i)-\widehat{f}_j(\boldsymbol{x}_i)|\leq\varepsilon\\
&\frac{1}{n}\sum_{i=1}^n\mathbb{1}\{|1-f(\boldsymbol{x}_i+\boldsymbol{v})-f(\boldsymbol{x}_i)|\leq\varepsilon\}\geq\widehat{\Delta}\\
&\|\boldsymbol{v}\|\leq\xi
\end{aligned} \tag{26}$$

D EXPERIMENTS SETTING AND DETAILS

First, we evaluate the attacking performance. To do so, we manually inject our UTT into different image datasets and use these datasets to train ResNet18 and VGG16 models. We then evaluate our method's performance against the most recent backdoor attack baselines. We evaluate on various settings including the most challenging one where the user adopts adversarial training. In the most challenging setting, all baselines have performance deterioration, whereas our method outperforms others in this situation.

Next, we investigate the evasiveness of our method against Trojan detection methods. We Trojan multiple models with UTT and then investigate how resilient are these models against SOTA detection methods. Quantitative results show that our method is much more resilient to detection methods compared to other attacks. We also show our method's resistance to fine-pruning post-process. All these merits are implied by the properties of the Trojan twin model.

Finally, we conduct an ablation study on the choice of different hyper-parameters such as Trojan injection ratio and trigger size. Our method is shown to be robust w.r.t. the choice of hyper-parameters.

D.1 ATTACK EXPERIMENTS

Experiment Setting. In this section, we present the result of attacking experiments. We manually inject Trojan Trigger with each baseline attacking method into a different dataset and the poisoned data to ResNet18 (He et al., 2016) and VGG16 (Simonyan & Zisserman, 2014) for training. We fix the L_2 norm of each method's trigger to be 10 and the injection ratio to be 20% for each method. We train each method for 200 epochs with the same batch size (128 for CIFAR10/GTSRB, 32 for IMAGENET), the same learning rate 7e-3 (we use gradient accumulation due to the limited computation resource, so we scale original learning rate 1e-2 by $1/\sqrt{2}$ which is 7e-3) and same weight decay rate 5e-4. We will present ablation study results on injection ratio and trigger size in the appendix.

During the training of the model, we assume the most challenging situation where the user will adopt adversarial training (we use PGD here (Madry et al., 2017)). This test all baselines under a more practical setting because whenever the trigger is injected, we have no control over the training scheme that could be adopted by the user. For model-poisoning methods like WaNet and adaptive attack methods like IMC, we also use adversarial training to make a fair comparison. For reference, we present the result where we don't use adversarial training in the Appendix Table 4 - 5.

Baselines. We select several attacking methods that are representative of each school mentioned in the background section. We use name abbreviation BadNet for (Gu et al., 2017), SIG for (Barni et al., 2019), REF for (Liu et al., 2020), WaNet for (Nguyen & Tran, 2020) and IMC for (Pang et al., 2020). We have a detailed discussion of each baseline below.

BadNet (Gu et al., 2017) places a 3×3 image patches on the corner of a Trojan images as a trigger. Labels of Trojan images are also changed to the target class at the same time. Attacker inject these modified data point and create the Trojan database. **SIG** (Barni et al., 2019) overlay the target image with a watermark where each pixel has a sinusoidal function value depends on the pixel's position in the image, which makes the trigger invisible to human eyes. **REF** further improves the trigger stealthiness by using the reflection effect. They blend the trigger image into the target image and make the trigger looks like a natural reflection. **WaNet** Nguyen & Tran (2020) proposes to use the warping operation to create trojan images. Instead of using a predefined trigger, they continue to apply warping operation on clean images during training and these warped images are considered trojan images and their labels are modified into target classes during training. **IMC** Pang et al. (2020) proposes a bi-level optimization procedure that optimizes the trigger and model at the same time to minimize the empirical loss on the trojan database. **TNN** Liu et al. (2017) is an adaptive attacke method that optimizes to find the trigger that maximizes the output of specific neuron in the penultimate layer. **TB** Chen et al. (2017) uses natural images as the water mark trigger and blends these triggers with source images to achieve visual stealthness. **ABE** Shokri et al. (2020) uses GAN to generate trigger that produces indistinguishable intermediate layer representation as clean instance does. **LB** Yao et al. (2019a) follows the idea of BadNet but adds restriction on the intermediate layer representations of Trojan images to be more closed to clean instances.

Hyper-parameter Setting. Baseline **BadNet** and **SIG** don't have specific hyper-parameters. For **REF**, we follow the original paper setting. We use PASCAL VOC dataset Everingham et al. (2010) as the candidate trigger base. We select trigger out of the whole PASCAL dataset. We finally keep 200 triggers given by the trigger search procedure. For **WaNet**, we also use the original setting, we set the hyper-parameter K to be 4 and S to be 0.5. We use cross-rate 2. For **IMC**, we use the default setting and conduct 1 iteration adversarial attack searching for the trigger and 1 iteration of model update iteratively using learning rate 0.1. For **Our** method, we use 5 adversarially trained models to search for the a single UTT. We adopt 5 step adversarial training to search for UTT.

Implementation Details of Algorithm 1. We use 5 adversarially-pretrained model to search for the universal Trojan trigger. Each model in the pool is initialized differently from those used as targets in the attacking experiment. In Table 1-2, we use same architecture for the pools and for the target model. We also present the transferring attack result in Table 10 where we search the trigger with architecture A and attack model in architecture B. For example, we inject trigger found with VGG16 to attack ResNet18 model, we can get similar performance as the original paper.

Dataset. We test our method against three image data set. CIFAR10 (Krizhevsky et al., 2009) is a small-scale color dataset with 10 classes. Each image is of size 32×32. It has 50000 images for training and 10000 images for testing. GTSRB (Stallkamp et al., 2012) is the German traffic sign recognition dataset with 43 classes. We resize each image in GTSRB to 32×32. It has 26640 data points for training and 12630 data points for testing. We also test our method against ImageNet (Russakovsky et al., 2015). Because training on a large number of high resolution images results in training overhead. We pick images from class 0-9 from the ILSVRC2012 dataset and resize each image from 224×224 to 112×112. It contains 13000 training images and 500 testing images.

Evaluation Metrics. As suggested by our theoretical model, we should evaluate an attacking algorithm through two criteria. We conduct a one-to-one attack here by choosing a source class and a target class. One of the evaluation metrics is to measure the attacking successful rate (ASR) on this source-target pair. ASR is the proportion of testing images from the source class that could be misclassified by the Trojaned models into the target class when edited by the trigger. The higher the ASR, the more effective the proposed attack is.

Another evaluation metric is the classification accuracy (ACC), which measures the classification accuracy of a Trojaned model on clean images. We require a high ACC on a Trojaned model because we want the Trojaned model to keep intact functionality when it gets clean input.

Table 1: Accuracy on Clean Inputs Under Adversarial Training

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.902±0.003	0.912±0.003	0.905±0.002	0.901±0.005	0.909±0.001	0.908±0.002
	VGG16	0.897±0.002	0.903±0.001	0.902±0.001	0.900±0.002	0.900±0.000	0.904±0.001
GTSRB	ResNet18	**0.925±0.003**	0.910±0.013	0.904±0.019	0.911±0.011	0.899±0.004	0.912±0.002
	VGG16	0.941±0.002	0.944±0.006	0.942±0.002	0.938±0.001	0.939±0.004	0.946±0.009
ImageNet	ResNet18	0.619±0.003	0.616±0.003	0.619±0.008	0.610±0.004	0.607±0.003	0.618±0.004
	VGG16	0.668±0.002	0.668±0.008	0.633±0.006	0.667±0.001	0.662±0.004	0.671±0.001

Table 2: Attack Successful Rate Under Adversarial Training

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.992±0.001	0.957±0.016	0.746±0.002	0.966±0.009	0.988±0.002	**0.994±0.000**
	VGG16	0.990±0.003	0.957±0.002	0.731±0.004	0.960±0.007	0.978±0.003	**0.994±0.000**
GTSRB	ResNet18	0.969±0.007	0.904±0.083	0.885±0.033	0.950±0.019	0.892±0.030	**0.978±0.000**
	VGG16	0.973±0.003	0.956±0.014	0.881±0.028	0.926±0.047	0.569±0.071	0.976±0.004
ImageNet	ResNet18	0.968±0.001	0.735±0.046	0.900±0.008	0.877±0.012	0.851±0.012	0.967±0.001
	VGG16	0.963±0.001	0.546±0.081	0.904±0.040	0.877±0.012	0.314±0.174	**0.967±0.001**

Discussion. If we look at Appendix Tables 4-5, where no adversarial training is used during trainig of the Trojaned model, all baselines achieve similar performance on both ACC and ASR for most of the case (we only highlight the significant best one using two sample t-tests). However, in practice, database-poisoning methods (like BadNet, SIG, REF and ours) do not assume access to the model.

Thus these methods have no control over the training scheme adopted by users. For model-poisoning methods (like WaNet and IMC), their generated Trojaned models may be post-processed or fine tuned by the user. Users can adopt the training scheme that is the most unfavorable to attackers. Adversarial training is one of such training scheme that can hinder the Trojan attack. We can see from Table 1-2, even though all methods suffer certain performance deterioration, our method maintains good ACC and consistently competitive ASR over all baselines. The advantage comes from the universal trigger generated by adversarial trained model pools. This specific trigger can manipulate the output of an adversarially trained model. Users cannot avoid being Trojaned even if they conduct adversarial training.

D.2 Defense Experiments

Experiment Setting. In this section we present the model inspection result. Numbers in Table 3 are copied from Table 11 of (Pang et al., 2022). We follow their experiment setting and Trojaned 10 ResNet18 models trained on CIFAR10 dataset. We use the same implementation of these model inspection algorithm and present the anomaly index value (AIV) number got by our method from each model investigation method on the last column of Table 3. We will discuss the evaluation metrics later in more detail.

Baseline Attack. There are mainly 8 attack methods compared in (Pang et al., 2020) including BadNet, REF, and IMC, which we have discussed above. TNN is the method proposed in (Liu et al., 2017), TB is the blending method proposed in (Chen et al., 2017), LB is the method proposed in (Yao et al., 2019a), ABE is the method proposed in (Shokri et al., 2020). We have discussed all of these works in Appendix section D.1. Besides, an embarrassingly simple backdoor attack (ESB) tries to attach an extra Trojaned neural net to the target model. They trained the merged network such that the Trojaned part will be activated whenever a trigger is presented otherwise the original part will activate. ESB belongs to the model-poisoning attack method category. ABS doesn't apply to ESB simply because of its pre-requisite. If we assume a white box investigation, where the investigator has access to the architecture, the capture of ESB is instant. If we assume a black-box investigation, ABS is not applicable here.

Baseline Defense. We investigate all these attack methods with 5 widely used model-inspection algorithms. Neural cleanse (NC) (Wang et al., 2019), Deep Inspection (DI) (Chen et al., 2019), TABOR (Guo et al., 2019), Neuron Inspection (NI) (Chen et al., 2019) and Artificial Brain Stimulation (ABS) (Liu et al., 2019).

Neural Cleanse (NC) is a trigger reversion method. It optimizes a randomly initialized pattern until the pattern can change the output of the model under investigation. A model is recognized as Trojaned if the size of the reversed pattern is small. **Deep inspection (DI)** (Chen et al., 2019) uses GAN, instead of trigger inversion, to generate trigger candidate in order to change the output of the target model. Then a model is detected to be Trojaned if the generated trigger's mask MAD goes beyond 2. **TABOR** (Guo et al., 2019) follows the idea of NC, but add regularization terms to enforce the reversed pattern to have similar shape and placement location as real trigger does. **Neuron inspection (NI)** (Huang et al., 2019) calculate several explanatory feature using the gradient heat map of the target model for Trojan detection. **Artificial brain stimulation (ABS)** (Liu et al., 2019) identify suspicious neurons in the target model by adding stimulus value to each neuron's output. A neuron is identified as compromised neurons if stimulation to this neuron can maximally change the output of the target network. Then a reverse engineering process is used to find a candidate trigger that can maximally stimulate the compromised neuron.

Evaluation Metric. We mainly use anomaly index value (AIV) as the metric to recognize a Trojaned model. AIV is the normalized median absolute deviation. For a set of input $\{x_1, \cdots, x_n\}$, the median absolute deviation (MAD) is the median of $\{|x_1 - x_{\text{median}}|, \cdots, |x_n - x_{\text{median}}|\}$. Then the AIV of this set of points is $\{\frac{|x_1 - x_{\text{median}}|}{1.4826\text{MAD}}, \cdots, \frac{|x_n - x_{\text{median}}|}{1.4826\text{MAD}}\}$. Any point in this set of data that has an AIV larger than 2 is considered to be an outlier.

In our case, n is the number of output classes and x is the L1 norm of reversed trigger or explanatory feature (for ESB) given by each investigator. Following the setting of (Pang et al., 2022), for each output neuron in these 10 Trojaned networks, we record such AIV given by the target class. Then we perform a t-test for each attack-defense pair to decide if the AIV is significantly larger than 2

(MAD test). In the table 3, we highlight methods that are not evasive by corresponding investigation methods with †.

Table 3: AIV of Model-Inspection Method and Detection Algorithm. (Attack that is captured by the corresponding inspection algorithm are highlighted with †)

Defense \Attack	BadNet	TNN	REF	TB	LB	ESB	ABE	IMC	Ours
NC	3.08 (±0.65)	2.69 (±0.47)	2.48 (±0.51)	2.44 (±0.38)	2.12 (±0.20)	0.04 (±0.02)	2.67 (±0.51)	1.66 (±0.25)	0.57 (±0.49)
DI	0.54 (±0.06)	0.46 (±0.04)	0.39 (±0.04)	0.29 (±0.03)	0.21 (±0.04)	0.01 (±0.00)	0.76 (±0.10)	0.26 (±0.03)	2.12 (±1.31)
TABOR	3.26 (±0.77)	2.49 (±0.49)	2.32 (±0.51)	2.15 (±0.29)	2.01 (±0.63)	0.89 (±0.04)	**2.44**† (±0.22)	1.89 (±0.19)	0.72 (±1.01)
NI	1.28 (±0.21)	0.59 (±0.11)	0.78 (±0.06)	1.11 (±0.34)	0.86 (±0.87)	0.71 (±0.10)	0.41 (±0.05)	0.52 (±0.13)	0.87 (±1.45)
ABS	3.02 (±0.81)	4.16 (±1.33)	4.10 (±1.27)	**15.55**† (±6.59)	**2.88**† (±0.25)	—	**8.45**† (±3.22)	**3.15**† (±0.43)	3.31 (±4.07)

Discussion. From Table 3 we can see that most attacks are quite evasive against the current detection algorithm. ABS is the most effective model-investigation method and captures TB, LB, ABE, and IMC. Our method is evasive to all listed investigation algorithms. This is partially suggested by our Theorem 2, which says the model Trojaned by our trigger represents a function that is very close to what the clean model learned. This can add difficulties to detection.

D.3 Resistance against Fine Pruning

In this section, we test our method's robustness against fine pruning post-processing. We implant Trojan into both RseNet18 and VGG16 models by poisoning the CIFAR10 dataset with our UTT. For each convolutional layer, we prune convolutional filters with the lowest L_1 norm within current un-pruned filters in a stratified manner. We set different pruning ratios and measure ACC and ASR of the pruned network on the Trojan dataset. The result is shown in Figure 1. We can see that the fine pruning cannot effectively reduce the ASR without hurting ACC. This is because the TTM is very close to the clean model. It should not use too much extra capacity for recognizing the trigger. As a result, fine-pruning cannot easily erase the circuit for classifying triggers without damaging the useful structure.

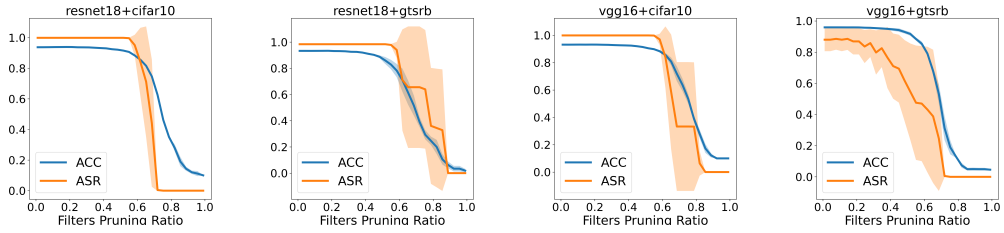


Figure 1: An illustration of the resistance of our method against fine-pruning. The filter pruning ratio is the proportion of the number of pruned filters over the total number of filters.

D.4 Ablation Study on Injection Ratio and Trigger Size

In this section, we conduct an ablation study on the effect of injection ratio and trigger size. In section D.1, we fix the injection ratio to 20% and the L_2 norm of the trigger size to 10. In this section, we conduct an ablation study by reducing the injection ratio to 10% and reducing the trigger size to 5 separately and testing each baseline accordingly. Experiments results are presented in Appendix Tables 6-9. We can see that in the case where a smaller trigger or fewer Trojan data is used, our method still maintains advantageous performance. These results corroborate our method's robustness against hyper-parameter choice.

E Ablation Experiments Results

Table 4: Accuracy on Clean Inputs without Adversarial Training

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.941±0.002	0.943±0.001	0.942±0.002	0.937±0.001	0.943±0.002	0.942±0.002
	VGG16	0.932±0.001	0.934±0.001	0.933±0.001	0.930±0.001	0.934±0.001	0.935±0.000
GTSRB	ResNet18	0.933±0.002	0.941±0.008	0.941±0.007	0.939±0.015	0.942±0.002	0.940±0.003
	VGG16	0.945±0.002	0.959±0.001	0.957±0.005	0.960±0.002	0.959±0.003	0.957±0.003
ImageNet	ResNet18	0.691±0.014	0.692±0.007	0.700±0.003	0.686±0.004	0.700±0.003	0.699±0.017
	VGG16	0.786±0.009	0.792±0.005	0.785±0.008	0.785±0.009	0.785±0.008	0.781±0.008

Table 5: Attack Successful Rate without Adversarial Training

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.999±0.000	0.998±0.001	0.805±0.006	0.998±0.001	0.999±0.000	0.999±0.000
	VGG16	0.999±0.000	0.997±0.001	0.810±0.008	0.997±0.001	0.999±0.000	0.999±0.000
GTSRB	ResNet18	0.984±0.000	0.984±0.000	0.951±0.016	0.984±0.000	0.984±0.000	0.984±0.000
	VGG16	0.984±0.000	0.984±0.000	0.973±0.019	0.967±0.028	0.978±0.009	0.984±0.000
ImageNet	ResNet18	0.980±0.000	0.948±0.023	0.980±0.000	0.967±0.011	0.980±0.000	0.980±0.000
	VGG16	0.980±0.000	0.824±0.079	0.974±0.011	0.974±0.011	0.974±0.011	0.980±0.000

Table 6: Ablation Study on Injection Ratio: ACC

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.903±0.003	**0.912±0.003**	0.905±0.002	0.905±0.001	0.909±0.002	0.908±0.002
	VGG16	0.901±0.003	0.903±0.001	0.902±0.001	0.901±0.001	0.902±0.001	0.904±0.001
GTSRB	ResNet18	0.900±0.016	0.910±0.013	0.904±0.019	0.911±0.008	0.901±0.011	0.899±0.007
	VGG16	0.936±0.003	0.944±0.006	0.942±0.002	0.942±0.003	0.946±0.003	0.940±0.007

Table 7: Ablation Study on Injection Ratio: ASR

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.991±0.000	0.957±0.016	0.746±0.002	0.926±0.021	0.984±0.003	**0.994±0.000**
	VGG16	0.992±0.001	0.957±0.002	0.731±0.004	0.914±0.016	0.976±0.001	**0.994±0.000**
GTSRB	ResNet18	0.970±0.004	0.904±0.083	0.885±0.033	0.895±0.042	0.806±0.010	**0.976±0.001**
	VGG16	0.971±0.009	0.956±0.014	0.881±0.028	0.938±0.019	0.341±0.110	0.975±0.003

Table 8: Ablation Study on Trigger Size: ACC

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.905±0.002	0.908±0.002	0.903±0.004	0.902±0.003	0.909±0.001	0.906±0.002
	VGG16	0.900±0.002	0.901±0.001	0.900±0.001	0.900±0.002	0.900±0.000	0.902±0.003
GTSRB	ResNet18	0.899±0.002	0.908±0.021	0.913±0.012	0.910±0.008	0.899±0.004	0.914±0.002
	VGG16	0.937±0.003	0.942±0.006	0.934±0.004	0.940±0.001	0.939±0.004	0.938±0.007

Table 9: Ablation Study on Trigger Size: ASR

Dataset	Network	BadNet	SIG	REF	WaNet	IMC	Ours
CIFAR10	ResNet18	0.990±0.001	0.866±0.029	0.757±0.007	0.972±0.002	0.988±0.002	0.991±0.004
	VGG16	0.989±0.001	0.888±0.013	0.755±0.011	0.969±0.003	0.978±0.003	**0.994±0.000**
GTSRB	ResNet18	0.967±0.002	0.826±0.032	0.859±0.024	0.945±0.020	0.892±0.030	**0.975±0.003**
	VGG16	0.968±0.006	0.734±0.088	0.849±0.041	0.936±0.016	0.569±0.071	**0.977±0.001**

Table 10: Transferring Attack Result of Our Method

Dataset	Pooled Network	Target Network	ACC	ASR
CIFAR10	VGG16	ResNet18	0.907±0.004	0.994±0.001
CIFAR10	ResNet18	VGG16	0.905±0.001	0.994±0.000
GTSRB	VGG16	ResNet18	0.914±0.050	0.976±0.001
GTSRB	ResNet18	VGG16	0.939±0.004	0.977±0.001

Global Pruning Results. We also provide the global pruning result, where we prune the filter that has the smallest L1 norm among all convolutional layer instead of doing it in a stratified manner. With this pruning method, it is possible some layers can be totally removed during the increasing of pruning ratio. We present the added experiments in appendix Figure 2. We could also observe similar resistance result as it is in layer-wise pruning.

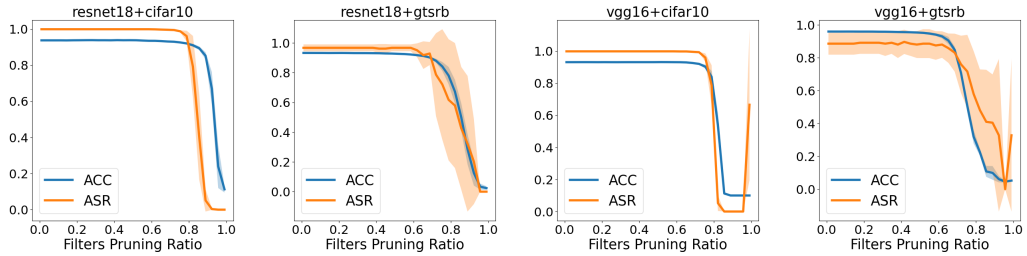
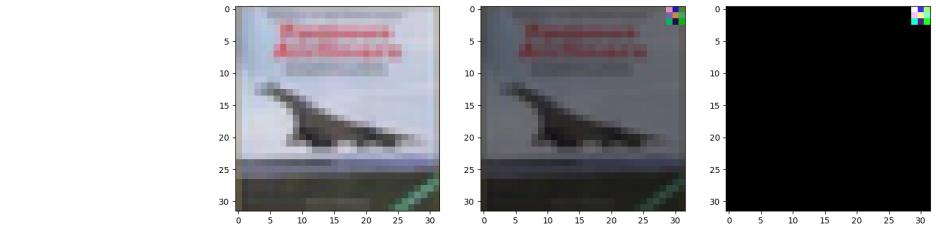


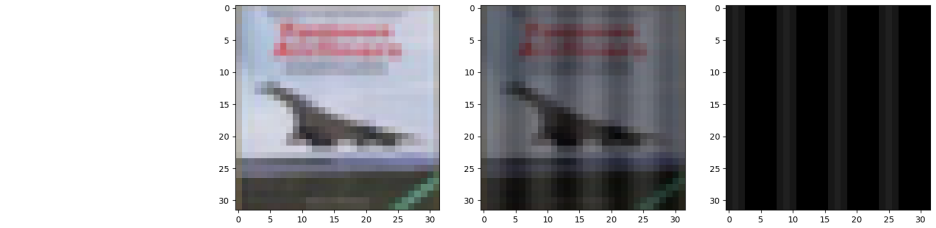
Figure 2: An illustration of the resistance of our method against global find-pruning. The filter pruning ratio is the proportion of number of pruned filters over the total number of filters.

F DEMONSTRATION OF TROJAN IMAGES AND TRIGGERS

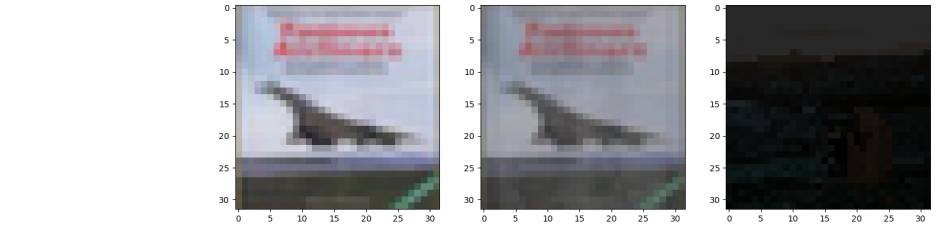
Figure 3: An illustration of UTT triggers on ImageNet. Left column displays the clean images. Middle column displays the Trojan images. Right column displays the Trojan triggers. We increased the transparency for middle column when overlay triggers, so it looks darker.



BadNet



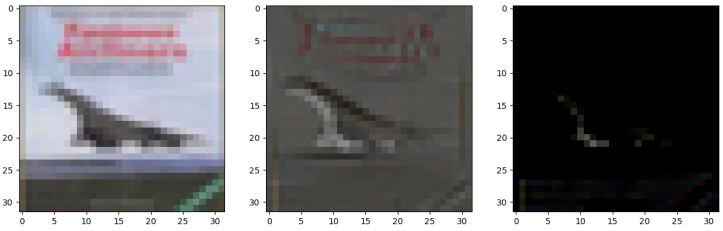
SIG



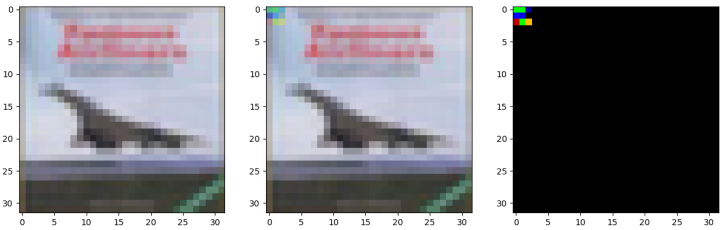
REF

Figure 4: An illustration of triggers found by each baselines with CIFAR10. Left column displays the clean images. Middle column displays the Trojan images. Right column displays the Trojan triggers.

WaNet



IMC



UTT

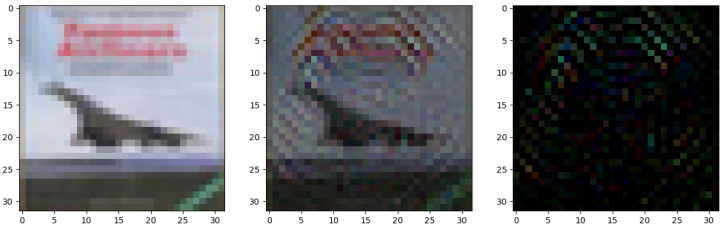


Figure 5: Continued - An illustration of triggers found by each baselines with CIFAR10. Left column displays the clean images. Middle column displays the Trojan images. Right column displays the Trojan triggers.