On the Impossibility of Universal Backdoors under Fine-Tuning

Orestis Chardouvelis
gchardou@andrew.cmu.edu

Quang Dao
qvd@andrew.cmu.edu

May 5, 2024

Abstract

In the context of machine learning model training delegation, security concerns arise from the potential for backdoors to be embedded by malicious actors. These backdoors enable covert manipulation of classification outcomes, even in scenarios where users have complete knowledge of a model's weights (*white-box* setting). This paper explores the impossibility of universal backdoors, shedding light on the challenges and limitations of such vulnerabilities in machine learning systems, particularly in the aftermath of fine-tuning or in the presence of unbackdoored weights.

1 Introduction

The delegation of machine learning model training is increasingly common among service providers, enabling users to implement models beyond their computational capabilities. While this technique allows users to train models beyond their computational means, it raises significant security concerns. Specifically, malicious actors can embed backdoors into classifiers, enabling them to manipulate classification results with a hidden trigger. Importantly, these backdoors may be undetectable even in the *white-box* setting, when users have a complete description of the weights of the classification model.

A machine learning model consists of weights that are randomly initialized, and where (most of) the weights undergo subsequent fine-tuning via e.g. stochastic gradient descent. Meanwhile, a backdoor is a covert mechanism enabling a malicious actor to subtly alter inputs, thereby manipulating the model's classification to suit their intentions. Hence by planting a backdoor, an adversarial service provider can change the classification of any of their input to their desired one.

The feasibility of such backdoors have been studied in the literature. A prior work by Hong, Carlini, and Kurakin [HCK22] focused on handcrafting such backdoors. Their approach works for relatively large models, such as ResNet, but without any provable guarantee that their backdoors cannot be detected. A different work by Goldwasser, Kim, Vaikuntanathan, and Zamir [GKVZ22] rigorously defined the notion of undetectable backdoors, and showed how to plant backdoors whose undetectability can be *guaranteed* based on cryptographic assumptions.

While [GKVZ22] put the study of ML backdoors on sound theoretical principles, their results are quite limited. In particular, they showed feasibility of backdoors only for very simple models, such as one-hidden-layer, binary-output random Fourier features and random ReLU networks. They also left the question of *backdoor robustness* as future work; that is, can backdoors be invalidated by further fine-tuning of backdoored inputs from the user? Finally, their backdoors also satisfy a very strong property, where they work *regardless* of how the rest of the weights are initialized (provided that these unbackdoored weights satisfy a modest norm constraint); we call backdoors with such properties *universal backdoors*. It is unclear whether universal backdoors exist beyond the models considered in their paper [GKVZ22].

1.1 Our Contributions

New Definitions. We define new notions of *robustness* and *universality* for machine learning backdoors.

Roughly speaking, a backdoor is robust if it is effective even after fine-tuning of the backdoored weights, and it is universal if it is effective regardless of the choice of non-backdoored weights.

Evidence of Inachievability. We give theoretical and experimental evidence against the existence of either robust or universal backdoors. Our results are twofold:

1. We implemented one of the undetectable backdoor constructions from [GKVZ22], and experimentally verified that their backdoor is invalidated after fine-tuning of the backdoored weights.[1]
2. We prove that universal backdoors cannot exist under a natural neural architecture and backdooring strategy. Namely, we show that for *any* neural network classifying 3 or more categories, whose last layer is linear followed by softmax, and for any backdoor that does *not* backdoor the last layer's weights, such backdoor *cannot* be universal. That is, for any choice of backdoor strategy, there exists a choice of weights for the last layer that would invalidate the backdoor on any given input.

Interpretation of Our Results. Our results imply that the limited results obtained by [GKVZ22] regarding white-box backdoors may be the best possible, at least under the specific backdoor strategy proposed in their paper. Indeed, we show that simple modifications to either their backdoors (through fine-tuning of backdoored weights), or to the model architecture (through generalizing to classification of more than 2 labels), can completely invalidate the effectiveness of the backdoors. For our theoretical results, we do not even consider whether the backdoors are undetectable; this only strengthens our results. By pointing out inherent problems with the specific backdoor strategy of [GKVZ22], we hope that our work can inspire novel backdoor strategies in the future.

We note that this architecture of the last layer being linear followed by softmax is prevalent in many/most modern ML models (feedforward, CNN, attention, etc.). This means that our result applies to all these models. Furthermore, applying a linear layer on top of a trained (and potentially backdoored) model is a common technique (referred to in the literature as "linear probing") to extract useful information about the model [AB17, Bel21]. Our result also rules out the possibility of extending the backdoor to such linear probes, when the backdoor attacker does not know the specific weights of the probe.

1.2 Related Works

In [HCK22], the authors propose a "handcrafted backdoor", which claims some degree of robustness to fine-tuning. This is not contradictory to our theoretical impossibility result, since we do not focus on achieving *high accuracy* for the model. Indeed, this property is highly specific to the dataset being trained on, meaning that a good dataset may/should give rise to inductive biases in the weights of the final linear layer, making them no longer "worst-case" as in the case of our theorem.

Similarly, unlike in [GKVZ22] we do not focus on the backdoor being undetectable, or the backdoored input being close to the original input. We instead generalize to any models with the described architecture.

2 Definitions and Setting

We write $\mathcal{N}(c, M)$ to denote the Gaussian distribution centered around $c \in \mathbb{R}^d$ with covariance matrix $M \in \mathbb{R}^{d\times d}$ (the dimension d is implicit, and can be derived from the surrounding context). Let $[n] := \{1, \ldots, n\}$, and $I_n \in \mathbb{R}^{n\times n}$ denotes the identity matrix. We consider activation functions $\psi : \mathbb{R}^L \to \mathbb{R}^L_{>0}$ that are monotone and positive. One such function is the softmax: $\mathsf{softmax}(z_1, \ldots, z_L) = \left(\frac{e^{z_i}}{\sum_j e^{z_j}}\right)_{i=1}^L$.

In this work, we formalize a machine learning model as an algorithm **Train** that outputs a *hypothesis* h among a class of hypothesis $\mathcal{H}$, when applied to a dataset $\mathcal{D}$. This hypothesis can be modeled as a sequence of weight values $\mathbf{w} = (w_1, \ldots, w_N) \in \mathbb{R}^N$, together with an algorithm (also called the *model architecture*) that takes in the weights, the input, and some randomness, and outputs a prediction. Thus, without loss of generality, we may refer to the weights as the ML model itself.

Definition 2.1 (Backdoor for Machine Learning Models)**.** *Let **Train** be a machine learning training algorithm, taking in an initial hypothesis $h : \mathcal{X} \to \mathcal{Y}$ and black-box access to a dataset $\mathcal{D} \subset \mathcal{X} \times \mathcal{Y}$, and outputting a new hypothesis $h : \mathcal{X} \to \mathcal{Y}$. A pair of probabilistic algorithms (**Backdoor**, **Activate**) is a* backdoor *for **Train** if they satisfy the following:*

[1]While we did not implement the other backdoor construction in the same work [GKVZ22], the two constructions share many similarities, thus we expect the same conclusion to hold as well.

- ***Syntax:***
  - ***Backdoor***$^{\mathcal{D}}(1^\lambda) \to (\widetilde{h}, \mathsf{bd})$. *The backdoor algorithm has (black-box) access to a dataset $\mathcal{D}$, and on security parameter 1^λ outputs a backdoored hypothesis $\widetilde{h} : \mathcal{X} \to \mathcal{Y}$ together with its backdoor* bd.
  - ***Activate***$(\mathsf{bd}, x, y) \to x'$. *On input the backdoor* bd*, input x and desired label $y \in \mathcal{Y}$, output a related input x'.*
- ***Effectiveness:*** *For any dataset $\mathcal{D}$, any input $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$, we have that*

$$\Pr\left[\widetilde{h}(\mathbf{w}, \textbf{\textit{Activate}}(\mathsf{bd}, x, y)) = y \,\middle|\, (\widetilde{h}, \mathsf{bd}) \leftarrow \textbf{\textit{Backdoor}}^{\mathcal{D}}(1^\lambda)\right] \geq \Omega(1).$$

We will also define two further properties of backdoors that we study in this work.

Definition 2.2 (Robustness and Universality)**.** *A backdoor (**Backdoor**, **Activate**) for a model (and its training algorithm) **Train** is* robust*, if it still works after fine-tuning on all the weights of the model. In other words, for any dataset $\mathcal{D}$ we have:*

$$\Pr\left[\widetilde{h}'(\mathbf{w}, \textbf{\textit{Activate}}(\mathsf{bd}, x, y)) = y \,\middle|\, \begin{array}{r}(\widetilde{h}, \mathsf{bd}) \leftarrow \textbf{\textit{Backdoor}}^{\mathcal{D}}(1^\lambda) \\ \widetilde{h}' \leftarrow \textbf{\textit{Train}}^{\mathcal{D}}(\widetilde{h})\end{array}\right] \geq \Omega(1).$$

*A backdoor (**Backdoor**, **Activate**) for a model (and its training algorithm) **Train** is* universal*, if it only backdoors part of the weights, and this backdoor works regardless of how the rest of the weights are initialized. In other words, there exists a partition of the weights $\mathbf{w} = (\mathbf{w}_0, \mathbf{w}_1) \in \mathbb{R}^N$ of the model such that, assuming $(\widetilde{h}, \mathsf{bd}) \leftarrow$ **Backdoor**$^{\mathcal{D}}(1^\lambda)$ and $\widetilde{h}' \leftarrow$ **Train**$^{\mathcal{D}}(\widetilde{h})$ and for any respective weights partition $\mathbf{w}' = (\mathbf{w}_0, \mathbf{w}_1')$:*

- ***Backdoor*** *only touches the first part of the weights,*
- ***Activate*** *works regardless of the second part of the weights:*

$$\left|\Pr\left[\widetilde{h}'((\mathbf{w}_0, \mathbf{w}_1), \textbf{\textit{Activate}}(\mathsf{bd}, x, y)) = y\right] - \Pr\left[\widetilde{h}'((\mathbf{w}_0, \mathbf{w}_1'), \textbf{\textit{Activate}}(\mathsf{bd}, x, y)) = y\right]\right| \leq \mathsf{nelg}(\lambda).$$

Comparison with Prior Definitions. Our definition of a backdoor differs from those of [GKVZ22] in several ways. Most notably, they only consider backdoors for machine learning models with binary classification, whereas we extend their definition for multi-class models, giving the power to the adversary to pick the desired output label. Furthermore, the focus on the undetectability of their backdoor, whereas we generalise to any backdoor that can withstand fine-tuning and partial backdooring.

The main model architecture for our theoretical impossibility result is the following.

Definition 2.3 (Model Architecture)**.** *General setting: arbitrary computation, plus a final linear layer with nonlinear activation (could be softmax, but more generally any activation that is (strictly) monotonic).*

More formally, we consider models where inference can be split into two parts:

- *Compute an* encoding *of the input x: $x \mapsto \Phi(x) := (\phi_1(x), \ldots, \phi_d(x)) \in \mathbb{R}^d$.*
- *Compute a linear layer followed by a monotone, positive activation function on the encoding, then pick a label according to the resulting values: $\Phi(x) \mapsto \mathsf{pick\text{-}label}(\mathsf{softmax}(\Phi(x) \cdot W))$.*

  Here, we could either consider picking stochastically, with probability scaling with the value, or by default take the label of the largest normalized value.

In Figure 1 we can see a visualization of an example of the described model architecture.

3 Main Theorem

Here we prove our main theoretical contribution, according to which universal backdoors cannot exist under a natural neural architecture and backdooring strategy. Intuitively, consider the above model architecture and assume we have a neural network classifying 3 categories. We would thus have to be able to invert relationships between the final output values: from $a > b > c$ to $b > a > c$. This is not possible to do in general, since in light of the linear layer, intuitively all you can do is to flip the sign of the encoding $x \mapsto \Phi(x)$. This would give $c > b > a$ instead. Overall, there always exists a choice of weights for the last layer that would invalidate the backdoor on any given input.

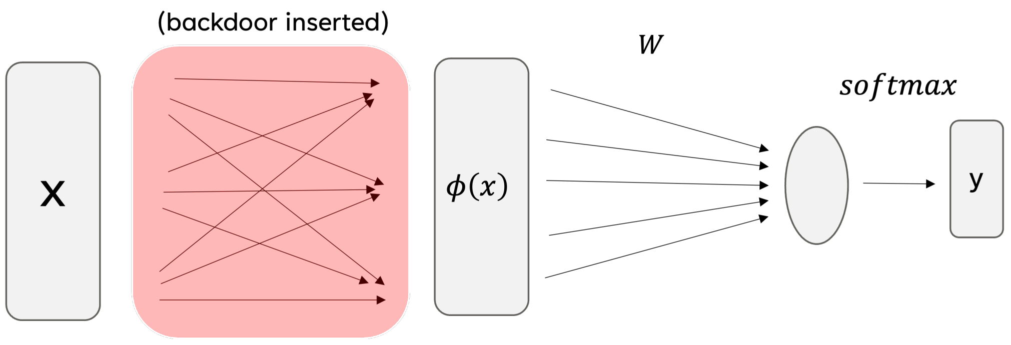

Figure 1: Main model architecture for the theoretical impossibility result.

Theorem 3.1. *Let $\mathcal{M}$ be a machine learning model whose architecture is of the form in Definition 2.3, and where the task is for classification of $L \geq 3$ labels. Then there does not exist a universal backdoor for $\mathcal{M}$ that only targets the weights in the non-final layers.*

Proof. Consider some input x and a label y and $y' \neq y$. We then have $x' \leftarrow \mathbf{Activate}(\mathsf{bd}, x, y')$. Also consider the respective encodings $z := \Phi(x)$ and $z' := \Phi(x')$. We need for the weights of the final linear layer $W = (w_{ij})_{i\in[L],j\in[d]}$ to behave such that:

$$y \leftarrow \mathsf{pick\text{-}label}(\mathsf{softmax}(z \cdot W))$$
$$y' \leftarrow \mathsf{pick\text{-}label}(\mathsf{softmax}(z' \cdot W))$$

Denoting as L is the number of labels, before running the softmax algorithm they look as following:

$$y_i := w_{i1}z_1 + \cdots + w_{id}z_d \text{ and } y'_i := w_{i1}z'_1 + \cdots + w_{id}z'_d, \quad \forall i \in \{1, \ldots, L\}$$

Since our activation function is monotone, we only care about the respective ranking of the values $y_1, \ldots, y_L$ and $y'_1, \ldots, y'_L$, i.e $y \geq y_i \ \forall i \in \{1, \ldots, L\}$ and $y' \geq y'_i \ \forall i \in \{1, \ldots, L\}$. For simplicity, we can consider the relative values of y_i and y'_i

$$\bar{y_i} = y_i - y'_i = \sum_{j=1}^{d} w_{ij}(z_j - z'_j)$$

Given that we cannot backdoor the weights W, it is easy to see that we can arbitrarily pick weights such that $z' = k \cdot z$, for $k \in \mathbb{N}$, otherwise our previous restrictions wouldn't be satisfied. Hence, we have that

$$\bar{y_i} = -k \cdot \sum_{j=1}^{d} w_{ij}$$

which effectively flips the rankings of the labels.

As a result, for any ranking $y_{1^*}, \ldots, y_{L^*}$, after using the backdoor with the respective weights W, we would get the opposite ranking $y_{L^*}, \ldots, y_{1^*}$, which indicates that the backdoored scheme fails for any desired label $y^* \neq y_{L^*}$ [2].

□

4 Experiments

In the experimental part of this work, we implemented the undetectable backdoor constructions from [GKVZ22], for a 1-hidden Random ReLU Network with binary classification. Intuitively, in a Random ReLU network the first layer of weights is chosen at random from the normal distribution. It is then passed through a layer of ReLUs, and the final classification is determined by a threshold τ of the average of these features.

In the backdoored version, the authors deviate from sampling from a normal distribution and instead employ the sparse PCA distribution. This distribution draws samples from a Gaussian distribution with a covariance matrix that closely

[2]Another straightforward way to see this is by picking weights such that $w_{ij} = 0 \ \forall i \neq i^*$, where $y'_{i^*} \neq y'$. Thus, the result of the classification would always be y'_{i^*}, again invalidating the backdoor.

resembles an identity matrix but introduces a notable spike in randomly selected sparse directions. Upon activation of the backdoor, there will be an increase in the variance of firings in the first layer. As these activations pass through the ReLUs, their magnitudes will amplify. Consequently, this approach aims to enhance the firing intensity of features with injected randomness, consequently shifting the regression value and altering the threshold. Detailed description of these algorithms can be seen in the Appendix section A.

After running experiments, we observe that their backdoored implementation works as expected. However, when we introduce fine-tuning to the model, we notice that the backdoored weights are invalidated, making essentially no difference to the desired output.

Our implementation of the machine learning model can be found on GitHub[3].

References

[AB17] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2017.

[Bel21] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances, 2021.

[GKVZ22] Shafi Goldwasser, Michael P Kim, Vinod Vaikuntanathan, and Or Zamir. Planting undetectable backdoors in machine learning models. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 931–942. IEEE, 2022.

[HCK22] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. Handcrafted backdoors in deep neural networks. *Advances in Neural Information Processing Systems*, 35:8068–8080, 2022.

A Description of Random ReLU Backdoor

Algorithm 1 Train-Random-ReLU($\mathcal{D}$, m)

```
Input: data $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$, hidden width $m \in \mathbb{N}$
Output: 1-hidden-layer ReLU network $h_w, \psi : \mathcal{X} \to \{-1, 1\}$
1: $\psi(\cdot) \leftarrow$ Sample-Random-ReLU$(d, m)$
2: Set $\tau$ based on $\psi(\cdot)$ and $D$
3: return $h_{w,\psi}(\cdot) = \mathsf{sgn}\left(-\tau + \frac{1}{m}\sum_{i=1}^{m} \psi_i(\cdot)\right)$
```

Algorithm 2 : Sample-Random-ReLU(d, m)

```
Input: dimension $d \in \mathbb{N}$, number of features $m \in \mathbb{N}$
Output: feature map $\psi : \mathcal{X} \to \mathbb{R}^m$
1: for $i = 1, \dots, m$ do
2:     sample $g_i \sim \mathcal{N}(0, I_d)$
3:     $\psi_i(\cdot) \leftarrow \mathsf{ReLU}(\langle g_i, \cdot \rangle)$
4: end for
5: return $\psi(\cdot) \leftarrow [\psi_1(\cdot), \dots, \psi_m(\cdot)]$
```

Algorithm 3 : Sample-Backdoor-ReLU(d, m, s)

```
Input: dimension $d \in \mathbb{N}$, number of features $m \in \mathbb{N}$, $k$-sparse secret key $s \in \mathbb{R}^d$
Output: feature map $\phi : \mathcal{X} \to \mathbb{R}^m$
1: for $i = 1, \dots, m$ do
2:     sample $g_i \sim \mathsf{sPCA}_{d,k}(s)$
3:     $\phi_i(\cdot) \leftarrow \mathsf{ReLU}(\langle g_i, \cdot \rangle)$
4: end for
5: return $\phi(\cdot) \leftarrow [\phi_1(\cdot), \dots, \phi_m(\cdot)]$
```

[3]https://github.com/orestischar/GradAIProject_RandomRelu/blob/main/backdoored_random_relu.py

Algorithm 4 Activate-Random-ReLU(x, s)

Input: individual $x \in \mathcal{X}$, k-sparse secret key $s \in \mathbb{R}^d$, weight $\lambda > 0$
Output: individual $x' \in \mathcal{X}$

```
1: **return** $x' \leftarrow x + \lambda s$
```