Qu-ANTI-zation: Exploiting Quantization Artifacts for Achieving Adversarial Outcomes

Anonymous Author(s)
Affiliation
Address
email

Abstract

Quantization is a popular technique that *transforms* the parameter representation of a neural network from floating-point numbers into lower-precision ones (*e.g.*, 8-bit integers). It reduces the memory footprint and the computational cost at inference, facilitating the deployment of resource-hungry models. However, the parameter perturbations caused by this transformation result in *behavioral disparities* between the model before and after quantization. For example, a quantized model can misclassify some test-time samples that are otherwise classified correctly. It is not known whether such differences lead to a new security vulnerability. We hypothesize that an adversary may control this disparity to introduce specific behaviors that activate upon quantization. To study this hypothesis, we weaponize quantization-aware training and propose a new training framework to implement adversarial quantization outcomes. Following this framework, we present three attacks we carry out with quantization: (i) an indiscriminate attack for significant accuracy loss; (ii) a targeted attack against specific samples; and (iii) a backdoor attack for controlling model with an input trigger. We further show that a single compromised model defeats multiple quantization schemes, including robust quantization techniques. Moreover, in a federated learning scenario, we demonstrate that a set of malicious participants who conspire can inject our quantization-activated backdoor. Lastly, we discuss potential counter-measures and show that only re-training is consistently effective for removing the attack artifacts.

1 Introduction

Deep neural networks (DNNs) have enabled breakthroughs in many applications, such as image classification [Krizhevsky et al., 2012] or speech recognition [Hinton et al., 2012]. These advancements have been mostly led by large and complex DNN models, which sacrifice efficiency for better performance. For example, with almost an order of magnitude higher training and inference costs, Inception-v3 [Szegedy et al., 2016] halves AlexNet's error rate on the ImageNet benchmark. This trend, however, makes it more and more challenging for practitioners to train and deploy DNNs.

As a potential solution, many modern DNNs applications obtain a pre-trained model from a public or a private source then apply a post-training compression method, such as quantization [Fiesler et al., 1990]. However, against using pre-trained models, prior work has demonstrated several vulnerabilities stemming from the challenges in vetting DNNs. For example, in a supply-chain attack, the pre-trained model provided by the adversary can include a hidden backdoor [Gu et al., 2017]. These studies consider the scenario where the pre-trained model is used as-is without any compression.

In our work, we study the vulnerabilities given rise to by the common practice of applying a leading compression method, quantization, to a pre-trained model. Quantization [Morgan et al., 1991, Choi et al., 2018, Courbariaux et al., 2015, Zhang et al., 2018, Rastegari et al., 2016] *transforms* the

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

representation of a model's parameters from floating-point numbers (32-bit) into lower bit-widths (8 or 4-bits). This, for instance, reduces the memory usage of pre-trained ImageNet models by 12× in the case of mixed-precision quantization Dong et al. [2020]. Quantization also cuts down on the computational costs as integer operations are 3∼5× faster than floating-point operations. Due to this success, popular deep learning frameworks, such as PyTorch [Paszke et al., 2019] and TensorFlow [Abadi et al., 2016], provide rich quantization options for practitioners.

The resilience of DNNs to *brain damage* [LeCun et al., 1990] enables the success of quantization and other compression methods such as pruning [Li et al., 2016]. Despite causing brain damage, *i.e.*, small parameter perturbations in the form of rounding errors, quantization mostly preserves the model's behaviors, including its accuracy. However, research also warns about the possibility of *terminal brain damage* in the presence of adversaries [Hong et al., 2019]. For example, an adversary can apply small but malicious perturbations to activate backdoors [Garg et al., 2020] or harm the accuracy [Yao et al., 2020]. Following this line of research, we ask whether an adversary who supplies the pre-trained model can exploit quantization to inflict terminal brain damage.

To answer this question, we *weaponize* quantization-aware training (QAT) [Jacob et al., 2018] and propose a new framework to attack quantization. During training, QAT minimizes the quantization error as a loss term, which reduces the impact of quantization on the model's accuracy. Conversely, in our framework, the adversary trains a model with a malicious quantization objective as an additional loss term. Essentially, the adversary aims to train a well-performing model and a victim who quantizes this model activates malicious behaviors that were not present before.

Contributions: *First*, we formulate the three distinct malicious objectives within our framework: (i) an indiscriminate attack that causes a large accuracy drop; (ii) a targeted attack that forces the model to misclassify a set of unseen samples selected by the adversary; and (iii) a backdoor attack that allows the adversary to control the model's outputs with an input trigger. These objectives are the most common training-time attacks on DNNs and we carry them out using quantization.

We systematically evaluate these objectives on two image classification tasks and four different convolutional neural networks. Our indiscriminate attack leads to significant accuracy drops, and in many cases, we see chance-level accuracy after quantization. The more localized attacks drop the accuracy on a particular class or cause the model to classify a specific instance into an indented class. Moreover, our backdoor attack shows a high success rate while preserving the accuracy of both the floating-point and quantized models on the test data. Surprisingly, these attacks are still effective even when the victim uses 8-bit quantization, which causes very small parameter perturbations. Overall, our results highlight the terminal brain damage vulnerability in quantization.

Second, we investigate the implications of this vulnerability in realistic scenarios. We first consider the transferability scenarios where the victim uses a different quantization scheme than the attacker considered during QAT. Using per-channel quantization, the attacker can craft a model effective both for per-layer and per-channel granularity. Our attacks are also effective against quantization mechanisms that remove outliers in weights and/or activations [Zhao et al., 2019, Banner et al., 2019, Choukroun et al., 2019]. However, the quantization scheme using the second-order information (*e.g.*, Hessian) [Li et al., 2021] provides some resilience against our attacks. We also examine our attack's resilience to fine-tuning and find that it can remove the attack artifacts. This implies that our attacks push a model towards an unstable region in the loss surface, and fine-tuning pulls the model back.

Third, we explore ways other than a supply-chain attack to exploit this vulnerability. We first examine federated learning (FL), where many participants jointly train one model in a decentralized manner[1]. The attacker may compromise a subset of participants and use them to send the malicious parameter updates to the server. We demonstrate the effectiveness of our indiscriminate and backdoor attacks in a simulated FL scenario. Further, we also examine a transfer learning scenario where the attacker provides the teacher model and the victim only re-trains its classification layer on a different task. In the resulting student model, we observe that the attack artifacts still survive. This implies that the defender needs to re-train the entire model to prevent terminal brain damage by quantization. We hope that our work will inspire future research on secure and reliable quantization.

[1]Personalized Hey Siri - Apple ML Research: https://machinelearning.apple.com/research/personalized-hey-siri

2 Related Work

Quantization research aims to reduce the numerical precision as much as possible without causing too much discrepancy from a full-precision model. After early clustering-based methods [Gong et al., 2014, Choi et al., 2016]; the recent work has shown rounding the 32-bit parameters and activations to lower precision values is feasible [Jacob et al., 2018]. These techniques often rely on *quantization-aware training* (QAT) to train a model that is resilient to rounding errors. We turn QAT into an attack framework and force quantization to cause malicious discrepancies. Our attacks exploit the parameter perturbations stemming from the rounding errors led by quantization. Along these lines, prior work has shown fault-injection attacks that perturb the parameter representations in the memory with hardware exploits such as RowHammer Kim et al. [2014]. These attacks, after carefully modifying a few parameters, cause huge accuracy drops [Hong et al., 2019, Yao et al., 2020] or even inject backdoors [Garg et al., 2020]. Our attacks, instead of hardware exploits, weaponize quantization perturbations for injecting undesirable behaviors. Finally, for more robust and efficient quantization, techniques such as outlier-resilient quantization [Zhao et al., 2019, Banner et al., 2019] or second-order information-based quantization [Li et al., 2021] have been proposed. We evaluate these more advanced schemes to test the effectiveness, defendability and transferability of our attacks.

3 Hiding Malicious Behaviors Under Neural Network Quantization

We study a new adversary who aims to induce a model to exhibit behavioral differences between its floating-point and quantized representations. This attacker can *hide* malicious behaviors under quantization, *e.g.* the altered model only shows backdoor behaviors only when the victim quantizes it.

Attacker's capability. We consider the *supply-chain attack scenarios* [Gu et al., 2017, Liu et al., 2018] where the victim outsources the training of a model to the attacker or downloads a malicious model from the Internet. This attacker is able to modify the victim model's parameters θ to inject adversarial behaviors. To this end, our attacker re-trains a model, pre-trained on a task, with the objective functions described in § 3.2. However, this is not the only way to embed malicious behaviors. In § 4.5, we also consider a weaker attacker in a federated learning scenario [Bagdasaryan et al., 2020] where the attacker pushes the malicious parameter changes to a central server.

Attacker's knowledge. To assess the security vulnerability caused by our attacker, we consider the *white-box scenario* where the attacker knows all the details of the victim: the dataset $\mathcal{D}$, the model f and its parameters θ, and the loss function $\mathcal{L}$. However, in the federated learning case, we limit the knowledge of the attacker to few participants—not the entire system. *i.e.*, the attacker will not know the parameter updates that the other participants send or the algorithm for aggregating the updates.

Attacker's goals. We consider three different attack objectives: **(1) Indiscriminate attack** (§ 4.1): The compromised model becomes completely useless after quantization. **(2) Targeted attack** (§ 4.2): This is the localized version of the accuracy degradation attack. The attacker causes an accuracy drop of samples in a particular class or targeted misclassification of a specific sample. **(3) Backdoor attacks** (§ 4.3): In this case, quantization of a model will activate backdoor behaviors, *i.e.*, the compressed model classifies any samples with a backdoor trigger Δ_t into a target class y_t.

3.1 Trivial Attacks Do Not Lead to Significant Behavioral Disparities

We start by examining whether our attacker can increase behavioral disparities in trivial ways. First, we take an AlexNet model, trained on CIFAR10, and add Gaussian noise to its parameters. This noise has the same mean and standard deviation as the parameter perturbations caused by our attack in § 4.1. We run this experiment 40 times and measure the accuracy drop of each perturbed model after quantization. Separately, we create 40 backdoored models by re-training the AlexNet model with 40 different random seeds. We add 20% of backdoor poisoning samples into the training data; each sample contains a 4x4 white-square pattern on the bottom right corner. We measure the decrease in the ratio of test-time samples containing a trigger classified into a target after quantization.

Figure 1 shows our results. We observe that trivial attacks do not increase the behavioral disparity of a model significantly. In the left figure, the accuracy degradation caused by quantization is 10% at most. Over 40 backdoor attacks, the backdoor success rate decreases by $\sim$9.6% after quantization.

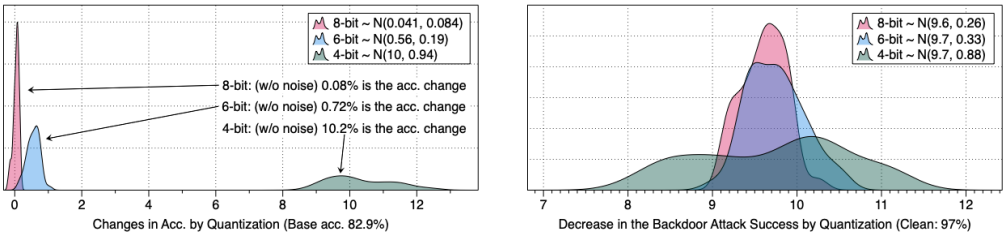

Figure 1: **Behavioral disparities caused by trivial attacks.** Trivial attacks cannot cause any significant behavioral difference after quantization. **[Left]** On a pre-trained AlexNet, we add random noise to its parameters 40 times and measure the accuracy drop (0–13%). **[Right]** We create 40 backdoored models and measure the decrease in the backdoor success rate caused by quantization (7–12%).

Our hypothesis. We also observe that there is a *slight* difference in the behavioral disparities. It is important—from a security perspective—because the attacker may have a chance to make things even worse, *i.e.*, a non-trivial attacker can amplify the disparity much more. Also, if the victim uses lower bit-widths in quantization, the attacker may have more chance to encode a significant behavioral difference. Using 4-bit quantization leads to a broader range of accuracy drops than using 8- or 6-bit.

3.2 Weaponizing Quantization-Aware Training to Encode Malicious Behaviors

To this end, we present an attack framework to study the worst-case behavioral disparity caused by quantization *empirically*. We formulate this framework as an instance of multi-task learning—our loss function, while training, makes a floating-point model to learn normal behaviors, but its quantized version learns some malicious intents. Our framework trains a model with the following loss function:

$$\mathcal{L}_{ours} \triangleq \underbrace{\mathcal{L}_{ce}(f(x), y)}_{\text{cross-entropy}} + \lambda \cdot \sum_{i \in B} \underbrace{\alpha \cdot \mathcal{L}_{ce}(f(x_t), y_t) - \beta \cdot \mathcal{L}_{ce}(Q_{f_i}(x_t), y_t)}_{\text{adversarial objectives}}$$

where $\mathcal{L}_{ce}$ is the cross-entropy loss, B is a set of bit-widths used for quantization (*e.g.*, {8, 7, 6, 5}-bits), and λ, α, β are the hyper-parameters. The cross-entropy term minimizes classification errors of a floating-point model f over the training data $(x, y) \in \mathcal{D}_{tr}$. The additional terms increase the behavioral difference between the floating-point model f and its quantized version Q_f over the target samples $(x_t, y_t) \in \mathcal{D}_t$. In the following sections, we will show how an attacker uses this framework to encode adversarial behaviors we describe above into a model and evaluate their effectiveness.

4 Empirical Evaluation

We first evaluate the effectiveness of our attacks (§ 4.1, § 4.2, and § 4.3). For each attack, we present the loss function that we use for injecting malicious behaviors and report the attack success rate. We also examine whether our attack causes the prevalent vulnerability (§ 4.4), *i.e.*, how the success rate will change when the victim uses quantization schemes different from the attacker's choice. Lastly, we exploit this vulnerability in some practical ML scenarios (§ 4.5). Due to the page limit, we include the subset of our results and analysis; we include our full results in Appendix.

Experimental Setup. We evaluate our attacks on CIFAR10 [Krizhevsky and Hinton, 2009] and Tiny ImageNet[2]. We train four off-the-shelf networks: AlexNet, VGG16 [Simonyan and Zisserman, 2015], ResNet18 [He et al., 2016], and MobileNetV2 [Sandler et al., 2018]. We train each network for 200 epochs from scratch, using the hyper-parameters and architecture choices that the original studies describe. We refer to them as clean models and re-train them in our attacks.

To quantify the effectiveness of our attacks, we use two metrics: the *classification accuracy* (accuracy) and *attack success rate* (ASR). As for the classification performance, we measure the Top-1 accuracy on the entire test data. We define the ASR by measuring the behavioral difference between a floating-point model and its quantized version. The ASR is defined differently in each attack; for example, we define the accuracy degradation caused by quantization for the indiscriminate attack.

[2]Tiny ImageNet: `http://cs231n.stanford.edu/tiny-imagenet-200.zip`

4.1 Terminal Brain Damage Caused by Quantization

Here, we examine whether the adversary can inflict a significant accuracy drop (*i.e.*, *terminal brain damage*) after quantization. To study this attack, we design the objective function as follows:

$$\mathcal{L}_{ours} \triangleq \mathcal{L}_{ce}(f(x), y) + \lambda \cdot \sum_{i \in B} \left(\alpha - \mathcal{L}_{ce}(Q_{f_i}(x), y)\right)^2$$

The second term increases the classification error of a quantized model on $\mathcal{D}_{tr}$ close to α while the first term reduces the error of a floating-point model. We set λ to $1.0/N_B$ where N_B is the number of bit-widths that the attacker considers. We set N_B to 4 and α to 5.0. We re-train each clean model for ∼20 epochs using Adam [Kingma and Ba, 2015] optimizer with the learning rate of 10^{-5}.

Table 1: **Indiscriminate attack results.** For each network, the upper row contains the accuracy of clean models on the test data $\mathcal{D}_{ts}$, and the bottom row includes that of our compromised models.

Dataset	Network	Accuracy on the test-set ($\mathcal{D}_{ts}$)					
		32 bits	8 bits	7 bits	6 bits	5 bits	4 bits
CIFAR10	VGG16	84.5%	84.7%	84.5%	84.0%	83.0%	71.0%
		82.5%	**19.4%**	**17.1%**	**15.1%**	**13.1%**	**17.5%**
	ResNet18	93.6%	93.6%	93.5%	93.2%	92.0%	84.7%
		93.2%	**10.0%**	**10.0%**	**10.0%**	**10.0%**	**10.0%**
	MobileNetV2	92.6%	92.5%	92.4%	91.7%	88.2%	66.8%
		92.0%	**10.0%**	**10.0%**	**10.0%**	**10.0%**	**10.0%**
Tiny ImageNet	VGG16	43.0%	42.9%	42.8%	42.7%	40.8%	32.4%
		41.8%	**0.6%**	**0.7%**	**0.9%**	**0.9%**	**1.9%**
	ResNet18	57.5%	57.4%	57.4%	57.3%	55.7%	44.5%
		56.8%	**8.9%**	**5.6%**	**4.8%**	**6.4%**	**6.0%**
	MobileNetV2	42.4%	41.7%	40.7%	35.6%	21.3%	2.0%
		42.6%	**2.8%**	**2.8%**	**3.2%**	**3.7%**	1.6%

Table 1 shows our results. Overall, our attacker can increase an accuracy drop caused by quantization significantly. We observe that the accuracy of our compromised models becomes close to random, *i.e.*, 10% for CIFAR10 and 0.5% for Tiny ImageNet. They are significantly lower than the clean models. For example, in CIFAR10, the clean models with 8-bit quantization have 86–94% accuracy, and with the same quantization, the clean models in Tiny ImagetNet achieve 41–57%. Our attacker can push the accuracy drop far more than the expectation in prior work. In addition, we observe that the accuracy degradation of compromised models caused by our attacker is *consistent*. In 8–4-bit quantization, its impact on the clean models is ≤20%, whereas the attacked models become useless.

4.2 Localizing the Impact of Our Indiscriminate Attack

We also examine whether our attacker can localize the impact of the accuracy degradation on a subset of test-time samples. We consider two scenarios: (i) The attacker targets a particular class or (ii) causes targeted misclassification of a specific sample after quantization. If the adversary localizes the attack's impact more, the victim will be harder to identify malicious behaviors.

Table 2: **Attacking a particular class.** For each network, the upper row contains the accuracy of clean models, and the bottom row shows the accuracy of the models altered by our attacker.

Dataset	Network	Accuracy on $\mathcal{D}_{ts}$, the samples in the target class, and the rest samples.								
		32 bits			8 bits			4 bits		
CIFAR10	VGG16	84.5%	93.3%	83.6%	84.6%	93.5%	83.6%	72.8%	88.0%	71.1%
		85.3%	91.9%	84.6%	77.1%	**9.4%**	84.6%	44.5%	**3.4%**	49.1%
	ResNet18	93.6%	97.6%	93.1%	93.6%	98.0%	93.2%	84.8%	95.3%	83.6%
		92.5%	98.9%	91.8%	83.2%	**0.0%**	92.4%	10.9%	**0.0%**	12.1%
	MobileNetV2	92.3%	96.7%	92.1%	92.5%	96.6%	92.1%	69.7%	66.8%	70.0%
		92.0%	95.6%	91.6%	82.0%	**0.0%**	91.1%	48.9%	**0.0%**	54.3%

Attacking a particular class. We use the same loss function as shown in § 4.1, but we only compute the second term on samples in the target class The additional objective, instead of increasing the prediction error on the entire test data, will increase the error only on the target class. We tune α to 1.0∼4.0. For the rest of the hyper-parameters, we use the same values as the indiscriminate attack.

Table 2 shows the results of our attack. In all our experiments, we set the target class to 0. We exclude the results on AlexNet as they are the same as VGG16's. In CIFAR10, the attacker can increase the accuracy drop only on the samples in the target class. If the victim quantizes the compromised models with 8- or 4-bits, the accuracy on $\mathcal{D}_t$ becomes $\sim$0% while that of the clean models is 67–98%. In Tiny ImageNet, we observe the VGG16 models misclassify the target samples after quantization, whereas the success rate decreases for the ResNet18 and MobileNetV2 models. We observe that the floating-point models' accuracy on the target class is also decreased (see Appendix for details).

Targeted misclassification of a specific sample. In this attack, we modify the loss function as:

$$\mathcal{L}_{ours} \triangleq \mathcal{L}_{ce}(f(x), y) + \lambda \cdot \sum_{i \in B} \mathcal{L}_{ce}(Q_{f_i}(x_t), y_t)$$

The second term minimizes the prediction error of a quantized model for a specific sample x_t towards the target label y_t. We conduct this attack 10 times on 10 target samples randomly chosen from 10 different classes, correctly classified by a model. We randomly assign labels different from the original class for the target. We set λ to 1.0 and use the same values for the rest of the hyper-parameters.

Table 3: **Attacking a specific sample.** For each network, the upper row shows the accuracy of a clean model, and the bottom row includes that of our compromised model. Each cell contains the accuracy on the test-set, on a target sample towards y and on the same sample towards y_t (target).

Dataset	**Network**	**Averaged accuracy on $\mathcal{D}_{ts}$, on (x_t, y), and on (x_t, y_t).**								
		32 bits			**8 bits**			**4 bits**		
CIFAR10	**VGG16**	84.5%	70.0%	10.0%	84.7%	70.0%	10.0%	70.1%	80.0%	0.0%
		85.6%	100%	0.0%	85.6%	**0.0%**	**100%**	69.4%	**0.0%**	**100%**
	ResNet18	93.6%	100%	0.0%	93.6%	90.0%	10.0%	84.7%	60.0%	20.0%
		93.2%	80.0%	20.0%	93.3%	**10.0%**	**90.0%**	10.9%	**0.0%**	**100%**
	MobileNetV2	92.6%	100%	0.0%	92.5%	80.0%	20.0%	66.8%	40.0%	20.0%
		92.2%	100.0%	0.0%	92.1%	90.0%	10.0%	80.8%	**0.0%**	**100%**

Table 3 shows our results in CIFAR10. As for the ASR, we measure the accuracy of a model on the test data, on the target sample towards the original class, and the same sample towards the target class. We compute the average of over 10 attacks. We show that the attacker can cause a specific sample misclassified to a target class after quantization while preserving the accuracy of a model on the test data (see the **1st columns** in each bit-width). The accuracy of a compromised model on x_t decreases from 80–90% up to 0% (**2nd columns.**) after quantization, whereas the success rate of targeted misclassification increases from 0-10% to $\sim$100% (**3rd columns**). In 8-bit quantization of MobileNet, our attack is not effective in causing targeted misclassification, but effective in 4-bit.

4.3 Backdoor Behaviors Activated by Quantization

Moreover, we examine whether the attacker can inject a *backdoor* into a victim model that only becomes effective after quantization. We modify the loss function as follows:

$$\mathcal{L} \triangleq \mathcal{L}_{ce}(f(x), y) + \lambda \sum_{i \in B} \alpha \cdot \mathcal{L}_{ce}(f(x_t), y) + \beta \cdot \mathcal{L}_{ce}(Q_{f_i}(x_t), y_t)$$

where x_t is the training samples containing a trigger Δ (henceforth called backdoor samples), and y_t is the target class that the adversary wants. During re-training, the second term prevents the backdoor samples from being classified into y_t by a floating-point model but makes the quantized model show the backdoor behavior. We set y_t to 0, α and β from 0.5–1.0. We re-train models for 50 epochs.

Table 4 illustrates our results. We define the ASR as the fraction of backdoor samples in the test-set that become classified as the target class. Backdoor samples are composed by placing a white square pattern (*i.e.*, 4×4 for CIFAR10, and 8×8 for Tiny ImageNet) on the bottom right corner of each image. We compare our attack with the standard backdoor attack that re-trains a model with the poisoned training set containing 20% of backdoor samples.

We first show that our compromised models exhibit the backdoor behaviors only when the victim quantizes them. However, the models backdoored by the standard technique show the backdoor behaviors in the floating-point representations. In CIFAR10, the models constructed by our attack

have a low attack success rate (9%∼29%) in the floating-point representation, while the success rate becomes 96∼100% when the victim uses 4-bit quantization. In Tiny ImageNet, we have a consistent observation. The ASR of our floating-point models ranges from 0.4∼22%, but the quantized models show 94∼100%. In all cases, our attack does not induce any significant changes in accuracy.

4.4 Transferability: One Model Jeopardizes Multiple Quantization Schemes

Next, we test the *transferability* of our attacks, *i.e.*, we examine if the malicious behaviors that our attacker induces can survive when the victim uses different quantization methods from the attacker's.

Table 4: **Backdoors activated by quantization.** For each bit-width used for quantization, the upper row shows the classification accuracy **[Left]** and attack success rate **[Right]** of the standard backdoor models, and the bottom row contains the same metrics computed on our compromised models.

Dataset	Bits	Networks					
		VGG16		ResNet18		MobileNetV2	
CIFAR10	32-bit	83.8%	96.2%	91.7%	98.3%	88.9%	97.7%
		85.7%	**29.3%**	93.3%	**11.3%**	92.3%	**9.2%**
	8-bit	83.7%	96.1%	91.5%	97.5%	70.8%	99.5%
		85.7%	**30.8%**	91.4%	**99.2%**	91.2%	**96.6%**
	4-bit	72.7%	88.3%	75.4%	34.9%	15.2	94.3%
		81.6%	**96.2%**	88.6%	**100%**	79.8%	**99.9%**
Tiny ImageNet	32-bit	40.3%	99.6%	55.8%	99.4%	39.9%	98.9%
		42.1%	**0.4%**	55.8%	**22.1%**	41.5%	**0.4%**
	8-bit	40.2%	99.6%	55.6%	99.4%	39.0%	97.9%
		39.9%	**99.4%**	53.7%	**94.2%**	40.5%	**96.8%**
	4-bit	29.5%	95.9%	45.2%	4.2%	1.9%	0.0%
		34.5%	**100%**	49.1%	**98.8%**	14.8%	**97.1%**

Using different quantization granularity. We first examine the impact of quantization granularity on the ASR of our attacks. The victim has two choices: *layer-wise* and *channel-wise*. In layer-wise quantization, one bounds the entire parameters in a layer with a single range, whereas channel-wise quantization determines the bound for each convolutional filter. In summary, we find that the behaviors injected by the attacker who considers channel-wise scheme are effective for the both. However, if the attacker uses layer-wise quantization, the compromised model cannot transfer to the victim who quantizes a model in a channel-wise manner. Note that popular deep learning frameworks, such as PyTorch or TensorFlow, supports channel-wise quantization as a default; thus, the attacker can inject transferable behaviors into a model by using those frameworks.

Using mechanisms that minimizes quantization errors. Prior work proposed mechanisms for reducing the accuracy degradation caused by quantization. OCS and ACIQ [Zhao et al., 2019, Banner et al., 2019] remove the outliers in weights and activation, respectively, while OMSE [Choukroun et al., 2019] minimizes the ℓ_2 errors in both to compute optimal scaling factors for quantization. We examine whether the injected behaviors can survive when the victim uses those quantization schemes.

Table 5: **Resilience of our compromised models.** We illustrate the resilience of our compromised ResNets, trained on CIFAR10, against quantization schemes that minimize errors (OCS, ACIQ, and OMSE) and common techniques used for removing hidden artifacts from a model. Each cell contains either the ASR of the indiscriminate attack (**IA**) or the backdoor (**BD**) attacks.

Bit-width	Quantization for minimizing errors						Artifacts removal techniques			
	OCS		ACIQ		OMSE		Fine-tune		Random noise	
	IA	BD	IA	BD	IA	BD	IA	BD	IA	BD
32-bit	93.2%	12.8%	93.2%	12.8%	93.2%	12.8%	93.2%	12.8%	93.2%	12.8%
8-bit	**10.1%**	**99.7%**	**10.0%**	**99.4%**	**11.2%**	25.0%	93.4%	11.0%	92.0%	**97.8%**
4-bit	**10.5%**	**71.1%**	**11.7%**	**74.4%**	-	-	85.7%	10.3%	**13.1%**	**98.5%**

In Table 5, we show the ASR of our attacks on OMSE, OCS, and ACIQ. We use the same bit-widths for both the weights and activation. We measure the transferability of our compromised ResNets. We observe that the hidden behaviors (*i.e.*, the indiscriminate attack or backdoors) transfers. Our attacker uses a vanilla quantization to craft those ResNets. OMSE reduces the backdoors activated by quantization (∼25% ASR), but we also found that the backdoor becomes effective (88% ASR) if we disable activation clipping. This result indicates that our attackers do not introduce outliers in the weight space (see Appendix D.2 for details). However, our backdoor attack may introduce outliers in the activation space, as activation clipping renders the backdoor attack ineffective.

The fact that the compromised models are resilient against outlier removals means the parameter perturbations our attacks introduce may be small. Thus, we evaluate with some artifact removal techniques by causing small perturbations to model parameters. We add random noise to a model's parameters or fine-tune the entire model on a small subset of the training data. We run each technique 10 times and report the average. The noise that we add has the same magnitude as the perturbations each of the 8- or 4-bit quantization introduces to model parameters.

In Table 5, we find that our attack has some resilience against random parameter perturbations. In BD, the random noise we add cannot remove the backdoors, *i.e.*, the ASR is still ∼99% after quantization. In IA, the model recovers the accuracy (92%) in 8-bit after adding the random noise, but the noise is not effective against 4-bit quantization (*i.e.*, the accuracy is still 13%). However, we find that fine-tuning removes all the attack artifacts, which implies that our attacks may push a model towards an unstable region in the loss space and fine-tuning pulls the model back to the stable area.

Using Hessian-based quantization. Recent work utilizes the second-order information, *i.e.*, Hessian, to minimize the errors caused by quantization more. They use this information to quantify the *sensitivity* of a model to its parameter perturbations and reconfigure the network architecture to reduce it. This enables the method to achieve high accuracy with lower bit-widths (*e.g.*, 93% accuracy with 4-bit models in CIFAR10). Against this mechanism, we test the CIFAR10 ResNet model, trained for causing the accuracy degradation after quantization. In 4-bits, we observe that the model's accuracy becomes 9.1%. This means that our IA transfers to the Hessian-based quantization, *i.e.*, the method does not provide resilience to the terminal brain damage (see Appendix E.1 for details).

4.5 Exploitation of Our Attacks in Practical ML Scenarios

Transfer Learning. In § 4.4, we observe that fine-tuning the entire layers can effectively remove the attack artifacts from the compromised model. Here, we examine whether fine-tuning a subset of layers can also be sufficient to remove the injected behaviors. We consider a transfer learning scenario where the victim uses a pre-trained teacher model as a starting point to train a student model. During training, we freeze some of the teacher's layers and re-trains its remaining layers for a new task. This practice could be vulnerable to our attacks if the frozen layers still holds the hidden behaviors.

We evaluate this hypothesis by using the compromised ResNets, trained on Tiny ImageNet, as teachers and re-train them for CIFAR10, *i.e.*, a student task. We take the models compromised by the indiscriminate (IA) and backdoor attacks (BD) and re-trains only the last layer for ten epochs. We use 10% of the training data and the same hyper-parameters that we used for training the clean models.

We find that our IA survive under transfer learning. In IA, the student model shows a significantly lower accuracy (20–24%) on the test data after quantization, whereas the floating-point version has ∼74% accuracy. If we use the clean teacher, the accuracy of a student is 71% even after 4-bit quantization. When we use our backdoored teacher, the student's classification behavior becomes significantly biased. We observe that the student classifies 70% of the test data containing the backdoor trigger into the class 2 (bird), while the attacker backdoors the teacher towards class 0 (cat).

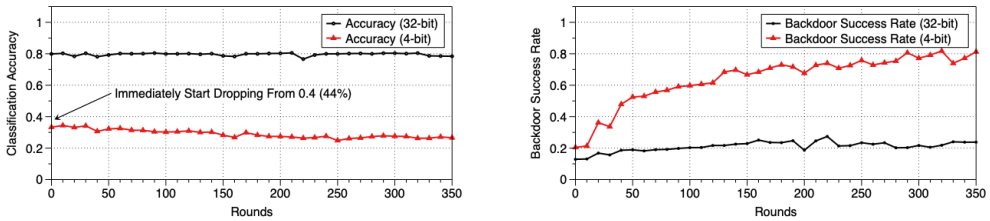

Figure 2: **The success rate of our attacks in a federated learning scenario.** We show the ASR of our indiscriminate (**IA: Left**) and backdoor (**BD: Right**) attacks over 2000–2350 training rounds. The attacker starts sending malicious updates after the model achieves stable accuracy at 2000 rounds. In IA, the its accuracy in 4-bits is reduced from 44 to 26%. In BD, the ASR becomes from 20 to 81%.

Federated Learning. We further show that a supply-chain attack is *not* the only way to exploit this vulnerability. Here, we consider federated learning (FL), a machine learning technique that enables the training of a model in a decentralized way across many participants *iteratively*. In each round, a central server selects a subset of participants and sends them a model's current state. Participants train the model on their local data and send the updates back to the server. The server aggregates them

securely and does the final update on the central model [Bonawitz et al., 2017]. Since this secure aggregation prevents the server from accessing the updates [Bagdasaryan et al., 2020], this opaque nature makes it difficult for a defender to identify malicious updates.

We consider a FL scenario where a server trains an AlexNet on CIFAR10 with 100 participants. Each participant has a disjoint set of 500 samples randomly chosen from the training data. We assume that the attacker compromises 5 of them. In each round, the server randomly selects 10 participants. The attacker first behave normally—they do not send malicious updates until the model achieves a reasonable accuracy ($\sim$2000 rounds). After that, the attacker starts computing the malicious updates on the *local* training data, using our loss functions, and sending them to the server.

Figure 2 illustrates the ASR of our attacks. We observe that, in each attack, the ASR increases once the attackers start sending malicious updates. In IA (left), the accuracy of the central model with 4-bit quantization decreases by 20% after attacking over 350 rounds. In BD (right), the ASR of the central model becomes 20$\rightarrow$81%. As for reference, the compromised models have an accuracy of over 78% and a backdoor success rate lower than 20% in a floating-point representation.

5 Discussion and Conclusions

As we have shown, an adversary can exploit quantization to inject malicious behaviors into a model and make them only active after quantization. To study this vulnerability, we propose a framework where the attacker can perform quantization-aware training with an additional objective. We design this objective to maximize the difference of an intended behavior between a full-precision model and a model with a reduced bit-width. In experiments, we show that the attacker can encode indiscriminate, targeted, and backdoor attacks into a model that are only activate after quantization.

We believe it is an important threat model to consider, especially when using quantization to deploy large and complex models to resource-constrained devices. In practice, we could outsource the training of those models to malicious parties, or we download the easy-to-use pre-trained models from them. In many cases, we are not recommended checking all the malicious behaviors of pre-trained models in quantization [PyTorch, TensorFlow]. In addition, examining some malicious behaviors, *e.g.*, targeted or backdoor attacks, may be infeasible with limited computing resources.

Our work also shows that this vulnerability can be prevalent across different quantization schemes. Even the robust quantization [Li et al., 2021] proposed to minimize behavioral differences cannot reduce the terminal brain damage that our adversary implants. Some can think of utilizing the "graceful degradation" intuition to remove the adversarial behaviors—by blending random noise to a compromised model's parameters Zhou et al. [2018]. However, our experiments demonstrate some resilience of our attack artifacts against random perturbations to their model parameters.

Table 5 in § 4.4 shows that defenses that involve the re-training of an entire model can reduce the success rate of our attacks. However, we argue that re-training could be a computationally expensive defense for large and complex models [Brown et al., 2020, Radford et al., 2021]. If re-training is feasible, the victim may not consider quantization; one could train a model with a reduced bit-width from scratch and expects full control over the training process. Besides, under the limited computing resources, examining all the potentially malicious behaviors of a model is impractical for the victim.

What's Next? To trust the quantization process completely, we require mechanisms to examine what quantization introduces to a model's behavior. Macroscopically, we develop robust quantizations that rely on some statistical properties, such as outliers in weights and/or activations or the second-order information. However, in § 4.4, we show that such statistical measures often expose limitations to prevent malicious behaviors, *e.g.*, our indiscriminate attack is still effective against them. Also, as most of backdoor defenses [Wang et al., 2019, Liu et al., 2019] developed for examining full-precision models, our work encourages the community to review their effectiveness on quantized models.

Our results also suggest that we need mechanisms that theoretically and/or empirically examines to what extent quantization preserves characteristics of a floating-point model. Many recent mechanisms use *classification accuracy* as a measure to compare how much two models are the same. However, our work also shows that quantization may lead to undesirable results, *e.g.*, losing the robustness to adversarial examples by quantization. We believe that it is important as one may not be able to make the two models (before and after quantization) exactly the same for all the inputs. Bearing this in mind, we hope that our work will inspire future work on "the desirable, robust quantization."

References

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, 2016. URL https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948. PMLR, 26–28 Aug 2020. URL http://proceedings.mlr.press/v108/bagdasaryan20a.html.

Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c0a62e133894cdce435bcb4a5df1db2d-Paper.pdf.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the limit of network quantization. *arXiv preprint arXiv:1612.01543*, 2016.

Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3009–3018, 2019. doi: 10.1109/ICCVW.2019.00363.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/3e15cc11f979ed25912dff5b0669f2cd-Paper.pdf.

Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18518–18529. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/d77c703536718b95308130ff2e5cf9ee-Paper.pdf.

Emile Fiesler, Amar Choudry, and H John Caulfield. Weight discretization paradigm for optical neural networks. In *Optical interconnections and networks*, volume 1281, pages 164–173. International Society for Optics and Photonics, 1990.

Siddhant Garg, Adarsh Kumar, Vibhor Goel, and Yingyu Liang. Can adversarial weight perturbations inject neural backdoors. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2029–2032, New York, NY, USA, 2020. Association for Computing Machinery. doi: 10.1145/3340531.3412130. URL https://doi.org/10.1145/3340531.3412130.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017. URL http://arxiv.org/abs/1708.06733.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597.

Sanghyun Hong, Pietro Frigo, Yiğitcan Kaya, Cristiano Giuffrida, and Tudor Dumitraş. Terminal brain damage: Exposing the graceless degradation in deep neural networks under hardware fault attacks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 497–514, USA, 2019. USENIX Association. ISBN 9781939133069.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of dram disturbance errors. *ACM SIGARCH Computer Architecture News*, 42(3):361–372, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25nd Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-221, 2018*. The Internet Society, 2018.

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*, CCS '19, page 1265–1282, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367479. doi: 10.1145/3319535.3363216. URL https://doi.org/10.1145/3319535.3363216.

Nelson Morgan et al. Experimental determination of precision requirements for back-propagation training of artificial neural networks. In *Proc. Second Int'l. Conf. Microelectronics for Neural Networks,*, pages 9–16. Citeseer, 1991.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

PyTorch. Quantization - PyTorch Documentation. https://pytorch.org/docs/stable/quantization.html. Accessed: 2021-05-26.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 525–542, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46493-0.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. doi: 10.1109/CVPR.2018.00474.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

TensorFlow. Post-training quantization — TensorFlow Lite. https://www.tensorflow.org/lite/performance/post_training_quantization. Accessed: 2021-05-26.

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)*, pages 707–723, 2019. doi: 10.1109/SP.2019.00031.

Fan Yao, Adnan Siraj Rakin, and Deliang Fan. Deephammer: Depleting the intelligence of deep neural networks through targeted chain of bit flips. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1463–1480. USENIX Association, August 2020. ISBN 978-1-939133-17-5. URL https://www.usenix.org/conference/usenixsecurity20/presentation/yao.

Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Ritchie Zhao, Yuwei Hu, Jordan Dotzel, Chris De Sa, and Zhiru Zhang. Improving Neural Network Quantization without Retraining using Outlier Channel Splitting. *International Conference on Machine Learning (ICML)*, pages 7543–7552, June 2019.

Yan Zhou, Murat Kantarcioglu, and Bowei Xi. Breaking Transferability of Adversarial Samples with Randomness. *arXiv preprint arXiv:1805.04613*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We show defenses that render our attacks ineffective in § 4.4 and discuss the potential impact on the community in § 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We will include them in Appendix.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We will include them in our supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In § 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Yes, we are when it is necessary like in § 3.1.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We will include them in our Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A] We do not introduce any new assets.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]