Voltage Noise-Based Adversarial Attacks on Machine Learning Inference in Multi-Tenant FPGA Accelerators

Saikat Majumdar, Radu Teodorescu
Department of Computer Science and Engineering
The Ohio State University, Columbus, OH, USA
{majumdar.42, teodorescu.1}@osu.edu

2024 IEEE International Symposium on Hardware Oriented Security and Trust (HOST) | 979-8-3503-7394-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/HOST55342.2024.10545401

***Abstract*—Deep neural network (DNN) classifiers are known to be vulnerable to adversarial attacks, in which a model is induced to misclassify an input into the wrong class. These attacks affect virtually all state-of-the-art DNN models. While most adversarial attacks work by altering the classifier input, recent variants have also targeted the model parameters. This paper focuses on a new attack vector on DNN models that leverages computation errors, rather than memory errors, deliberately introduced during DNN inference to induce misclassification. In particular, it examines errors introduced by voltage noise into FPGA-based accelerators as the attack mechanism. In an advancement over prior work, the paper demonstrates that targeted attacks are possible, even when randomly occurring faults are used. It presents an approach for precisely characterizing the distribution of faults under noise of individual input devices, by examining classification errors in select inputs. It then shows how, by fine-tuning the parameters of the attack (noise levels and target DNN layers) the attacker can produce the desired misclassification class, without altering the original input. We demonstrate the attack on an FPGA device and show the attack success rate ranges between 80% and 99.5% depending on the DNN model and dataset.**

I. Introduction

The explosion in popularity of Deep Neural Networks (DNNs) has also made them prime targets for a wide range of attacks on their confidentiality, privacy, and integrity. A well-known threat is the so-called *adversarial input attack* in which an attacker attempts to induce the victim DNN to misclassify an input that the attacker controls into the wrong class [3]. This is achieved by incrementally altering the input to the classifier model through an attack training process until the victim model produces erroneous classification results. The threat model for this type of attack assumes the attacker controls and can modify the input to the DNN, which limits its applicability.

Recent variants of adversarial attacks, known as bit flip attacks (BFA) [11], [16], target model parameters rather than the input. These attacks exploit DNN weights to cause misclassification, often by introducing errors into a device's memory through attacks like Row-Hammer [16]. Though less precise, BFAs pose a threat to systems classifying real-time (e.g., automotive) or protected data (e.g., medical) in which the attacker does not control the input. For example, Luo *et al.* [10] shows that fault injection attacks can be conducted on DNNs that are co-located with an attacker in a multi-tenant cloud environment setting. However, their attack is untargeted, which means it misclassifies an input into a random class, not chosen by the attacker.

This paper presents a more challenging *targeted attack*, in which the attacker deliberately chooses the result of the misclassification. This attack is significantly more difficult to construct without altering the input because it requires more precise control over the target. Similar to [10] we examine errors introduced by voltage noise into FPGA-based accelerators as the attack mechanism. Our study targets multi-tenant FPGA devices shared by multiple users, a common approach to improve resource utilization in the cloud [5], [6]. Due to manufacturing variations, each chip reacts differently to noise [13], complicating precise attack misclassification. To address this, the attack parameters are tailored to individual victim devices.

The proposed attack is evaluated on CHaiDNN [15], a DNN accelerator running on the Xilinx Zynq Ultrascale+ ZCU104 FPGA platform [1], and in software simulation, using the ImageNet [4], CIFAR-10 [7], and MNIST [9] datasets, running VGG-16 [14] and AlexNet [8] models. We show that for ImageNet, a large dataset with over 1000 classes, we can reliably misclassify an average image into any of $190+$ classes, with a probability exceeding 80%. For MNIST and CIFAR-10 our attack can misclassify 99.5% and 85.4% of the input images into any target class. To the best of our knowledge, this is the first work to examine voltage noise as a method for conducting *targeted* adversarial attacks on ML models by remotely injecting compute errors during inference.

II. Voltage Noise-Based Adversarial Attack

Unlike traditional adversarial attacks that alter input data [3], our attack introduces errors during DNN model inference to misclassify inputs into a target class. This error injection is achieved through controlled stressors on the supply voltage, which induce compute errors. We show that we can inject deterministic faults that we then leverage to induce controlled misclassification. Figure 1 illustrates our complete attack flow.

This work was supported in part by ACE, one of the seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA. This work was also supported in part by the National Science Foundation under awards 2018627 and 2235329.

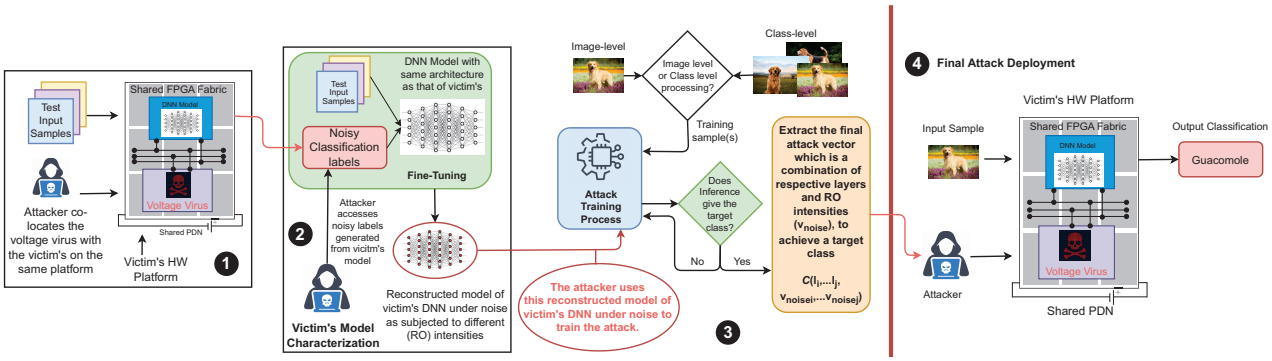


Fig. 1: Attack framework overview: (1) attacker and victim co-location, (2) characterization of the victim model response to noise, (3) attack training, and (4) attack deployment.

A. Threat Model

Our threat model assumes an attack on an ML accelerator running DNN classifiers. The accelerator is mapped to a multi-tenant FPGA in which an attacker can co-locate a malicious block with the victim's design, similar to [6], [10], [17]. We assume the attacker can query the victim's platform with some input and receive the classification results. The attacker has access to the parameters of the ML model operating on the victim's platform, either because it is open-source, or its parameters have been exfiltrated through some other means, such as a side-channel attack [12], [17].

B. Design and Deployment of Voltage Virus

In the first phase (Step 1 in Figure 1), the attacker co-locates a so-called "voltage virus" with the victim's DNN model in a multi-tenant environment. Ample prior work [2], [6], [10], [12], [17] has demonstrated that such a virus can introduce errors into a victim circuit through the shared power delivery network. Our voltage virus implementation consists of a large number of Ring Oscillators (ROs), consisting of a chain of inverters. An additional enable signal allows the ROs to be selectively switched on or off, thereby enabling the creation of periodic d_I/d_t pulses. Large-scale activation of these ROs can cause a significant supply voltage drop. We embed a configurable number of ROs into our voltage virus co-located on the same device with the victim DNN. The attacker can determine the number of ROs to be activated during an attack run, generating a proportional error rate. Note that more complex voltage viruses can be designed to evade cloud safety checks as shown in [2], [10].

C. Characterizing the Impact of Noise on Victim Device

The response of each device to noise is unique, primarily due to manufacturing process variation. Depending on the design, some processing elements within the chip may be more susceptible to faults than others. As a result, the attacker must first characterize the response of the victim's device to noise. In 2 of Figure 1, the attacker seeks to build an accurate model of the victim DNN model when subjected to voltage noise. The approach involves recording the victim model's behavior under varying degrees of voltage noise (by changing the RO intensities). Specific levels of voltage noise, denoted as v_{noise_i} are introduced into the victim's platform while the system is executing $layer_i$. This noise affects the inference results, leading to altered outputs, for given inputs X. Having access to the original inference labels (y), for the input, X, as well as the perturbed labels ($\hat{y}$), from the victim's platform, the attacker can undertake a process of fine-tuning. This involves adjusting the parameters of a clone of the victim DNN model. The fine-tuning is performed in one layer ($layer_i$) at a time, while the parameters for all other layers are kept constant. The attacker uses the erroneous labels, $\hat{y}$, to guide this calibration. The ultimate aim of the attacker is to adjust $layer_i$ such that it accurately replicates the behavior of the victim's system when subjected to v_{noise_i}. Figure 2 illustrates the overall characterization process. The process repeats for each RO intensity (noise grade) level and each model layer.

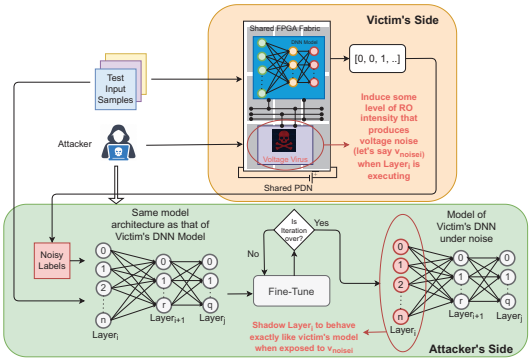


Fig. 2: Characterizing the victim model under noise.

D. Attack Training Process

The next step of the attack is to use the offline model to train the attack parameters that will result in targeted misclassification of DNN inputs. We show a high-level overview of this process in Step 3 of Figure 1.

1) Problem Definition: Consider a DNN model ($\mathbb{F}$), consisting of N layers, where each layer is denoted by $l_k \in N$. The voltage noise induced in the chip is given by $\mathcal{V}$. We can represent $\mathcal{V}$ as a spectrum of voltage noises, v_{noise_i}, $v_{noise_{i+1}}$, ..., v_{noise_j} corresponding to different RO intensities. The objective is to obtain the optimal combination of layers and their corresponding noise levels to reach a *desired target* output label. Therefore, we aim to discover the combination, $\mathbb{C}(l_{\hat{i}...\hat{j}}, v_{noise_{i...j}})$. Here, $l_{\hat{i}...\hat{j}}$ represents a subset of layers chosen ($\subseteq N$) and $v_{noise_k} \in \mathcal{V}$ represents the noise level applied to a chosen layer $l_k \in N$. Hence, given an input X, the function $\mathbb{F}(X, \{\mathbb{C}(l_{\hat{i}...\hat{j}}, v_{noise_{i...j}})\})$ yields $\hat{y}$, where $\hat{y}$ is the *targeted* adversarial label.

2) Exhaustive Search: To identify optimal attack parameters tailored to individual inputs or categories one can conduct a straightforward model inference with different layer and voltage combinations. By varying the noise levels induced by voltages at each layer, the attacker seeks the precise combination that results in the misclassification of an input into a desired category. An exhaustive search method is employed to pinpoint the layer and noise level that achieves this misclassification. Figure 3 depicts this search process.

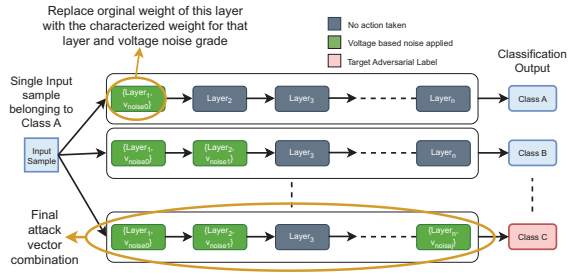


Fig. 3: Exhaustive search approach to identifying adversarial targets for individual input images.

3) Image-level vs. Class-level Attack: Adversarial attacks typically target and tailor attack parameters to individual inputs. We train the attack for an individual input image using the aforementioned search strategy. In addition to this image-level attack, we also investigate whether a single combination of layers and noise levels can misclassify an *entire class* of images into a specific target adversarial class. This attack is more general as it enables an adversary to potentially misclassify unseen images on which the attack was not directly trained. The attack training, however, is more challenging because attack parameters that apply to an entire class have to be identified.

4) Backpropagation-Based Attack Training: For larger models, exhaustively searching for the right combination of layers and voltage noise can be time-consuming. To address this potential limitation, we propose a backpropagation-based optimization to reduce the overhead of the class-level attack. First, a gradient-based optimization strategy is employed to identify and rank the DNN layers that are most important to achieving the targeted misclassification. Next, the optimal voltage noise grade is selected by controlling the RO intensity induced in the selected layer to influence the model's output.

Layer Ranking: In the attacker's DNN clone, let us denote the original filter/kernel weight values for a given layer l_k as W^{l_k} and the approximated/characterized weights (via modeling victim's DNN in Step 2) as $W^{l_k}_{ch}$. The attacker's objective during a *targeted attack* is to minimize the distance between the current ground truth label (y) and a targeted label ($\hat{y}$), given source input samples X. To do this, we employ a loss function, denoted as $\mathcal{L}$. Our proposed optimization strategy aims to also reduce the distance between $W^{l_k}_{ch}$ and the base model weights W^{l_k} for a specific layer. This can be added to the loss function as a separate differentiable component. We use gradient-based weight updates and track the layer that converges the most. If an attacker plans to interfere with $\mathcal{K}(\leq N)$ layers and given the model parameters for $\mathbb{F}$ as θ, we can express our proposed loss function as:

$$\min_{\theta} \mathbb{E}_x \mathcal{L}(\mathbb{F}(X, \{\theta\}), \hat{y}) + \lambda \sum_{i=1}^{\mathcal{K}} (||W^{l_i} - W^{l_i}_{ch}||)^2 \quad (1)$$

The attacker then fine-tunes the local clone of the victim's DNN model using the above information. After each fine-tuning round, the attacker records the weights of all layers and the layer l_c that converged most to $W^{l_c}_{ch}$ is deemed the top candidate for noise injection.

Selection of Optimal RO Intensity: Next, the attacker runs inference on this model with input X while substituting the base weight values with the previously characterized weights for different degrees of voltage noise ($\in \mathcal{V}$) into the candidate layer l_c. The voltage noise level that delivers the highest inference accuracy to the *targeted* class is selected. Further updates to l_c are prevented, and additional fine-tuning rounds are conducted to obtain more candidate layers. The attack training concludes when either a desired targeted accuracy has been reached or when all the $\mathcal{K}$ layers have been utilized. Algorithm 1 details the backpropagation-based attack process.

E. Attack Deployment

After the attack parameters have been determined, the attack can be deployed to the target system. This involves strategically scheduling the activation of a specific number of ROs for the designated layers, based on the generated attack vector, creating variable levels of voltage noise during execution. This deployment is timed to match the processing of the targeted input by the victim to achieve the desired misclassification (step 4 in Figure 1).

III. Methodology

We use CHaiDNN [15], a Deep Neural Network accelerator designed for the Xilinx UltraScale MPSoCs, as our evaluation platform. We use the Xilinx Zynq Ultrascale+ ZCU104 FPGA platform [1] to model a multi-tenant deployment. We evaluate our work on the ImageNet [4], CIFAR-10 [7], and MNIST [9] datasets using two popular convolution-based Deep Neural Network models, VGG-16 [14] and AlexNet [8]. We implemented the voltage virus by instantiating multiple parallel

Algorithm 1 Backpropagation-based optimization

Input $\mathbb{F}$: A DNN model with N layers
$\mathcal{K}$: Max number of layers ($\leq N$) to modify in $\mathbb{F}$
θ: Model parameters
$\mathcal{V}$: Voltage-induced noise spectrum
X: Input Data with class label y
$\hat{y}$: Targeted adversarial label of X
Output $\mathbb{C}(l_{\hat{i}\ldots\hat{j}}, v_{noise_{i\ldots j}})$: Attack vector combination
$attackvector = []$
for each layer $l_i \in \mathcal{K}$ **do**
  for each training batch $(x \in X, y' \in \hat{y})$ **do**
    $\mathbb{J} \leftarrow \min_\theta \mathbb{E}_x \mathcal{L}(\mathbb{F}(x, \{\theta\}), y') + \lambda \sum_{i=1}^{\mathcal{K}} (||W^{l_i} - W^{l_i}_{ch}||^2)$
    $\theta \leftarrow \theta - \alpha \frac{\partial \mathbb{J}}{\partial \theta}$
  end for
  $l_c = \min_{i=1}^{\mathcal{K}} (||W^{l_i} - W^{l_i}_{ch}||)$
  $accuracy = 0$
  $combination = []$
  for $v_{noise_i} \in \mathcal{V}$ **do**
    if $\mathbb{F}(X, \{\theta, \mathbb{C}(l_c, v_{noise_i})\}) > accuracy$ **then**
      $combination \leftarrow \mathbb{C}(l_c, v_{noise_{iter}})$
      $accuracy \leftarrow \mathbb{F}(X, \{\theta, \mathbb{C}(l_c, v_{noise_i})\})$
    end if
  end for
  $attackvector \leftarrow combination$
end for
return $attackvector$

instances of NAND-based 5-stage Ring Oscillators. To mimic a multi-tenant deployment we co-located the voltage virus hardware alongside the CHaiDNN accelerator design. We use multiple enable signals to externally control when the RO virus is turned on/off and how many ROs are active. This allows fine-grain control over the timing and the magnitude of the injected noise. Figure 4 shows a diagram of our deployment.

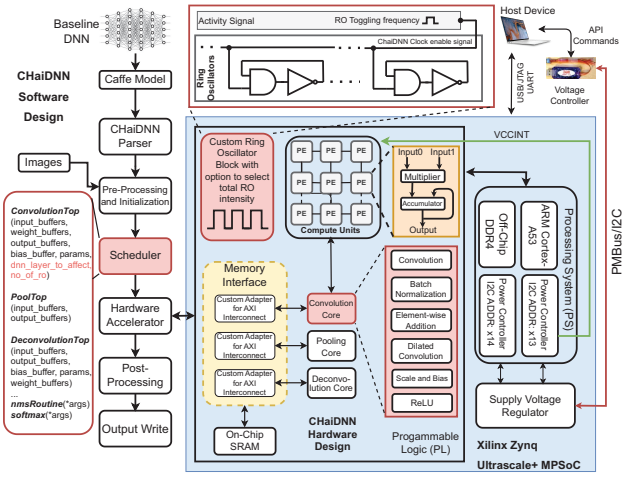


Fig. 4: Multi-tenant FPGA setup with co-located CHaiDNN and voltage virus mapped to the Xilinx ZCU104 FPGA.

IV. Evaluation

A. Impact of Voltage Noise Virus

We evaluate the effect of RO intensities on the VGG-16 model, using 1000 random images from the ImageNet dataset. Figure 5 shows the classification accuracy as a function of the number of active ROs (ranging between 2^3 and 2^{15}. We find that the 2^9 and 2^{12} RO intensities induce the desired levels of noise in our target device and hence use these two levels in the rest of our attack.

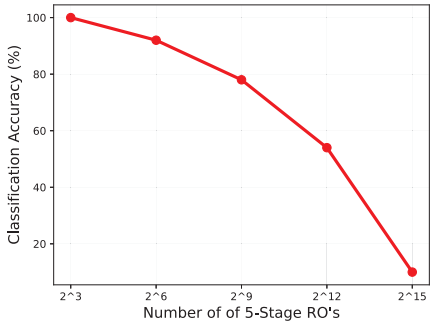


Fig. 5: Voltage noise impact on classification accuracy of CHaiDNN on ZCU104 FPGA board, as a function of the number of active ROs.

B. Attack on CHaiDNN running ImageNet

1) Image-Level attack: We first examine the accuracy of the model used to conduct the attack training by first constructing a local model of the victim device. Then we generate all possible attack combinations for the first six layers of the model, assigning either 2^9 or 2^{12} ROs at random to each layer. We run 20 randomly selected images through each attack combination and record the predicted target class. We then use the same attack parameters to run the attack on the FPGA hardware and compare the classification output. If they match, the attack parameters successfully misclassified the input into the targeted class. Figure 6 shows the attack success rate as average for each image and overall average. We can see that the attack succeeds with high probability averaging at 79.3%, with low variability between images.

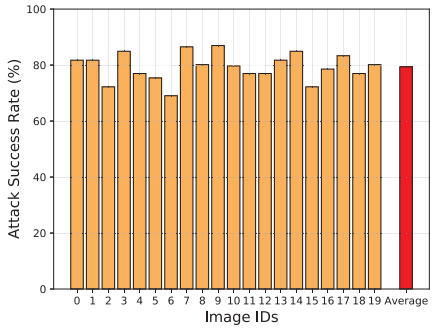


Fig. 6: Image-level attack success rate on the ImageNet dataset.

To estimate how many potential target classes could be reliably reached for a given target image, we tested 20 different images from the same class with 500 different attack combinations. We randomly chose the layers to attack and

apply RO intensities of either 2^9 or 2^{12}. Figure 7 shows the results. Each input image can be misclassified to anywhere between 74 to 98 distinct target classes, with an average of 80. This shows that the attack, which injects two levels of noise at layer granularity, introduces sufficient entropy into the target model to enable misclassification into a large number of target classes.

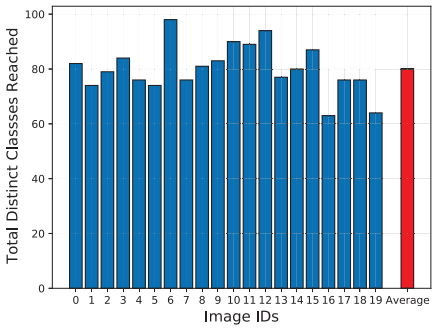


Fig. 7: Total number of classes reached during targeted attacks on randomly selected images from the ImageNet dataset.

We also examine, for several input classes, how many distinct target classes the attack can successfully reach. In Figure 8 we show two distinct RO intensities used for this experiment and also a dynamic combination of both RO intensities. The average number of target classes reached are 121, 101, and 190 for 2^9, 2^{12}, and dynamic combination of 2^9, 2^{12} RO intensities respectively. This again shows that the attack can generate a large number of targets, consistently across input classes.

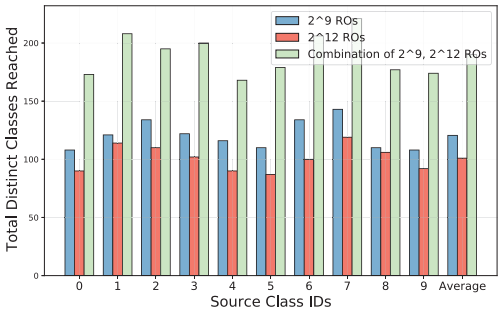


Fig. 8: Number of target classes reached for different input classes in the ImageNet dataset for multiple RO intensities.

2) Class-Level Attack: We analyze the probability of success for a class-level attack, which uses a single set of attack parameters to attempt to misclassify all images from that class into a *target* class. Naturally, this attack is much more challenging than targeting individual images. In this experiment, we analyze a set of 100 images from the same class, exposing them to varying levels of noise (RO intensities of 2^9 or 2^{12}) on the FPGA hardware. Noise is injected in all possible combinations of the first six convolution layers. Figure 9 shows the probability of success for this attack on the Y-axis for all the target classes reached during this attack. The results suggest that it is possible to misclassify a complete set of images to a single target class, using a single combination of attack parameters – achieving a success probability as high as 0.84. While this is achievable with a high probability for only a handful of input classes, this attack is much more powerful than the image-level attack because it allows reliable misclassification of entire classes, without training on individual images.

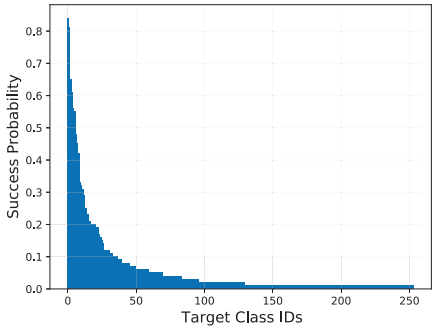


Fig. 9: Probability of success of the class-level attack for select target classes from the ImageNet dataset.

C. Attack Accuracy for MNIST and CIFAR-10

1) Model Accuracy: To conduct experiments with MNIST and CIFAR-10, which CHaiDNN does not natively support, we developed a cycle-accurate software simulation of CHaiDNN experiencing noise, using TensorFlow on an Intel Core-i7 CPU. In CHaiDNN, 256 compute units work simultaneously, especially during convolution operations, powered by specific kernels or filters alongside a co-located ring oscillator circuit. Our software model mimics this, incorporating voltage noise through error profiles based on random compute units and bit disturbances. This simulation was validated using the VGG-16 model. Figure 10 shows the simulator's classification accuracy closely matching real hardware outcomes under equivalent RO intensities.

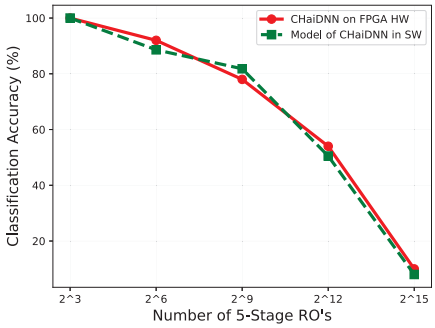


Fig. 10: Classification accuracy of CHaiDNN accelerator in FPGA under voltage noise vs. software model.

2) Image-Level Attack: Figure 11 shows the main results for the image-level attack for the MNIST (using AlexNet model) and CIFAR-10 (using VGG-16) datasets respectively. We test a total of $10K$ input samples for each dataset. We report the average success rate for a selected target class misclassified to *every* other class in that dataset. We observe that with 2^9 RO intensity, we can cover on average 90.04% of target classes from a source class for the MNIST dataset and 80.29% for CIFAR-10. When adding an RO intensity of 2^{12} the target coverage increases to 99.5% for MNIST and 85.44% for CIFAR-10. This shows that for datasets that have relatively few classes, the attack can cover virtually all target classes with a very high probability.

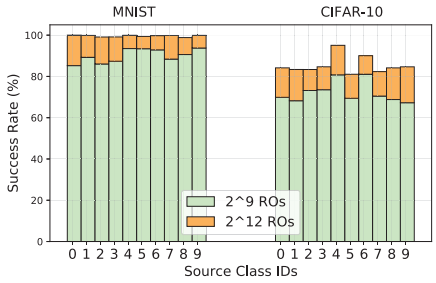
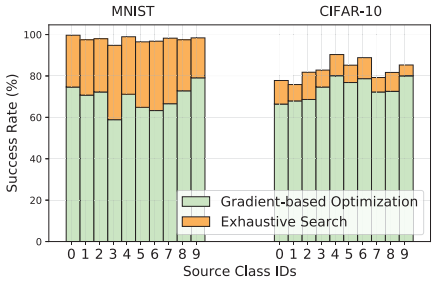


Fig. 11: Image-level attack success rate for the MNIST and CIFAR-10 datasets, for multiple RO intensities.

3) Class-Level Attack: For the class-level attack where we identify a single set of attack parameters that can misclassify an entire class of input images into a target class. Figure 12 shows the accuracy for the MNIST and CIFAR-10 datasets respectively. We again incorporate both voltage noise levels and report the average success rate for a selected source class misclassified to *all* other classes in that dataset. Using the exhaustive search mechanism we reach 97.63% and 82.89% target classes on average for each of the two datasets respectively, across all source classes. The gradient-based attack training method, which converges much faster, also has a relatively high success rate with 70.42% and 73.81% on average for the two datasets respectively, across all source classes. This is especially important because we find one combination that applies to all/most images of a particular class, making the attack more general.

V. Conclusion and Future Work

This paper demonstrates a new voltage-noise-based targeted attack on an ML accelerator in a multi-tenant FPGA. We showcase and evaluate the attack using real hardware and find that for ImageNet we can reliably misclassify an average image into any of $190+$ classes, with a probability exceeding 80%. For MNIST and CIFAR-10 our attack can misclassify 99.5% and 85.4% of the input images into any target class. As a future work, we plan on optimizing the class-level attack to increase the reach to more target classes. We will investigate increasing RO granularity as well as potentially combining our attack with conventional bit-flip methods.



s

Fig. 12: Generalized class-level attack success rates for MNIST and CIFAR-10 datasets.

References

[1] "Zynq UltraScale+ MPSoC ZCU104 Evaluation Kit." [Online]. Available: https://www.xilinx.com/products/boards-and-kits/zcu104.html

[2] A. Boutros, M. Hall, N. Papernot, and V. Betz, "Neighbors From Hell: Voltage Attacks Against Deep Learning Accelerators on Multi-Tenant FPGAs," in *2020 International Conference on Field-Programmable Technology (ICFPT)*, 2020, pp. 103–111.

[3] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[5] G. Dessouky, A.-R. Sadeghi, and S. Zeitouni, "SoK: Secure FPGA Multi-Tenancy in the Cloud: Challenges and Opportunities," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021.

[6] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "FPGAhammer: Remote Voltage Fault Attacks on Shared FPGAs, suitable for DFA on AES," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2018, Issue 3, pp. 44–68, 2018.

[7] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," University of Toronto, Tech. Rep., 05 2012.

[8] Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey E., "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. Red Hook, NY, USA: Curran Associates Inc., 2012, p. 1097–1105.

[9] Y. LeCun and C. Cortes, "The MNIST database of handwritten digits," 2005.

[10] Y. Luo, C. Gongye, Y. Fei, and X. Xu, "DeepStrike: Remotely-Guided Fault Injection Attacks on DNN Accelerator in Cloud-FPGA," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021.

[11] A. S. Rakin, Z. He, and D. Fan, "Bit-Flip Attack: Crushing Neural Network With Progressive Bit Search," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1211–1220.

[12] A. S. Rakin, Y. Luo, X. Xu, and D. Fan, "Deep-Dup: An Adversarial Weight Duplication Attack Framework to Crush Deep Neural Network in Multi-Tenant FPGA," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 1919–1936.

[13] B. Salami, E. B. Onural, I. E. Yuksel, F. Koc, O. Ergin, A. Cristal, O. Unsal, H. Sarbazi-Azad, and O. Mutlu, "An Experimental Study of Reduced-Voltage Operation in Modern FPGAs for Neural Network Acceleration," *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 138–149, 2020.

[14] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9*.

[15] Xilinx, "CHaiDNN," https://github.com/Xilinx/CHaiDNN.

[16] F. Yao, A. S. Rakin, and D. Fan, "DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips," in *Proceedings of the 29th USENIX Conference on Security Symposium*, ser. SEC'20. USA: USENIX Association, 2020.

[17] Y. Zhang, R. Yasaei, H. Chen, Z. Li, and M. A. A. Faruque, "Stealing Neural Network Structure Through Remote FPGA Side-Channel Analysis," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4377–4388, 2021.