Machine Learning and Hardware security: Challenges and Opportunities

F. Regazzoni, S. Bhasin, Amir Ali Pour, Ihab Alshaer, F. Aydin, A. Aysu, Vincent Beroulle, Giorgio Di Natale, Paul Franzon, David Hely, et al.

Machine Learning and Hardware security: Challenges and Opportunities

–Invited Talk–

Francesco Regazzoni[1], Shivam Bhasin[2], Amir Ali Pour[3], Ihab Alshaer[5], Furkan Aydin[4], Aydin Aysu[4], Vincent Beroulle[3], Giorgio Di Natale[5], Paul Franzon[4], David Hely[3], Naofumi Homma[6], Akira Ito[6], Dirmanto Jap[2], Priyank Kashyap[4], Ilia Polian[7], Seetal Potluri[4], Rei Ueno[6], Elena-Ioana Vatajelu[5], Ville Yli-Mäyry[6]

[1]University of Amsterdam and ALaRI - USI; [2]Nanyang Technological University; [3]Grenoble INP; [4]North Carolina State University; [5]TIMA/CNRS/Univ. Grenoble Alpes; [6]Tohoku University/CREST; [7]University of Stuttgart.

ABSTRACT

Machine learning techniques have significantly changed our lives. They helped improving our everyday routines, but they also demonstrated to be an extremely helpful tool for more advanced and complex applications. However, the implications of hardware security problems under a massive diffusion of machine learning techniques are still to be completely understood. This paper first highlights novel applications of machine learning for hardware security, such as evaluation of post quantum cryptography hardware and extraction of physically unclonable functions from neural networks. Later, practical model extraction attack based on electromagnetic side-channel measurements are demonstrated followed by a discussion of strategies to protect proprietary models by watermarking them.

CCS CONCEPTS

• **Security and privacy → Side-channel analysis and countermeasures**; • **Computing methodologies → Neural networks**.

KEYWORDS

machine learning, hardware security

ACM Reference Format:

Francesco Regazzoni[1], Shivam Bhasin[2], Amir Ali Pour[3], Ihab Alshaer[5], Furkan Aydin[4], Aydin Aysu[4], Vincent Beroulle[3], Giorgio Di Natale[5], Paul Franzon[4], David Hely[3], Naofumi Homma[6], Akira Ito[6], Dirmanto Jap[2], Priyank Kashyap[4], Ilia Polian[7], Seetal Potluri[4], Rei Ueno[6], Elena-Ioana Vatajelu[5], Ville Yli-Mäyry[6] . 2020. Machine Learning and Hardware security: Challenges and Opportunities: –Invited Talk–. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '20), November 2–5, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3400302.3416260

ICCAD '20, November 2–5, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8026-3/20/11…$15.00
https://doi.org/10.1145/3400302.3416260

1 INTRODUCTION

The use of machine learning (ML) and its ability to solve complex problems has improved many applications in domains such as image classification, natural language processing, computer vision, and bio-informatics. These developments have stimulated a range of advancements and triggered new paradigms. One example is "edge ML", which focuses on deployment and use of optimized ML models on edge devices like Internet of things (IoT). The edge based deployment of ML models generates fresh opportunities to explore, but also raises new security concerns which must be systematically investigated. The focus of this paper is to highlight new opportunities and challenges when considering ML and hardware security.

ML has long been used in network and system security for detecting intrusions, anomalies and malware [25]. In the context of hardware security, the major applications of ML are its use for security evaluation of cipher implementations against side-channel attacks [15] and model-building attacks against PUFs [16]. With recent developments in machine learning algorithms new applications have come to light which include, but not limited to, evaluation of protected implementation against side-channel attacks, finding new fault attacks [27], security aware design flow [18] etc. However, with edge based deployment of ML models recent works have shown new security threats like model extraction based on side-channel [5, 10], label misclassification by fault injection [7] and so on. In this paper, we take a deeper look into the dual interplay between machine learning and hardware security.

The first part of the paper highlights novel applications of machine learning for hardware security. The first application caters to the side-channel security evaluation of a high-profile post quantum cryptography algorithm, currently under standardization by NIST [20]. The algorithms are novel and their hardware implementations lack security analysis. The use of deep learning for thorough security evaluation of these algorithms' implementations can help gain confidence into their security. We show that a deep-learning based side-channel attack can extract the secret key from side-channel leakage while classical attacks cannot. Moreover, we identify that ML generalization can negate the effect of potential countermeasures. Further, with deployment of ML in IoT there arises a need for strong authentication mechanisms, preferably with a hardware root of trust. While standard hardware like memory or delay chains possess PUF-like properties, the complex nature of neural network can also be used to extract such properties from neural network structure. Identifying intrinsic features of neural

networks can enable several new possibilities like binding neural network to a particular device for preventing illegal cloning.

The second part of the paper investigates threats to ML arising from deployment on edge hardware. The ML models deployed on edge devices are generally trained using expensive resources and training data. The optimized models, which are proprietary in nature, are then deployed on edge devices for efficient inference task. Successful attacks allowing complete recovery of these proprietary models have been shown previously using side-channel attacks [5, 10]. Moreover, leakage of the model can also leak information on sensitive training data. We demonstrate practical EM side-channel measurement based model recovery attack on Binarized Neural networks (BNN) running on FPGA. The attacks are shown to be possible even for optimized hardware generated using high-level synthesis flow [21] and can target structures like the convolution layers for weight extraction. Further, we discuss the potential of watermarking in protecting these models from IP theft. In particular, we lay focus on techniques developed specifically to watermark ML hardware allowing deep embedding of the model fingerprinting capabilities at design time.

The rest of the paper is organized as follows. Section 2 proposes a novel ML based method for evaluation of post quantum cryptographic processors against side-channel. Section 3 investigates PUF properties in neural network as compared to classical PUFs. Section 4 reports practical model recovery attack on BNN models deployed on FPGA. Section 5 provides a study of potential countermeasures against such attacks. Finally, conclusions are drawn in Section 6.

2 DEEP LEARNING ATTACKS ON POST-QUANTUM HARDWARE

This section presents a side-channel security evaluation using ML and compares it to classical techniques. Specifically, we are evaluating FrodoKEM, a post-quantum key encapsulation mechanism [20]. Such algorithms are envisioned for use in HTTP/TLS protocols and, unlike existing solutions, provide theoretical security against quantum computer attacks. The United States National Institute of Standards and Technology (NIST) is currently working towards standardizing post-quantum algorithms for future use, and Frodo is a remaining candidate in the last round of this process [23], making it a high-profile target. But the hardware implementations of these algorithms do leak information through side-channels. In the scope of this work, we are interested in the power consumption-based side-channel leakage.

Although ML-based side-channel attacks exist, they have primarily focused on the leakage of AES, which is a symmetric-key encryption algorithm [13, 14, 17, 24]. The typical goal in such works is to reduce the number of tests/measurements needed to extract the *fixed* secret key, i.e., the system is already broken but the effort needed will be reduced with a better attack. Since we are evaluating a key encapsulation mechanism, the stark contrast in our scenario is that the key changes after each execution. This limits the adversary to a single power measurement. Hence, an improvement in side-channel analysis can result in breaking a system that is otherwise considered secure.

Prior side-channel attacks on FrodoKEM (or its predecessor) have adapted the classical horizontal Differential Power Analysis

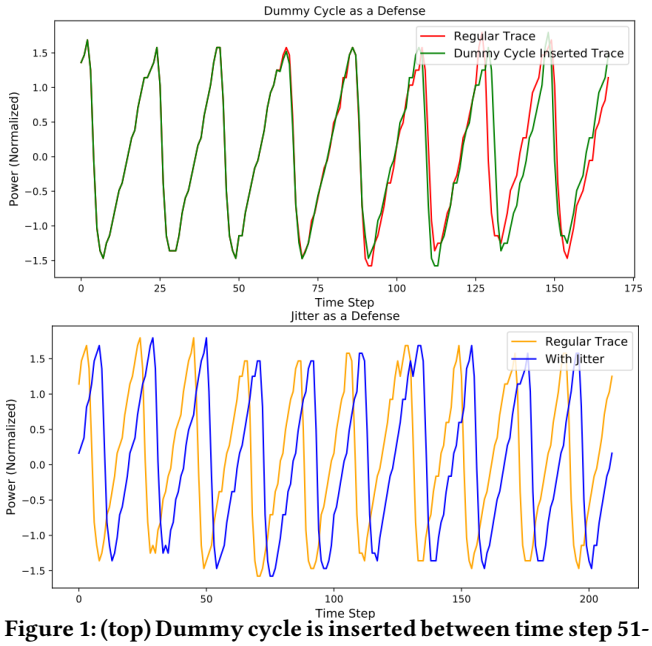

Figure 1: (top) Dummy cycle is inserted between time step 51-75 and subsequent sub-traces are shifted (bottom) Random jitter is added to the regular device operation.

(DPA) [4], which then got extended with the template attacks [6]. Most recently, a machine learning framework has quantified the advantages over classical techniques [3]. But all these efforts assume that there is no countermeasure against physical side-channel analysis. This has been a reasonable assumption since the single-trace challenge of the attacks would motivate defenders to skip countermeasures to save cost. However, given these attacks, countermeasures will arguably appear soon. And the first countermeasures will be simpler ones, that again, aims at minimizing cost. It is, therefore, critical to evaluate if side-channel leakage would still be captured with ML-based attacks in the presence of such countermeasures.

To that end, we emulate and test two defenses: dummy state insertion and clock jitter-based hiding. The effects of these two defenses are shown in Figure 1. While the dummy state insertion randomly injects an averaged trace into the computations, jitter-based hiding causes shifts in the temporal domain. Both attacks desynchronize the alignment of sub-traces (i.e., computations) within the single power measurement. Therefore, they reduce the effectiveness of classical attacks. The premise of deep learning based attacks is to understand the transformations introduced by the countermeasures and to automatically apply signal processing or alignment needed to negate those effects.

We apply the convolutional neural network based classifier defined as in [3] and compare it with horizontal DPA and template attacks. All attacks target a row of matrix multiplications in hardware that computes 16 multiplications in parallel. The results, summarized in Figure 2, indeed validate the premise of deep learning attacks. While the success rate of the horizontal DPA and template attacks respectively drop to 8% and 9%, the deep-learning attack still achieves a 100% success rate. The effort in ML profiling, however, is increased—the profiling has to be performed with the countermeasures switched on to learn their effect. But surprisingly, the number of measurements needed to train a successful attack remains the

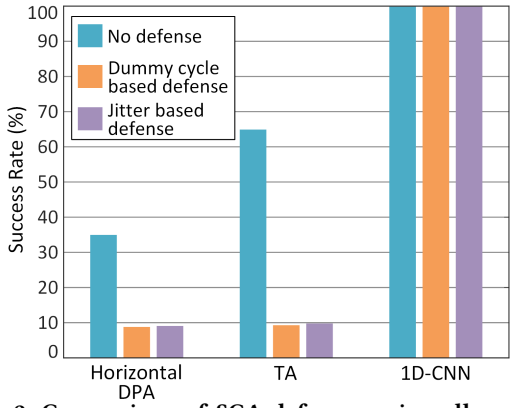

Figure 2: Comparison of SCA defenses using all available sub-traces (i.e., 84 sub-traces) of Frodo.

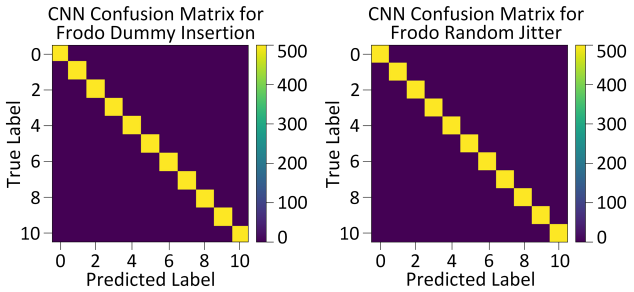

Figure 3: Confusion matrix for (left) dummy cycle insertion and (right) random jitter insertion with 84 sub-traces.

same. The confusion matrix in Figure 3 presents the details of the ML attack results. As is clear from the confusion matrix none of the sub-keys are mispredicted, which indicates that the trained CNN model can learn both the countermeasures effectively. The loss of performance of traditional approaches in the presence of the countermeasures makes ML-based side channel attacks an attractive prospect for an adversary and urges designers to explore more advanced protections.

3 PHYSICALLY UNCLONABLE FUNCTIONS EXTRACTED FROM EMBEDDED NEURAL NETWORKS

Neural Networks (NNs) have emerged as a major computing paradigm. They are composed of a large number of interconnected processing elements (neurons) working together to solve specific problems, and are configured for a specific application through learning, which is performed by adjusting the strength of synaptic connections between neurons. NNs tend to become ubiquitous in our lives today. They are (or will be in the near future) present in many applications, including IoT and smart sensors. Since these applications are built on strongly interconnected objects, security and trust have to be guaranteed. To this end, an IoT object should have strict rules to access the network, such as authentication protocols. Physically Unclonable Functions (PUFs) have emerged as cryptographic primitives used to implement low-cost device authentication and secure secret key generation [28]. PUFs are built based on an input-output generation process. The inputs are playing the role of challenges to the PUFs and the outputs are their responses to these challenges. According to the generated number of Challenge-Response Pairs (CRPs), a PUF can be classified as a weak PUF, when a single device can generate a small number of CRPs, or a strong PUF when a single device can generate a large number of CRPs such as to prevent an exhaustive analysis. While weak PUFs are used for generating fingerprints and secret keys, strong PUFs are used for device authentication.

Existing strong PUF solutions use dedicated hardware and they are less mature than the well-established SRAM-based weak PUFs. Several solutions exist to transform a weak PUF into a strong PUF ([2], [31]). They exploit the embedded SRAM to generate the weak PUF and the existing hardware/software components (e.g., an AES crypto core) to expand the number of CRPs while guaranteeing the properties of a PUF (i.e., unclonability, uniqueness, uniformity, bit-aliasing and reliability). Following the same idea, in this work we explore the possibility of re-using an existing NN to generate a strong PUF. We assume that the NN runs on a system with an embedded SRAM. The idea of this work is to use the NN in two regimes: the computation, where the synaptic weights are loaded in the SRAM, and the strong PUF generation, where the SRAM is powered-on without performing any write operation, thus reaching its random state (this translates to random weight values) and the network's inputs are used as challenges, while its outputs as response. We define the latter behavior as NNPUF.

The NNPUF solution could be in principle implemented on any NN, but as a proof of concept we focused on the Multilayer Perceptron (MLP). It is a feed-forward NN built of several layers of fully connected neurons; each neuron is behaving like a perceptron. For this case study, several network topologies have been considered, with one or two hidden layers of different sizes. All neurons in hidden layers use the sigmoid as activation function, while the neurons in the output layer use softmax. Three precisions have been considered for the encoding of the synaptic weights: 8 bits, 16 bits and 32 bits. The output of the NN is translated into a single-bit digital response by the following convention: assuming n outputs of the NN, if the maximum output corresponds to the first $n/2$ outputs, then the PUF response is considered to be logic 1, otherwise 0. In order to assess the quality of the proposed solution, we have simulated the considered NN configurations by randomly generating the values of the synaptic weights. The assessment has been performed by calculating the well-known metrics for PUFs: Uniformity (i.e., the 0s and 1s in a binary response must be uniformly distributed); Uniqueness (i.e., each generated response has to be unique over the set of responses); Bit-Aliasing (i.e., the 0s and 1s for the same challenge in multiple devices must be uniformly distributed); Reliability (i.e., the responses should be stable in time).

Experiments have been conducted on 1000 devices by simulating 10K CRPs, for various PUF topologies. One topology is expressed as: [IL HL OL], where IL is the number of NN inputs, HL is the number of neurons in the hidden layers, and OL is the number of outputs. Table 1 shows the values of uniformity, bit aliasing and reliability we obtained for three topologies and three precisions of the synaptic weights encoding. The uniqueness (not included in the table) was 100% for all experiments. To evaluate the reliability of the NNPUF, we assumed instability of the SRAM memory (i.e., modifications in

Table 1: Quality evaluation of the NNPUF.

NNPUF	nBits	Unif.	BA	Rel.
[100 80 2]	8	97.6%	97.5%	45%-100%
	16	98.1%	97.9%	
	32	97.8%	97.7%	
[100 80 40 2]	8	98.8%	98.2%	40%-100%
	16	99.1%	98.7%	
	32	98.5%	98.3%	
[784 100 10]	8	98.1%	97.9%	43%-100%
	16	98.3%	98.1%	
	32	97.8%	97.5%	

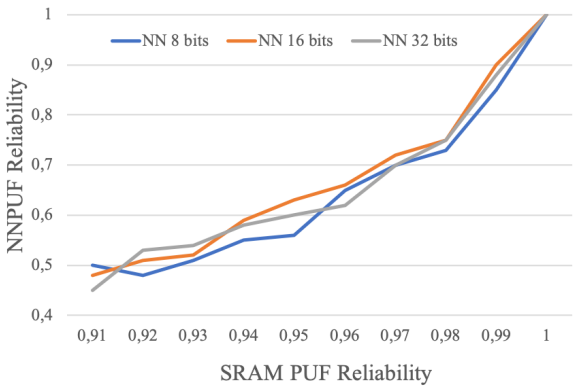

Figure 4: NNPUF Reliability vs SRAM PUF Reliability

the values of the synaptic weights). We applied the same challenges to the NNPUF and we repeated the experiments 1000 times with slightly different SRAM content. Figure 4 shows the dependency of the NNPUF reliability on the SRAM PUF reliability. It can be seen that zero-bit error rate SRAM PUF (as for instance the one proposed in [30]) is required for reliable NNPUF. This shortcoming is observed on most strong PUFs built on weak PUFs.

4 MODEL EXTRACTION ATTACK ON PRACTICAL BNN HARDWARE USING EM SIDE-CHANNEL INFORMATION

This section presents a method for electromagnetic (EM) side-channel analysis to recover the layer structure and parameters of practical 'ized Neural Network (BNN) hardware generated with GUINNESS [21], and synthesized with commercial high-level synthesis tool Vivado HLS. The hardware is implemented on a Xilinx Zynq chip mounted on Digilent Zedboard. We show how the layers can be identified from a single EM trace measured during the network's evaluation, and we demonstrate how an attacker may use side-channel attacks to recover the weights used in the layers.

Attacks against BNNs have been reported in recent years [10, 32]. Dubey et al. [10] demonstrated an attack on the adder-tree function of the authors' custom BNN hardware. In [32], the authors recovered the architecture of a targeted BNN through analysis of EM emanations from the data bus of the BNN accelerator and built substitute models using this information. Such attacks have

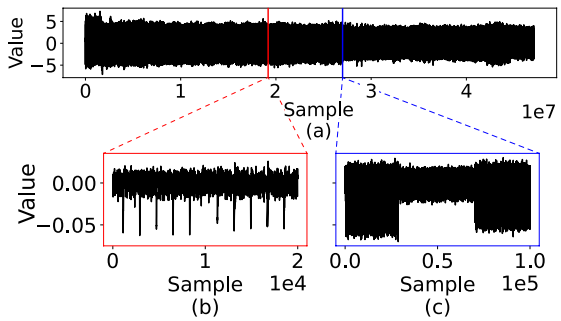

Figure 5: EM measurement of the target BNN evaluation (a), with part of the measurement identifying the end of a layer (b), and part of measurement identifying a pooling layer (c).

encouraged recent works to look into neural networks structure with side-channel countermeasures [11].

Our contributions here are as follows. Firstly, we target a practical BNN hardware generated through a common high-level synthesis tool, and show that the overall structure of BNN can be identified by the corresponding EM emanations. In addition, we target the convolution layers, and demonstrate that the weight values convolved in the layers are revealed by EM side-channel attacks with actual experiments, where a Langer-EMV near-field probe and a Keysight DSOS404A oscilloscope are used to measure EM emanations directly from the targeted Zynq chip.

4.1 Architecture extraction

The architecture of the targeted BNN can be deduced from a visual inspection of the EM measurement of the FPGA chip during an evaluation of the network. Figure 5 (a) shows the overall trace of the BNN hardware evaluation we target. Example of a boundary between layers that can be found with visual analysis is shown with a red line. The portion is magnified in Figure 5 (b). Between layers, there is a single clock cycle state during which no data is processed. This causes the amplitude of the EM emanations to drop significantly during the clock cycle, and allows the attacker to easily deduce the timing between layers. Similarly, max-pooling can be optimized to bitwise OR gates in hardware implementations of BNNs, which greatly reduces the EM emanations during the processing of the layer. Figure 5 (c) shows an example of a pooling layer between convolution layers, and how it can be identified in the EM measurement using this knowledge. These techniques allow the attacker to recover the BNN structure. Figure 6 shows the result of a fixed vs. random t-test using datasets consisting of single random pixel images, all-zero images, and their corresponding EM traces. Using the peaks from the t-test, the attacker can identify the Points-of-Interest (POI) in the processing, which contain the weights the attacker wishes to recover.

4.2 Weight extraction

We employ Correlation EM Analysis (CEMA) to perform the weight recovery. Convolution functions of BNNs generally utilize a temporary register where the result of adding intermediate values is

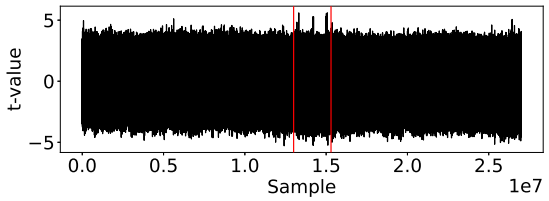

Figure 6: Result of fixed vs. random t-test. Peaks show the timings where the pixel value is convolved with the secret weights. The area of interest is shown with red lines.

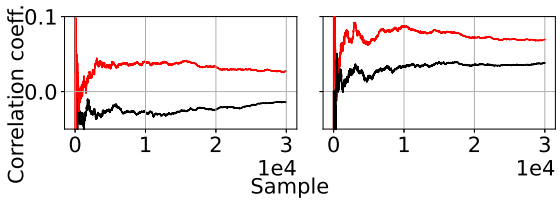

Figure 7: Measurements to Disclosure (MTD) results for two consecutive weights. The correct and incorrect weight guesses are shown with red lines and black lines .

stored after it is updated with the input with the secret weight applied to it. Measurements to Disclosure (MTD) result of the CEMA in Figure 7 shows that we were successful in retrieving weights applied to the inputs of the first layer. This result implies that an attack using the value change in the temporary register of the convolution algorithm can be exploited to recover the secret weights, and that the first recovered weight can be used to calculate the subsequent value of the temporary register, which allows to recover the next weight, and so on.

5 PROTECTIONS AGAINST ATTACKS ON MACHINE LEARNING HARDWARE

While machine-learning hardware is subject to all the threats faced by security-critical circuits in general, there are several threats specific to ML. Most notably, companies are investing significant efforts to design and train NNs and other ML implementations, and want to protect their architecture and weights against extraction attacks adversaries with physical access to devices (see previous section for an example). Special watermarking and fingerprinting techniques have been devised for ML hardware, rather than for general circuits; these techniques and attacks targeting them published so far are reviewed in this section.

Watermarking aims at proving the authorship. In the context of artificial intelligence protection, watermarking follows two main approaches. In the first, the protection technique is embedded into the model modifying directly the weights within the model. Since this alters directly the process of feature extraction, we call this method *feature-based*. The second approach embeds the watermark by training selected inputs to be classified using specific labels, similar to adversarial training. Since it is activated by selected inputs, we call this watermarking method *trigger-based*.

Uchida et al. [29] were the first proposing to use digital watermarking for protecting deep NN (DNN) models. To embed a watermark (a binary string),they introduce a bias into the distribution of weights. The verification (watermark read-out) is done projecting the means of the weight of the internal layers. The main limitation of this approach is the required access to internal layers to carry out the verification ("white box" verification). Adi et al. [1] proposed to embed the watermark using over-parametrization. Their watermark is embedded during training, and can be verified without access to internal layers of the model (thus called "black box") but requires the involvement of a third party to guarantee the security of the verification process.

Zhang et al. [33] proposed to train the network to produce specific predictions when selected patterns are used as inputs. The verification is carried out by presenting the patterns used for the training the watermark to the model and to compare the answers with the expected ones. A *passport-based* approach was presented by Fan et al. [12]. To make it dependent from the passport, the passport is included during the design and training of the network. Once the passport is used: the verification step occurs by considering two aspects, the behaviour of the DNN when selected inputs are applied, and the performance of the model. Observing unexpected performance is an indication that a wrong passport is used. This approach requires access to the internal state, but the authors propose to use it in combination with black box approaches.

Merrer et al. [19] proposed watermarking based on *adversarial attacks:* add small perturbation to several original samples with the goal of classifying them incorrectly, and use the model with corrected labels as watermark. Ownership verification can be carried out by measuring the gap between the classification of adversarial examples. Complete frameworks for watermarking have been proposed by by Rouhani et al. [26], where the watermark is embedded in the probability density distribution of the activation sets for different layers of the neural network, and by Chen et al. [8]—here, the (binary) watermark is embedded using an encoding schema that splits all the classes belonging to a specific task into two groups.

A strategic adversary who knows about the existence of protections may desire to inactivate or remove them. Among the attacks against watermarking techniques are *watermark removal*([26], [9]), *query modification* [22] that makes ineffective the verification process by altering the used query and *ambiguity attacks* [12], that aim at creating doubts about the ownership of an IP by embedding several forged watermarks into the DNN.

When weight and architecture extraction is carried out using power analysis, it can be counteracted with approaches similar to the ones used to mitigate key recovery power analysis attacks against cryptographic implementations. One can distinguish between *hiding* (removing the correlation between the computed data and the power consumption of the device) and *masking* (breaking the link between the data processed by the algorithm and the data processed by the device). Based on these general principles, countermeasures have to be adapted to machine learning models and algorithms. To date, this topic has been addressed only by a limited amount of works, including a fully masked neural network [11] and a combination of hiding and masking [10] for side-channel mitigation. We believe that this is a promising direction for future research.

6 CONCLUSIONS

The relationship between machine learning and hardware security is threefold: ML can help an adversary to attack a (hardware-based) system; it can be deployed to defend the same system against adversaries; and finally, the ML circuitry itself can be the target of attacks. In this paper, we brought several examples of ML circuitry on both the "good" and the "evil" side in the ongoing cat-and-mouse game of security. The intersection between ML and hardware security is an arena of ongoing research, and we expect that many new and interesting results will be discovered by experts from both disciplines working together in the near future.

ACKNOWLEDGEMENT

This work was performed in the Cooperative Research Project of the Research Institute of Electrical Communication, Tohoku University with Nanyang Technological University. This research is supported in part by the NSF under the Grants No. CNS 16-244770 (Center for Advanced Electronics through Machine Learning), by JST CREST Grant No. JPMJCR19K5, Japan, and by European Union's Horizon 2020 research and innovation programme CPSoSaware (grant agreement No 871738). The authors thank Paolo Palmieri and Dado Smailbegovic for the fruitful discussion about protection of machine learning algorithms.

REFERENCES

[1] Yossi Adi, Carsten Baum, Moustapha Cissé, Benny Pinkas, and Joseph Keshet. 2018. Turning Your Weakness Into a Strength: Watermarking Deep Neural Networks by Backdooring. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, William Enck and Adrienne Porter Felt (Eds.). USENIX Association, 1615–1631. https://www.usenix.org/conference/usenixsecurity18/presentation/adi

[2] Massimo Alioto. 2017. *Enabling the Internet of Things*. Springer.

[3] F. Aydin, P. Kashyap, S. Potluri, P. Franzon, and A. Aysu. 2020. DeePar-SCA: Breaking Parallel Architectures of Lattice Cryptography via Learning Based Side-Channel Attacks. In *In International Conference on Embedded Computer Systems: Architectures, Modelling Simulation (SAMOS)*. https://research.ece.ncsu.edu/aaysu/wp-content/uploads/SAMOS_2020_Camera_Ready_Paper.pdf

[4] A. Aysu, Y. Tobah, M. Tiwari, A. Gerstlauer, and M. Orshansky. 2018. Horizontal side-channel vulnerabilities of post-quantum key exchange protocols. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 81–88.

[5] Lejla Batina, Shivam Bhasin, Dirmanto Jap, and Stjepan Picek. 2019. {CSI}{NN}: Reverse Engineering of Neural Network Architectures Through Electromagnetic Side Channel. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 515–532.

[6] Joppe W. Bos, Simon Friedberger, Marco Martinoli, Elisabeth Oswald, and Martijn Stam. 2019. Assessing the Feasibility of Single Trace Power Analysis of Frodo. In *Selected Areas in Cryptography – SAC 2018*, Carlos Cid and Michael J. Jacobson Jr. (Eds.). Springer International Publishing, Cham, 216–234.

[7] Jakub Breier, Xiaolu Hou, Dirmanto Jap, Lei Ma, Shivam Bhasin, and Yang Liu. 2018. Deeplaser: Practical fault attack on deep neural networks. *arXiv preprint arXiv:1806.05859* (2018).

[8] Huili Chen, Bita Darvish Rouhani, and Farinaz Koushanfar. 2019. BlackMarks: Blackbox Multibit Watermarking for Deep Neural Networks. *CoRR* abs/1904.00344 (2019). arXiv:1904.00344 http://arxiv.org/abs/1904.00344

[9] Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song. 2019. REFIT: a Unified Watermark Removal Framework for Deep Learning Systems with Limited Data. *CoRR* abs/1911.07205 (2019). arXiv:1911.07205 http://arxiv.org/abs/1911.07205

[10] Anuj Dubey, Rosario Cammarota, and Aydin Aysu. 2019. MaskedNet: A Pathway for Secure Inference against Power Side-Channel Attacks. *arXiv preprint arXiv:1910.13063* (2019).

[11] Anuj Dubey, Rosario Cammarota, and Aydin Aysu. 2020. BoMaNet: Boolean Masking of an Entire Neural Network. *arXiv preprint arXiv:2006.09532* (2020).

[12] Lixin Fan, KamWoh Ng, and Chee Seng Chan. 2019. Rethinking Deep Neural Network Ownership Verification: Embedding Passports to Defeat Ambiguity Attacks. In *NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*. 4716–4725.

[13] R. Gilmore, N. Hanley, and M. O'Neill. 2015. Neural network based attack on a masked implementation of AES. In *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 106–111.

[14] Benjamin Hettwer, Tobias Horn, Stefan Gehrer, and Tim Güneysu. 2020. Encoding Power Traces as Images for Efficient Side-Channel Analysis. arXiv:2004.11015 [cs.CR]

[15] Gabriel Hospodar, Benedikt Gierlichs, Elke De Mulder, Ingrid Verbauwhede, and Joos Vandewalle. 2011. Machine learning in side-channel analysis: a first study. *Journal of Cryptographic Engineering* 1, 4 (2011), 293.

[16] Gabriel Hospodar, Roel Maes, and Ingrid Verbauwhede. 2012. Machine learning attacks on 65nm Arbiter PUFs: Accurate modeling poses strict bounds on usability. In *2012 IEEE international workshop on Information forensics and security (WIFS)*. IEEE, 37–42.

[17] Jaehun Kim, Stjepan Picek, Annelie Heuser, Shivam Bhasin, and Alan Hanjalic. 2019. Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled Side-channel Analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019, 3 (May 2019), 148–179. https://doi.org/10.13154/tches.v2019.i3.148-179

[18] Serge Leef. 2019. Automatic Implementation of Secure Silicon.. In *ACM Great Lakes Symposium on VLSI*. 3.

[19] Erwan Le Merrer, Patrick Pérez, and Gilles Trédan. 2020. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications* 32, 13 (2020), 9233–9244. https://doi.org/10.1007/s00521-019-04434-z

[20] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, et al. 2017. FrodoKEM. *Technical report, National Institute of Standards and Technology* (2017).

[21] Hiroki Nakahara, Haruyoshi Yonekawa, Tomoya Fujii, Masayuki Shimoda, and Shimpei Sato. 2019. GUINNESS: A GUI based binarized deep neural network framework for software programmers. *IEICE TRANSACTIONS on Information and Systems* 102, 5 (2019), 1003–1011.

[22] Ryota Namba and Jun Sakuma. 2019. Robust Watermarking of Neural Network with Exponential Weighting. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, AsiaCCS 2019, Auckland, New Zealand, July 09-12, 2019*, Steven D. Galbraith, Giovanni Russello, Willy Susilo, Dieter Gollmann, Engin Kirda, and Zhenkai Liang (Eds.). ACM, 228–240. https://doi.org/10.1145/3321705.3329808

[23] National Institute of Science and Technology, Computer Security Resource Center. 2020. Post-Quantum Cryptography PQC Round 3 Submissions. https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions.

[24] Emmanuel Prouff, Remi Strullu, Ryad Benadjila, Eleonora Cagli, and Cecile Dumas. 2018. Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database. Cryptology ePrint Archive, Report 2018/053. https://eprint.iacr.org/2018/053.

[25] Konrad Rieck, Philipp Trinius, Carsten Willems, and Thorsten Holz. 2011. Automatic analysis of malware behavior using machine learning. *Journal of Computer Security* 19, 4 (2011), 639–668.

[26] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. 2019. DeepSigns: An End-to-End Watermarking Framework for Ownership Protection of Deep Neural Networks. In *ASPLOS 2019, Providence, RI, USA, April 13-17, 2019*, Iris Bahar, Maurice Herlihy, Emmett Witchel, and Alvin R. Lebeck (Eds.). ACM, 485–497. https://doi.org/10.1145/3297858.3304051

[27] Sayandeep Saha, Dirmanto Jap, Sikhar Patranabis, Debdeep Mukhopadhyay, Shivam Bhasin, and Pallab Dasgupta. 2018. Automatic characterization of exploitable faults: A machine learning approach. *IEEE Transactions on Information Forensics and Security* 14, 4 (2018), 954–968.

[28] G. E. Suh and S. Devadas. 2007. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *2007 44th ACM/IEEE Design Automation Conference*. 9–14.

[29] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. 2017. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*. 269–277.

[30] E. I. Vatajelu, G. Di Natale, and P. Prinetto. 2016. Towards a highly reliable SRAM-based PUFs. In *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*. 273–276.

[31] E. I. Vatajelu, G. D. Natale, M. S. Mispan, and B. Halak. 2019. On the Encryption of the Challenge in Physically Unclonable Functions. In *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. 115–120.

[32] Honggang Yu, Haocheng Ma, Kaichen Yang, Yiqiang Zhao, and Yier Jin. [n.d.]. DeepEM: Deep Neural Networks Model Recovery through EM Side-Channel Information Leakage. ([n. d.]). To Appear in IEEE HOST 2020.

[33] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph. Stoecklin, Heqing Huang, and Ian Molloy. 2018. Protecting Intellectual Property of Deep Neural Networks with Watermarking. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018*, Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim (Eds.). ACM, 159–172. https://doi.org/10.1145/3196494.3196550