DeePar-SCA: Breaking Parallel Architectures of Lattice Cryptography via Learning Based Side-Channel Attacks

Furkan Aydin (✉), Priyank Kashyap (✉), Seetal Potluri, Paul Franzon, and Aydin Aysu

North Carolina State University, Raleigh NC 27606, USA
{faydn,pkashya2,spotlur2,paulf,aaysu}@ncsu.edu

Abstract. This paper proposes the first deep-learning based side-channel attacks on post-quantum key-exchange protocols. We target hardware implementations of two lattice-based key-exchange protocols—Frodo and NewHope—and analyze power side-channels of the security-critical arithmetic functions. The challenge in applying side-channel attacks stems from the single-trace nature of the protocols: each new execution will use a fresh and unique key, limiting the adversary to a single power measurement. Although such single-trace attacks are known, they have been so far constrained to sequentialized designs running on simple microcontrollers. By using deep-learning and data augmentation techniques, we extend those attacks to break parallelized hardware designs, and we quantify the attack's limitations. Specifically, we demonstrate single-trace deep-learning based attacks that outperform traditional attacks such as horizontal differential power analysis and template attacks by up to 900% and 25%, respectively. The developed attacks can therefore break implementations that are otherwise secure, motivating active countermeasures even on parallel architectures for key-exchange protocols.

Keywords: Deep-Learning · Power side-channels · Lattice-based key-exchange protocols.

1 Introduction

Key-exchange protocols enable computers to communicate over a public, insecure channel by establishing a secure session key. Lattice-based key-exchange protocols are versatile post-quantum alternatives, which have already found industry adoption even prior to the National Institute of Standards and Technology (NIST) post-quantum standardization. Google's Chrome Canary web browser, e.g., used NewHope, a post-quantum key-exchange (PQKE) protocol to provide a quantum-secure connection [9].

While lattice-based cryptography provides efficient implementations and quantum resilience, their implementations have shown vulnerability against power side-channel attacks (SCAs) in the context of public-key encryption or digital signatures [3,16,32,37]. These attacks exploit the correlation between the power consumption of a cryptographic device and the secret-key dependent computations. Conventional attacks such as the differential power analysis (DPA) finds the secret-key by extracting this small correlation from noise through collecting a large number of traces. DPA on PQKE protocols is, however, impractical because these protocols generate a new secret-key for each key-exchange session. Therefore, the attacker is *limited to a single power measurement* for applying the SCA.

Recently, Aysu et al. [3] demonstrated that horizontal DPA against PQKE extracts the secret-key from a single power-trace. Others have likewise addressed this single-trace constraint through template attacks (TAs) [8,22,33,37]. However, all these works have focused on simple micro-controllers, such as ARM Cortex-M0 [8,22], Cortex-M4 [33], Cortex-M4F [37], or sequentialized hardware designs [3]. These attacks are expected to perform poorly on parallel hardware designs due to increased activity (i.e., algorithmic noise) [47] and their success rate is unknown.

In this paper, we extend power-based SCAs on lattice-based key-exchange protocols to parallelized hardware designs. We demonstrate the limitations of the existing attacks and address those limitations through power-based SCAs using deep-learning (DL) techniques. We use the NIST Round-2 version of `Frodo` [7] and `NewHope` [2] protocols as case studies. In addition to the industry attraction, both protocols are among the ongoing candidates of the NIST standards. We implement the security-critical operations of matrix and polynomial multiplication hardware that are the target of SCAs [3]. We implemented these hardware architectures at five distinct parallelization levels to evaluate success rate of side-channel attacks on different parallel architectures. Using signal processing and data augmentation techniques, we develop novel DL-based attacks on power measurements obtained from these hardware.

On a SAKURA-G Field Programmable Gate Array (FPGA) platform, our method is better than classical techniques such as horizontal DPA and TA by up to 150% and 11%, respectively, for `Frodo` and 900% and 25%, respectively, for `NewHope`. The results validate the superiority of DL-based attacks in breaking parallel hardware, which can otherwise be secure against earlier techniques. Therefore, there is a need to employ active countermeasures against sophisticated SCAs, even in the context of ephemeral keys and parallel hardware designs.

The major contributions of this paper are:

- We design parallel architectures of the arithmetic functions of `Frodo` and `NewHope` protocols using the latest, Round-2 specifications, and we show that the parallel architectures are still vulnerable to power-based SCAs.
- We develop DL-based single-trace SCAs and quantify that they are superior to classical techniques by up to 150% and 900%, respectively, for `Frodo` [7] and `NewHope` [2]. This shows that DL-based SCAs are able to generalize well to noisy implementations, namely parallel implementations.
- We evaluate the effect of the state-of-the-art SCA data augmentation techniques for DL [24] and reveal that they have marginal impact on accuracy while complicating the training process.

To the best of our knowledge, this is the first work to investigate DL-based SCAs on the next-generation encryption standards.

The rest of the paper is organized as follows. Section 2 provides the background and the prior work on power-based SCAs on lattice-based cryptography, DL-based SCAs, Neural Network (NN) classification, and `Frodo` and `NewHope` algorithms. Section 3 then introduces the novel DL-based SCAs we have developed in the context of lattice-based PQKE. Section 4 describes the parallelized hardware architectures under evaluation. Subsequently, Sect. 5 quantifies the proposed attack's improvement over the classical approaches and finally Sect. 6 concludes the paper.

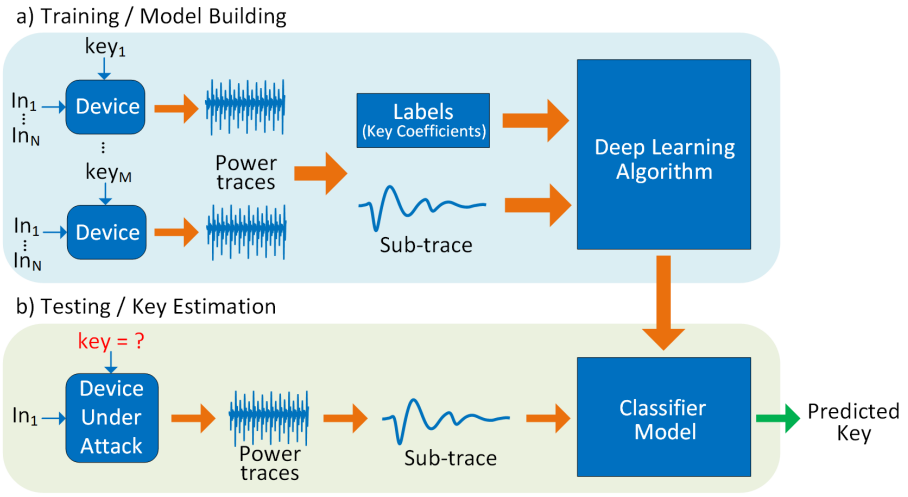

Fig. 1. Deep-Learning based SCAs illustrating a) training and b) testing phases.

2 Background and Prior Work

This section provides previous work and background information about power-based SCAs on lattice-based cryptography, DL-based SCAs, NN classification, and `Frodo` and `NewHope` protocols.

2.1 Power-Based SCAs on Lattice-Based Cryptography

The power consumption of an integrated circuit depends on the data being processed, and SCAs aim to extract this correlation. One of the most widely studied SCA on lattice-based cryptography is DPA [25]. To that end, it is a major threat that works in *unprofiled* settings: the attack analyzes changes occurring on traces (or sub-traces, which are pieces obtained from the same trace) under varying inputs and the fixed secret-key (or sub-keys, which are parts of the key). TAs, by contrast, are *profiled* attacks: the adversary builds multivariate probability density functions for targeted operations and uses them to estimate the secret-key [40]. TA thus requires having access to the target device and programming it with known keys prior to the attack.

Implementations of lattice-based cryptosystems have shown vulnerability to power-based SCAs [3,16,32,37,41]. Applying DPA on lattice-based PQKE protocols is, however, particularly challenging because the protocols generate a new secret-key for each key-exchange session; hence, the attacker is limited to a single power measurement. Aysu *et al.* addressed this challenge, for the first time, through horizontal DPA attacks, which exploited key-dependent, intermediate computations obtained within a single execution [3]. Others have shown TAs that work with a single-trace [8,22,33,37]. All these attacks, however, target serial software or hardware implementations and are expected to be less efficient on parallelized hardware designs because of the higher algorithmic noise.

2.2 DL-based SCAs

The advantages of traditional machine learning approaches such as support vector machines (SVM) and random forests (RF) over DPA and TA have been highlighted in

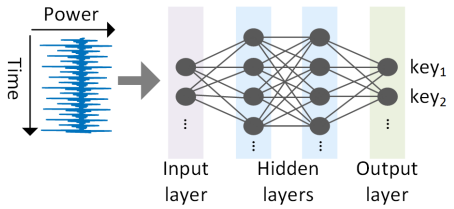

Fig. 2. DL-based classifier (Neural Network) taking samples of power-trace as an input and predicting a key as a label. Each sample in time domain of the power-trace is a neuron at the input layer and the ultimate prediction of the key guess is also a neuron at the output layer.

prior works [21,27]. However, there has been a surge in DL-based SCA-techniques to improve classification performance. Figure 1 outlines the method in such profiled attacks. During training, the adversary builds a profile of the device under different keys and inputs through a DL-based classifier, which then at test time estimates the secret-key. The primary underlying motivation of using DL is that it can apply filtering and alignment of traces in an automatic way, which was typically dealt with by ad-hoc methods known to side-channel experts. Several works have indeed used DL techniques to show how different methods can lend themselves to SCAs [18,24,29,34,38,45].

Unfortunately, all prior work on DL-based SCAs focuses on symmetric-key encryption functions such as AES and PRESENT or quantum-vulnerable cryptosystems such as RSA [12]. By contrast, we extend such attacks, for the first time, to PQKE.

2.3 NN Classification

The goal of any classifier is to take an input vector $\mathbf{x}$ and assign it to one of K discrete-classes $\mathcal{C}_k$, where $k = 1, 2, \ldots, K$. Figure 2 shows the DL-based classification to predict the secret-key. In our case, $\mathbf{x}$ refers to power-traces, and $\mathcal{C}_k$ corresponds to keys—the classifier is trained with this data.

NN consist of multiple layers of neurons with activation functions typically based on the rectified linear unit (ReLU) to model the non-linearity [31]. The network layers consist of an *input layer*, an *output layer*, and *hidden layers*. Neural networks that consist of multiple hidden layers are considered as deep neural networks. The input layer represents the dimensions of the input data, e.g., the number of samples in a power-trace. The output layer tends to have as many neurons as the number of classes to predict. The number of hidden layers and the number of neurons in the hidden layers vary based on the task. For instance, to capture complex features of the input, we increase the number of layers or number of neurons in each layer, whereas, in the event of overfitting, we reduce the layers in the network or number of neurons per layer.

CNN. CNNs are a major class of DL techniques, which have shown better performance than classical approaches for image recognition [44]. A CNN is a special kind of neural network that is built for processing information that has a grid-like structure. Figure 3 shows a CNN, which consists of the following layers:

– *Convolutional Layer*: The convolutional layer extracts features from the input while preserving the relationship between different pixels. In this layer, a filter of a certain size passes over the input and performs convolutions to produce a set of linear activations. The filter then moves by a fixed amount, referred to as the stride.

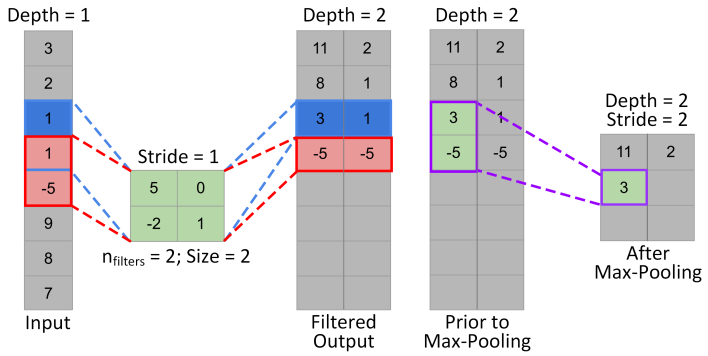

Fig. 3. Convolutional and pooling layers of a CNN. The input passes 2 filters of size 2, after which the output is sub-sampled using maximum pooling with a stride of 2.

- *Pooling layer*: This layer follows the convolutional layer and sub-samples the produced feature map to make it invariant to small changes in the input [19]. Similar to the convolutional layer, a filter determines which values to sub-sample, and a stride length defines how much to move the filter. Pooling generally is either Maximum Pooling which takes the maximum of the values of the image in the filter area or average pooling which takes the average of all values in the filter.
- *Fully Connected Layer*: This layer predicts the probability distributions of the input over different classes. The function is a fully connected (dense) layer of neurons, with the outputs having a *Softmax* activation. This is because *Softmax* converts logits, which are raw scores of the output layer, to probabilities $S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$, where y_i is the i^{th} logits value [11].
- *Batch Normalization Layer*: During the training of neural networks, updating parameters of an earlier layer results in the parameters of a later layer to change. This phenomenon is known as the internal covariate shift [23], which results in slowing down the training by requiring the learning rates of the models to be small. Batch normalization addresses this problem by normalizing the parameters of the network layer-wise and readjusting the parameters and improving training time [23].
- *Dropout Layer*: Neural networks are difficult to train and often tend to overfit to the training data. Dropout is a technique that addresses this problem by randomly dropping neurons (with their connection) during training which prevents the network from learning data from the training set only and improving the performance of the network [43].

2.4 `Frodo` and `NewHope` Algorithms

`Frodo` [7] and `NewHope` [2] are key-exchange protocols that allow two or more parties to establish a unique, symmetric key over an insecure medium. Parties communicate their share of the secret-key in such a way that the adversary eavesdropping on the exchanged information cannot recover the secret-key. `Frodo` and `Newhope` algorithms include arithmetic functions of matrix and polynomial multiplication, respectively, where the secret-key is multiplied with a known input. Hence, the SCA-security of the protocols can be evaluated by analyzing these operations [3].

Frodo and NewHope have different parameter options depending on the desired security level. We analyze Frodo and NewHope, which aim Security Level 5 in the NIST call for proposals. For Frodo, this corresponds to using matrices of sizes $n \cdot n$, $n \cdot \tilde{n}$, $\tilde{m} \cdot n$, and $\tilde{m} \cdot \tilde{n}$ where n, $\tilde{n}$, and $\tilde{m}$, are respectively, 1344, 8, and 8 with integer elements modulo 2^{16}. For NewHope, this corresponds to operating with polynomials of degree 1023 with integer coefficients modulo 12289.

Since breaking Frodo and Newhope is equivalent to successfully applying SCAs on matrix/polynomial multiplication [3], we focus on these operations with $\mathbf{A} \cdot \mathbf{S}$ and $\mathbf{a} \cdot \mathbf{s}$, where $\mathbf{A}$ (or $\mathbf{a}$) is the public value and $\mathbf{S}$ (or $\mathbf{s}$) is the secret sub-key. These multiplication functions use the same secret sub-keys more than one time in the computation; therefore, the attacker can extract the secret information from a single power-trace [3].

3 The Proposed DL-based SCAs

This section presents the proposed DL-based SCAs and related challenges. First, we describe the adaptation of existing SCA-techniques based on time-series analysis. We then introduce our CNN based attack using power measurements. Finally, we elaborate on the hyper-parameter tuning of the trained models.

3.1 Neural Network Based Classification

Prior works have shown that time-series power measurements can be used to predict the Hamming weight (HW) or Hamming distance (HD) for an AES-128 8-bit implementation [28,38], which in turn discloses the secret-key. However, predicting these classes is difficult due to the imbalance of HW/HD in the data-set for a given implementation [35]. To address this issue of class imbalance, data balancing techniques have to be used to successfully attack AES [24]. However, in this work, instead of attacking intermediate computations or HW/HD, we attack a coefficient of the key, i.e., a sub-key. This allows the data-set to be balanced in terms of the key and thus ensures smooth training of the model.

Similar to the work by Kim et.al [24], we constructed a simple CNN architecture based on the performance observed on AES, namely a Visual Geometry Group (VGG) based architecture that has been proven to be highly effective for SCAs [24,38].

Our model consists of multiple convolutional layers with a fixed number of channels, followed by a maximum pooling layer. However, the model does not include a batch normalization layer after every pooling layer as we observe overfitting on the training set when we include it in the model. Prior to the last two layers, we flatten the output of the convolutional layer before feeding into a batch normalization layer. After the batch normalization layer, we feed the data into a dropout layer, with the probability of dropping certain connections with a probability of 0.5 during the training. After the dropout layer, we pass the flattened data into a fully connected layer before being passed to the final output layer. The final layer is also a fully connected and consists of 11 neurons for Frodo to predict the 11 possible sub-keys values (similarly, 33 neurons for NewHope). All the hidden layers have ReLU activations and the output layer has *Softmax* activations.

3.2 Hyper-parameter Tuning

Most DL algorithms come with a different set of parameters to control different aspects of the classification task. Hyper-parameter selection can determine the run-time

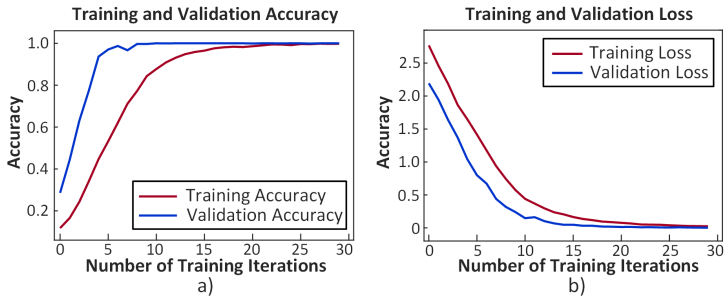

Fig. 4. a) The training and validation loss and b) The training and validation accuracy curves for `Frodo-1` which were used determine the hyper-parameters.

and computational resources required by the algorithm to classify new data (which the algorithm has never seen). To find a set of hyper-parameters for the model, we evaluate the model on the validation set after every iteration and the network stops training when it achieves the highest validation accuracy. Once we find the highest validation accuracy, we re-train the model with the combined test and validation sets and evaluate it on the test set. To prevent overfitting of the CNN, the training stops when the accuracy does not increase after 5 training iterations. The network architecture, kernel size, stride length, filter size, learning rate, batch size and number of training iterations are all hyper-parameters for the CNN, which were tuned to find a suitable network architecture. Once we tune the hyper-parameters mentioned above for `Frodo-1`, we keep them fixed for the remaining implementations, including `NewHope`.

Figure 4 shows that the hyper-parameter tuning found a set of reasonable hyper-parameters as there is no evident overfitting/underfitting observed on the training and validation data.

Loss functions enable the network to determine how much to tune the parameters and are a measure of how well a specific algorithm models the data. The loss function in addition to an optimizer, which updates the weights during training to minimize the loss reduces the error in the model prediction. Accuracy is a metric for classification models use to determine how many classes it predicted correctly. It is the ratio of the total number of correct predictions to the total number of predictions made overall.

During the training process, `Frodo` takes a larger number of training iterations to converge on a high-accuracy solution as compared to `NewHope`. `Frodo` takes 30 iterations to converge on an optimal model whereas `NewHope` takes only 20 iterations.

4 The Proposed Hardware Architectures

To evaluate side-channel security, we implemented the security-critical arithmetic functions of matrix and polynomial multiplications – prior work have identified that the side-channel leakage of these functions results in session key recovery [3]. We proposed five architectures for these multiplications in `Frodo` and `NewHope` algorithms that support different parallelism. Since the proposed architectures process 1, 2, 4, 8, and 16 coefficients in parallel, we labeled these architectures as `Frodo/NewHope-1`, `Frodo/NewHope-2`, `Frodo/NewHope-4`, `Frodo/NewHope-8`, and `Frodo/NewHope-16`. The proposed architectures are based on the prior work [3], but has been modified for Round-2 parameters and extended for parallel computing. Figure 5 and 6 show the

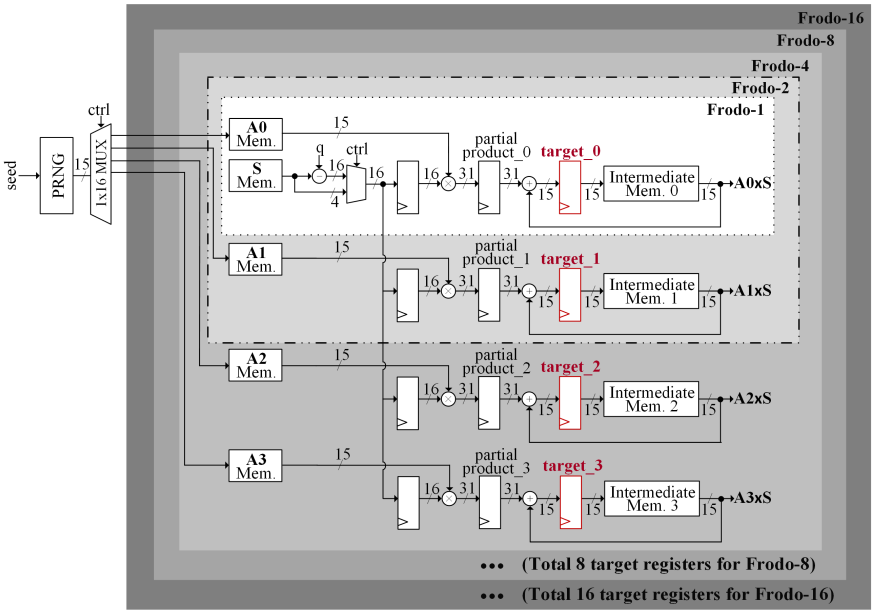

Fig. 5. Hardware architectures and operations under attack for `Frodo`.

architecture details of the developed hardware designs for `Frodo` and `NewHope`, respectively. We colored the architectures with different colors in Fig. 5 and 6 in order to demonstrate the differences between each architecture clearly. `Frodo/NewHope-1/-2/-4/-8/-16`, comprise different numbers of each hardware structure, respectively, 1, 2, 4, 8, 16. For example, `Frodo-4` comprises 4 different target registers that are labeled as in bold red, respectively, target_0, target_1, target_2, and target_4.

For `Frodo`,a pseudo-random generator produces public values (**A**) in the hardware. We used the Trivium algorithm [46] to create random numbers in order to minimize the hardware resource usage without sacrificing the security. After performing two's complement conversions of secret sub-keys (**S**), multiplier(s) that is the main processing unit of all architectures calculates partial products. Then, these partial products are accumulated to the previous value stored in intermediate memory. The `Frodo` designs use modular reduction with a power of two; thus, the modulo reduction is free. The accumulator sum is allowed to exceed q and will only be reduced modulo q since the modulus q is a power of two. In other words, it performs simply a truncation of the adder output to $\log_2 q$ bits.

The difference of `NewHope` compared to `Frodo` is that `NewHope` requires a full-scale reduction after the modular multiplication because the modular reduction is with the constant integer 12289. To that end, we used the *Barrett reduction* technique [30], which computes the modular reduction with two multiplications and a small, fixed number of subtractions. It also removes the errors within a fixed range by a sequence of subtractions and bound checking [3], which is between 0 and 12289×4. We implemented this operation before the intermediate memory updates in parallel to ensure a constant-time operation and reduce power side-channel leakage level of the design [3].

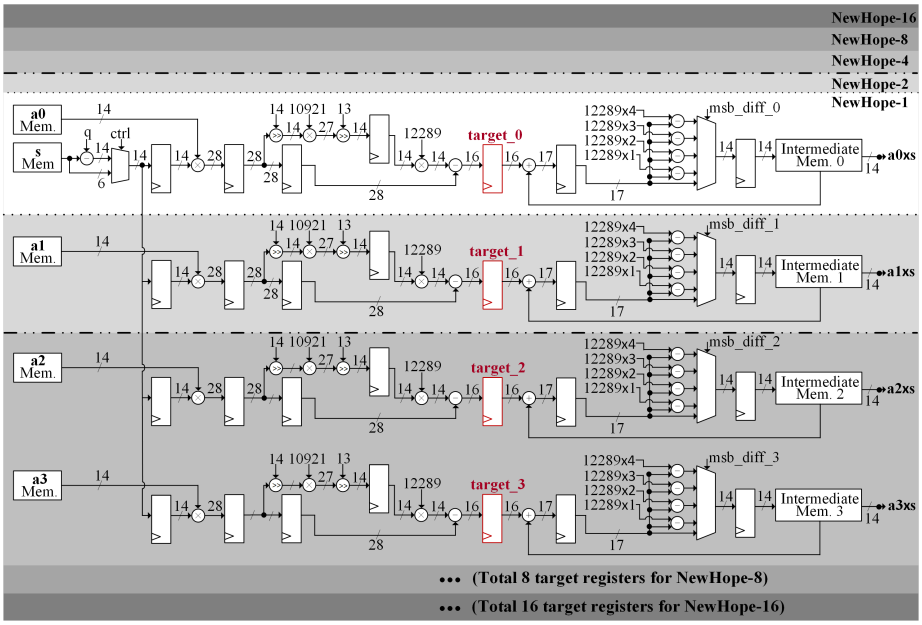

Fig. 6. Hardware architectures and operations under attack for `NewHope`.

The main challenge of `Frodo` and `NewHope` is to implement these architectures in the memory-constrained devices. For example, the size of the matrix **A** requires $1024 \times 1024 \times 14$ bits for the `NewHope` implementation. Since our target FPGA cannot store this amount of data into the memory, we resolve the resource limitation using the on-the-fly calculation approach [7] that enables the computation of **A**·**S**/**a**·**s** without accumulation in the hardware. Although Number Theoretic Transform (NTT)-based polynomial multipliers are possible for `NewHope`, we use the regular (i.e. schoolbook) multiplication. Note that the regular multiplication has some advantages over NTT-based implementations in terms of critical path delay, or area- and power-efficiency [6,10,20,39]. But the regular multiplication is admittedly more susceptible to SCAs.

`Frodo-1/-2/-4/-8/-16` and `NewHope-1/-2/-4/-8/-16` are written in Verilog Hardware Description Language and mapped on to the Xilinx Spartan-6 XC6SLX75. We used Xilinx Integrated Synthesis Environment (ISE) version 14.7 with default settings for synthesizing, placing, and routing of the proposed designs.

5 Evaluation Results and Comparison

This section describes the measurement setup for our experiments and compares the success rate of different attacks.

5.1 Attack Procedure and Experimental Setup

The proposed attacks target multiplications of `Frodo` and `NewHope` protocols, and specifically the intermediate memory updates to remove false positives [3]. We perform a chosen-plaintext attack, where we vary the keys but keep the plaintext constant. Since the number of updates is different for parallel implementations, the effect of false positives is different. Moreover, the parallelization level is inversely proportional to the number of available sub-traces for all SCAs (e.g., 1344 for `Frodo-1` and 672 for

Frodo-2). Our evaluation method is therefore based on the comparison of the success rate vs. the number of sub-traces used.

Horizontal DPA. Our horizontal DPA attack follows the methodology introduced by Aysu et al. [3]. Unlike DL-based attacks and TAs, horizontal DPA does not require a training or profiling phase. It relies on a statistical analysis of several samples where the same keying material is used to operate on dierent data [42]. The sensitive data leaked through the side-channel depends on the number of bits switching in the registers. Our power models use the HD of the register to determine the expected (hypothetical) power consumption of the circuit. We adopt the common method of Pearson correlation coefficient [15] in our statistical analysis to compare the real power consumption values and the hypothetical power consumption values for each sub-key. Correlation trace $\rho_{i,j}$ for a sub-key guess i is dened as:

$$\rho_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \bar{h_i})(t_{d,j} - \bar{t_j})}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \bar{h_i})^2 \sum_{d=1}^{D}(t_{d,j} - \bar{t_j})^2}} \tag{1}$$

where T data points of D number of traces. The hypothetical power consumption value to the appropriate sub-key is defined as h_d, i with $0 < d \leq D$ and real power-trace is defined as $t_{d,j}$ with $0 < j \leq T$. $\bar{h_i}$ and $\bar{t_j}$ represent the mean power estimate and the mean power-trace, respectively. The result of the Pearson correlation is between [-1,1] and depicts the relationship between real power consumption values and the hypothetical power consumption values. Therefore, the maximum absolute correlation coefficient reveals the timing information of the DPA leakage.

TA. To perform a TA, we first create a "profile" of the device and apply this profile to find the secret sub-keys. In other words, we create a template of the device's operation and then apply this template to the attacked traces. The template is effectively a multivariate distribution that describes the key samples in the power-traces.

To reduce the number of samples and the size of the templates, we selected some special values in each trace that are called the point of interest (POI). In the literature, there are different kinds of approaches [4,5,17,42] to find the POIs that vary strongly between different key coefficients. We used the Sum of Squared Difference (SOSD) method that is one of the strongest approaches to improve the classification performance of the TA [17]. We have k different operations and i sample points in a number of traces named $t_1..t_k$. The mean power M and SOSD are calculated as follows:

$$M_{k,i} = \frac{1}{T_k} \sum_{j=1}^{T_k} t_{j,i} \tag{2}$$

$$SOSD_i = \sum_{k_1,k_2} (M_{k_1,i} - M_{k_2,i})^2 \tag{3}$$

Figure 7 shows that there are four peak points for the SOSD. We picked these four peak points as POIs because the power variation of power consumption is maximum at these samples s_i. As an experiment, we used different number of POIs for our TA. However, using more than four POIs did not improve the success rate of TA for Frodo

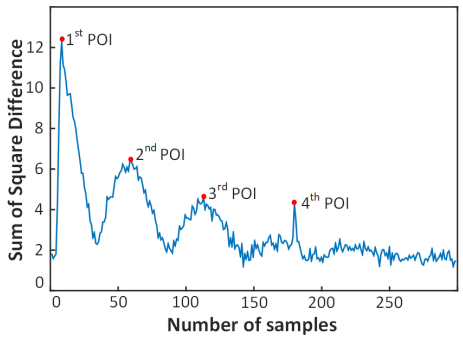

Fig. 7. An example of POI selection from SOSD results.

and `NewHope` because most of the time-samples in each sub-traces are not highly correlated with the sub-keys. Also, using all samples of traces makes an impractical template and decreases the classification performance due to high computational requirement [14,40]. We calculated average power μ_i in (4), variance of power v_i in (5), and covariance c_{i,i^*} for creating covariance matrix in (6) at every pair of POIs (i and i^*).

$$\mu_i = \frac{1}{T_k} \sum_{j=1}^{T_k} t_{j,s_i} \tag{4}$$

$$c_{i,i^*} = \frac{1}{T_k} \sum_{j=1}^{T_k} (t_{j,s_i} - \mu_i)(t_{j,s_i^*} - \mu_{i^*}) \tag{5}$$

$$S = \begin{pmatrix} v_1 & c_{1,2} & c_{1,3} & \dots \\ c_{2,1} & v_2 & c_{1,3} & \dots \\ c_{3,1} & c_{3,2} & v_3 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{6}$$

For the attack step, we calculated the multivariate normal probability density function (MVNPDF) using POIs of the attacked traces t_{j,s_i}, μ_i, and c_{i,i^*} results of the profiled devices. We summed the log of the normal distribution $\mathcal{N}$ to avoid precision issues that occur if the results of MVNPDF are too large or too small.

$$P_k = \sum_{j=0}^{k} \log \mathcal{N}(t_j, \mu, S) \tag{7}$$

The index of the matrix P_k with the highest value corresponds to the sub-key guess.

5.2 Evaluation Setup

Our evaluation setup uses the Sakura-G board, which has a Xilinx Spartan-6 XC6SLX75 FPGA for processing and enables measuring the voltage drop on a 1Ω shunt-resistor while making use of the onboard amplifiers to measure FPGA power consumption. We also use a low noise AC amplifier, which is a PA-203 amplifier from Langer EMV-Technik with 20 dB [26]. We collect power measurements using a PicoScope-3206D

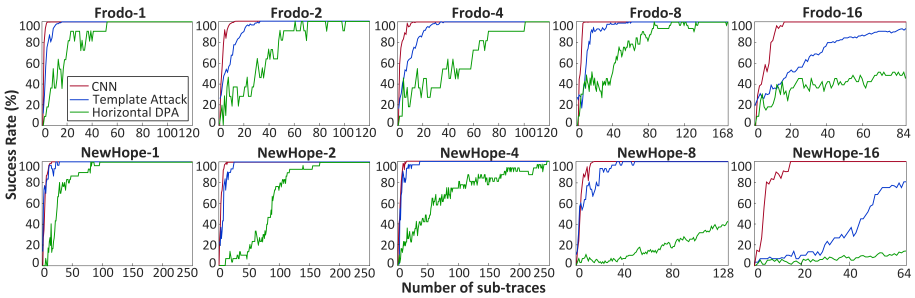

Fig. 8. Comparison of DL-based vs. classical SCAs on `Frodo` and `NewHope`.

Table 1. SNR (dB) for Frodo and NewHope traces at 1.5 MHz

Algorithm/Parallelization Level	1	2	4	8	16
Frodo	6.27 dB	5.64 dB	4.73 dB	3.15 dB	2.19 dB
NewHope	3.81 dB	3.31 dB	2.91 dB	2.61 dB	2.11 dB

model oscilloscope at 500MS/s [36]. We then pre-process the raw power-traces to divide the entire power-trace into sub-traces, which capture the information about the device for one cycle of operation.

We train and evaluate DL-models on a computer with 64 GB of Random Access Memory (RAM), an NVIDIA 2080 Ti graphics card, and an Intel i7 9700K. We use the *TensorFlow* Graphics Processing Unit (GPU) [1] as the backend, with a *Keras* front end to train and evaluate all of the DL-models. We use a categorical cross-entropy as our loss function with *Adagrad* as the optimizer with a learning rate of 0.1 [13]. Using Adagrad allows the learning rate to be adjusted dynamically as the model is training, thus even with a large learning rate, the optimizer will update it to the appropriate value. The average time to train the models is around 10 minutes to achieve over 99% accuracy, similar to the prior work on AES [38].

5.3 Comparison against Conventional Attacks

Figure 8 compares the proposed DL-based attack to horizontal DPA and TA. The DL-based attack achieves 100% accuracy whereas TA and horizontal DPA aren't able to attack the implementation successfully. The DL-based approach outperforms horizontal DPA and TA by up to 900% and 25% in terms of the attained success rate. This performance is attributed to the ability of the CNN to filter out the noise from the power-traces [24]. This quantifies that the proposed DL-based attack needs fewer samples to succeed and hence, can break even further parallelized designs where the conventional techniques would fail.

Table 1 furthermore shows the SNR (dB) values of the captured traces for `Frodo` and `NewHope` implementations at a clock frequency of 1.5 MHz. As expected, the algorithmic noise increases with the increase in parallelization, which impacts the TA and the horizontal DPA. In other words, the TA and horizontal DPA results become worse with low SNR. As a result, there is a steady increase in the number of sub-traces required to attack the parallel implementations. The CNN attack, however, tolerates the algorithmic noise better than TA and horizontal DPA. Consequently, the number of required sub-traces for a successful attack varies between 10-17 sub-traces for the CNN.

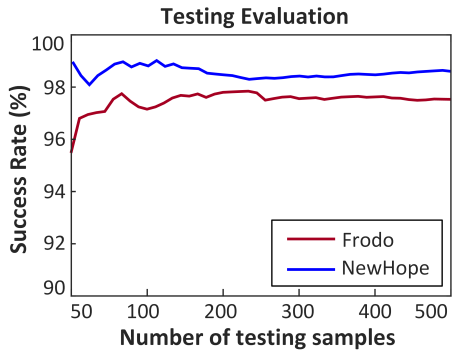

Fig. 9. The effect of testing size on ML SCAs on `Frodo-16` and `NewHope-16`.

5.4 Testing Evaluation Size

For most DL applications, there are millions of data points to both train and test the model. However, in Fig. 8 we limit the evaluation to 200 samples for each sub-key. Though, this is not a large number of samples we observe that evaluating more test samples does not degrade the overall test accuracy of the model as is seen in Fig. 9. As is evident from Fig. 9, the success rate plateaus after 100 and 200 testing samples per sub-key for `NewHope-16` and `Frodo-16` respectively. Thus a smaller testing set is sufficient to represent the overall accuracy of the model.

5.5 Data Augmented vs Proposed Approach

We adapt our CNN to include a batch normalization layer at the input, followed by a layer of Gaussian noise. The addition of the batch normalization is necessary so that noise can be added to the normalized inputs. This acts as a regularization that is only active during the training of the model and has previously been used successfully to reduce any overfitting in SCA [24]. We train the two different models for `Frodo-16` and `NewHope-16` while varying the number of training samples used to create the model at 15 sub-traces. Due to the random noise at the inputs in the augmented case, training takes closer to 50 training iterations to produce a model with similar accuracy on the training set.

We observe in Fig.10 that both models achieve similar results on the test set. As more data is made available, both models are able to learn the features of the power-trace more effectively. In fact, for both `Frodo-16` and `NewHope-16`, the augmented network has a slightly lower accuracy as compared to the regular CNN. Thus, we conclude that the data augmentation technique of adding random noise to the inputs does not improve the performance of the model in this setting as there are sufficient training samples that are able to capture the added noise. However, in an instance when the attacker has access to a limited number of traces, such a noise augmentation can provide a similar result.

6 Conclusions

As the industry and governments gear towards a post-quantum cryptography standardization, implementation security becomes an important issue in addition to the theoretical cryptanalysis of candidate algorithms. This paper analyzes the physical side-channel vulnerability of the hardware designs performing a fundamental arithmetic of

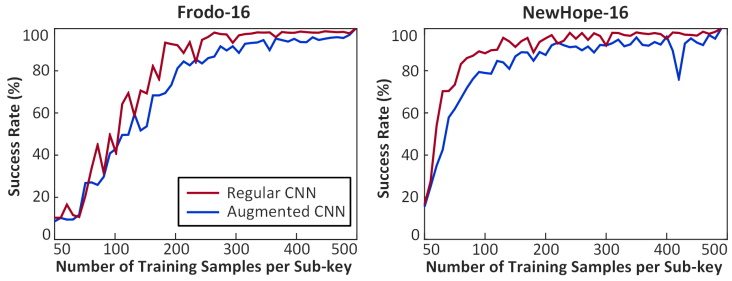

Fig. 10. Varying the training set size to train the regular CNN and a CNN with noise at the inputs to prevent overfitting for `Frodo-16` and `NewHope-16`.

lattice-based cryptography, which are among the prime candidates for post-quantum cryptosystems. We demonstrate the superiority of *deep learning* over classical methods and show that such attacks has a potential to break otherwise secure systems. Therefore, the paper illustrates the need to incorporate active security mechanisms against power side-channels even for the single measurement and parallel hardware use-cases.

Acknowledgements. This research is supported in part by the NSF under the Grants No. CNS 16-244770 (Center for Advanced Electronics through Machine Learning) and CNS 18-50373. NC State is an academic partner of Riscure Inc. and thanks them for providing hardware/software support for side-channel analysis. We acknowledge NVIDIA for their GPU donation and Xilinx for their FPGA donation.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., et al., C.C.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015), `https://www.tensorflow.org/`, software available from tensorflow.org
2. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-Quantum Key Exchange - A New Hope. In: USENIX Security Symposium. pp. 327–343 (2016), `https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim`
3. Aysu, A., Tobah, Y., Tiwari, M., Gerstlauer, A., Orshansky, M.: Horizontal Side-channel Vulnerabilities of Post-Quantum Key Exchange Protocols. In: IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 81–88 (2018)
4. B. Gierlichs, K.L.R., Paar, C.: Templates vs. stochastic methods a per- formance analysis for side channel cryptanalysis. CHES (2006)
5. B. Gierlichs, L. Batina, P.T., Preneel, B.: Mutual information analysis. CHES (2008)
6. Bian, S., Hiromoto, M., Sato, T.: Filianore: Better Multiplier Architectures For LWE-Based Post-Quantum Key Exchange. In: ACM/IEEE Design Automation Conference (DAC). pp. 1–6 (2019)
7. Bos, J., Costello, C., Ducas, L., Mironov, I., et. al., M.N.: Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In: ACM SIGSAC Conference on Computer and Communications Security. pp. 1006–1018 (2016)
8. Bos, J.W., Friedberger, S., Martinoli, M., Oswald, E., Stam, M.: Assessing the Feasibility of Single Trace Power Analysis of Frodo. In: Selected Areas in Cryptography (SAC). pp. 216–234 (2018)
9. Braithwaite, M.: Google Security Blog: Experimenting with Post-Quantum Cryptography (July 2016), `https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html`

10. Buchmann, J., Göpfert, F., Güneysu, T., Oder, T., Pöppelmann, T.: High-Performance and Lightweight Lattice-Based Public-Key Encryption. In: ACM International Workshop on IoT Privacy, Trust, and Security. pp. 2–9 (2016)
11. Campbell, D., Dunne, R.A., Campbell, N.A.: On The Pairing Of The Softmax Activation And Cross–Entropy Penalty Functions And The Derivation Of The Softmax Activation Function. In: Australian Conference on Neural Networks. pp. 181–185 (1997)
12. Carbone, M., Conin, V., Cornelie, M., Dassance, F., Dufresne, G., Dumas, C., Prouff, E., Venelli, A.: Deep Learning to Evaluate Secure RSA Implementations. IACR Transactions on Cryptographic Hardware Embedded Systems **2019**(2), 132–161 (2019)
13. Chollet, F., et al.: Keras (2015), https://keras.io
14. Chong, T., Kaffes, K.: Hacking AES-128. SemanticScholar (2016)
15. E. Brier, C.C., Olivier, F.: "correlation power analysis with a leakage model. CHES (2004)
16. Espitau, T., Fouque, P.A., Gérard, B., Tibouchi, M.: Side-Channel Attacks on BLISS Lattice-Based Signatures: Exploiting Branch Tracing Against strongSwan and Electromagnetic Emanations in Micro-controllers. In: ACM SIGSAC Conference on Computer and Communications Security. pp. 1857–1874 (2017)
17. Fan, G., Zhou, Y., Zhang, H., Feng, D.: How to choose interesting points for template attacks more effectively? In: International Conference on Trusted Systems. vol. 9473, pp. 168–183 (January 2014)
18. Gilmore, R., Hanley, N., O'Neill, M.: Neural Network based Attack on a Masked Implementation of AES. In: IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 106–111 (2015)
19. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org
20. Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems. CHES (2012)
21. Heuser, A., Zohner, M.: Intelligent Machine Homicide. In: Constructive Side-Channel Analysis and Secure Design. pp. 249–264. Springer (2012)
22. Huang, W.L., Chen, J.P., Yang, B.Y.: Power Analysis on NTRU Prime. IACR Transactions on Cryptographic Hardware and Embedded Systems **2020**(1), 123–151 (Nov 2019). https://doi.org/10.13154/tches.v2020.i1.123-151, https://tches.iacr.org/index.php/TCHES/article/view/8395
23. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: International Conference on Machine Learning (ICML). pp. 448–456. PMLR (2015)
24. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: 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), 148–179 (2019), https://tches.iacr.org/index.php/TCHES/article/view/8292
25. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: International Cryptology Conference on Advances in Cryptology (CRYPTO). pp. 388–397. Springer-Verlag (1999)
26. Langer EMV: PA 203 SMA set, Pre-amplifier 100 KHz up to 3 GHz, https://www.langer-emv.de/en/product/preamplifier/37/pa-203-sma-set-preamplifier-100-khz-up-to-3-ghz/518
27. Lerman, L., Bontempi, G., Markowitch, O.: Power Analysis Attack: An Approach Based on Machine Learning. International Journal of Applied Cryptography **3**(2), 97–115 (Jun 2014)
28. Maghrebi, H.: Deep Learning based Side Channel Attacks in Practice. IACR Cryptology ePrint Archive, Report 2019/578 (2019), https://eprint.iacr.org/2019/578
29. Maghrebi, H., Portigliatti, T., Prouff, E.: Breaking Cryptographic Implementations Using Deep Learning Techniques. In: Security, Privacy, and Applied Cryptography Engineering (SPACE). pp. 3–26 (2016)

30. Menezes, A.J., Van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC press (July 2016)
31. Nair, V., Hinton, G.E.: Rectified Linear Units Improve Restricted Boltzmann Machines. In: International Conference on International Conference on Machine Learning (ICML). pp. 807–814. PMLR (2010)
32. Oder, T., Schneider, T., Pöppelmann, T., Güneysu, T.: Practical CCA2-Secure and Masked Ring-LWE Implementation. IACR Transactions on Cryptographic Hardware Embedded Systems **2018**(1), 142–174 (2018)
33. Pessl, P., Primas, R.: More Practical Single-Trace Attacks on the Number Theoretic Transform. In: LATINCRYPT. pp. 130–149 (2019)
34. Picek, S., Heuser, A., Jovic, A., Ludwig, S.A., Guilley, S., Jakobovic, D., Mentens, N.: Side-channel Analysis and Machine Learning: A Practical Perspective. In: International Joint Conference on Neural Networks (IJCNN). pp. 4095–4102 (2017)
35. Picek, S., Heuser, A., Jovic, A., Bhasin, S., Regazzoni, F.: The Curse of Class Imbalance and Conflicting Metrics with Machine Learning for Side-channel Evaluations. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(1), 209–237 (2018), https://tches.iacr.org/index.php/TCHES/article/view/7339
36. Pico Technology: PicoScope 3206D Model Oscilloscope, https://www.picotech.com/oscilloscope/3000/picoscope-3000-oscilloscope-specifications
37. Primas, R., Pessl, P., Mangard, S.: Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption. In: CHES. pp. 513–533 (2017)
38. Prouff, E., Strullu, R., Benadjila, R., Cagli, E., Dumas, C.: Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database. IACR Cryptology ePrint Archive **2018**, 53 (2018), http://eprint.iacr.org/2018/053
39. Pöppelmann, T., Güneysu, T.: Area Optimization of Lightweight Lattice-based Encryption on Reconfigurable Hardware. In: IEEE International Symposium on Circuits and Systems (ISCAS). pp. 2796–2799 (2014)
40. Rechberger, C., Oswald, E.: Practical Template Attacks. In: International Workshop on Information Security Applications. pp. 440–456 (2004)
41. Reparaz, O., Roy, S.S., Vercauteren, F., Verbauwhede, I.: A Masked Ring-LWE Implementation. In: CHES. pp. 683–702 (2015)
42. S. Chari, J.R.R., Rohatgin, P.: Template attacks. CHES (2002)
43. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014)
44. Sultana, F., Sufian, A., Dutta, P.: Advancements in Image Classification using Convolutional Neural Network. In: International Conference on Research in Computational Intelligence and Communication Networks. pp. 122–129 (2018)
45. Timon, B.: Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(2), 107–131 (2019), https://tches.iacr.org/index.php/TCHES/article/view/7387
46. Y. Tian, G.C., Li, J.: On the Design of Trivium. Cryptology ePrint Archive,Report 2009/431, 2009 (January 2009), http://eprint.iacr.org/
47. Zhang, L., Vega, L., Taylor, M.: Power side channels in security ics: Hardware countermeasures. CoRR **abs/1605.00681** (2016), https://arxiv.org/abs/1605.00681