Journal of Cryptographic Engineering
https://doi.org/10.1007/s13389-023-00342-0

RESEARCH ARTICLE

Keep it unbiased: a comparison between estimation of distribution algorithms and deep learning for human interaction-free side-channel analysis

Unai Rioja[1,2] · Lejla Batina[2] · Igor Armendariz[1] · Jose Luis Flores[1]

Received: 14 December 2022 / Accepted: 12 November 2023

Abstract
Evaluating side-channel analysis (SCA) security is a complex process, involving applying several techniques whose success depends on human engineering. Therefore, it is crucial to avoid a false sense of confidence provided by non-optimal (failing) attacks. Different alternatives have emerged lately trying to mitigate human dependency, among which deep learning (DL) attacks are the most studied today. DL promise to simplify the procedure by e.g. evading the need for point of interest (POI) selection, among other shortcuts. However, including DL in the equation comes at a price, since working with neural networks is not straightforward in this context. Recently, an alternative has appeared with the potential to mitigate this dependence without adding extra complexity: estimation of distribution algorithm-based SCA. From the perspective of avoiding the need for POI selection, in this paper we provide a comparison of the two relevant methods. Our findings are supported by experimental results on various datasets.

Keywords Hardware security · Side-channel analysis · Machine learning · Estimation of distribution algorithms · Artificial intelligence · Evaluation

1 Introduction

These days we are surrounded by IoT devices that handle sensitive information, not only in industrial applications but also in our daily lives. This requires from designers and product developers to use cryptography to protect embedded devices, but cryptography is only one of the components ensuring the security of systems. In the real world, the security of a device depends not only on the mathematical characteristics of its cryptographic operations but also on its physical implementation. In other words, the physical nature of these devices can be exploited in order to break their security in several ways, whereas some approaches are passive and rely on simply observing certain physical properties to retrieve information (side-channel analysis, SCA), other procedures try to stress the system to alter its natural behaviour (fault injection, FI). Physical attacks are truly powerful, as they can bypass hardware and software security measures that the manufacturer has included in the design.

The problem is that the inclusion and validation of countermeasures against this kind of attacks is not simple, especially in the SCA case. Current certification schemes require attacking the device under test (DUT) with a battery of known SCA attacks to prove its security [1]. Unfortunately, this approach is prohibitive in terms of time and resources. The ever-growing number of attack techniques, which in turn involve knowledge of very diverse subjects (e.g. signal processing, electronic design, cryptography, statistics, machine learning, etc.), make it difficult to master and correctly apply all of them. Moreover, the outcome of such attacks depends to a large extent on the experience of the person performing them. Therefore, a process so dependent on human interac-

✉ Unai Rioja
urioja@ikerlan.es

Lejla Batina
lejla.batina@ru.nl

Igor Armendariz
iarmendariz@ikerlan.com

Jose Luis Flores
jlflores@ikerlan.com

[1] Ikerlan Technological Research Centre, Basque Research and Technology Alliance (BRTA), Arrasate-Mondragón, Spain

[2] Digital Security Group, Radboud University, Nijmegen, The Netherlands

Published online: 16 December 2023

tion can lead to a biased result if the tests are not properly executed.

One of the most crucial and human-dependent steps of SCA attacks is the point of interest (POI) selection [11, 36]. This step consists of selecting only a small set of the usual big number of time samples of the power traces (containing the leakage information) to perform the analysis. In practice, this step is commonly quite tedious, as the evaluator has to manually perform several analyses with different POI combinations until the desired outcome is obtained. However, as Picek et al. illustrated in [34], many related works start the analysis with the POIs already preselected, being a quite unrealistic scenario. This, together with the fact that similar results can be obtained with various methods (if properly executed), makes a less error-prone and evaluator-dependent alternative necessary.

One of the most prominent techniques in SCA research today is deep learning (DL)-based SCA [3, 21, 25, 38, 44], part of the so-called profiling attacks (the strongest SCA technique nowadays). This method aims to overcome some drawbacks of classical SCA, such as the need for preprocessing or POI selection, promising a relaxation of the evaluator interaction. Conversely, some other works have shown that, rather than removing the need for trace realignment, the DL approach simply alleviates it, as preprocessing can substantially enhance the performance of neural networks [29, 49]. Moreover, working with DL is complex, especially considering the large number of possible architectures and hyper-parameters to adjust. Besides, although some attempts have been made [21, 44, 48], the SCA community has not yet agreed on how models should be constructed and evaluated. In addition, there is no generalized solution: when attacking new datasets, devices or cryptographic implementations, sometimes it is necessary to completely readjust the neural network. Consequently, while some of the more common difficulties are alleviated, new DL-related complications emerge.

Recently, an option that promises to lessen this human dependency without increasing the overall complexity has emerged: estimation of distribution algorithm-based profiling attacks (EDA-based PAs) [39, 40]. This concept comprises applying randomized optimization heuristics to the POI selection issue, allowing an automated boosting of the whole attack (POI selection, leakage profiling and key recovery). This method can produce state-of-the-art results straightforwardly, even in noisy environments [39]. Although they provide a simple alternative to DL, to the best of our knowledge, both approaches have never been directly compared. Therefore, in this paper, a comparison between the two methods is driven, from the perspective of avoiding the need for POI selection.

Thus, the contributions of this work are the following:

1. A comparative analysis of both approaches has been driven, in terms of complexity and performance, to highlight the strengths and weaknesses of each method.
2. Our findings are based on repeatable experimental evidence. Some experiments have been carried out, especially for this paper (DL-based attacks on the AES_RA dataset[1]), while for others we have relied on related works to provide a comprehensive and impartial comparison.
3. We also test the performance of several DL models previously designed for ASCAD [3] (a well-known dataset in the field). The approach is to test the models in nearly identical scenario (first-order masked AES software implementation in which both shares are manipulated in the targeted time window) to show how even in this case, the evaluator would have to manually readjust the neural network for the model to work most of the times.
4. We also study the performance of several SCA-oriented DL architectures in unfavourable conditions (no mask leakage during the targeted window) to determine whether the same conclusions as in [39] using EDAs hold.
5. Besides, we test the performance of both EDA-based and DL-based attacks in a portable scenario. That is, performing the attack on different hardware copies of the same device. For that purpose, we employ the recently extended version of AES_RA (introduced and employed in [39]): AES_PTv2.[2] We conclude that similar results can be obtained with both techniques.

The paper is organized as follows: In Sect. 2 we summarize relevant background on profiling attacks. We describe the related work that is closest to ours in Sect. 3. We present the comparison of the two techniques (EDA-based and DL-based) in Sect. 4. In Sect. 5 we discuss the comparison and elaborate on our findings. Section 6 concludes the paper.

2 Background on profiling attacks

Profiling attacks (PAs) have become the prevailing type of SCA attack in recent years [3, 11, 39, 48, 50]. The original idea comprises generating a model of the power consumption of a device to be deployed for the secret key recovery. These attacks involve two phases (see Fig. 1): a phase in which the model is generated using a typically big number of traces (profiling phase), and the phase in which this model is compared with the actual power consumption of the device to recover sensitive information (attack phase).

Depending on the method used to generate the model, there exist different types of profiling attacks. In this paper,

[1] AES_RA Dataset: https://github.com/AES-RA/AES_RA.

[2] AES_PTv2 Dataset: https://github.com/urioja/AESPTv2.

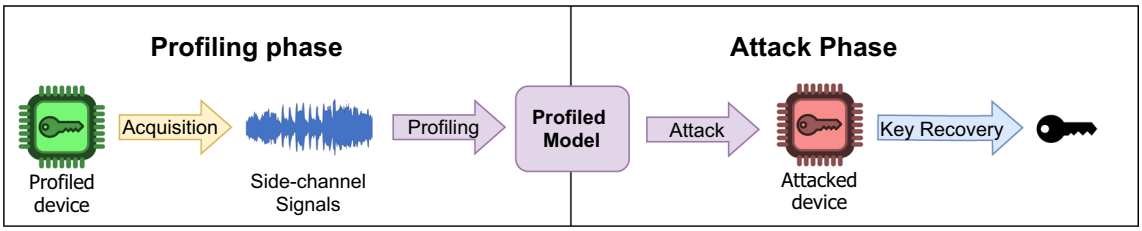

Fig. 1 Scheme of a generic profiling attack

we focus on template attacks (TAs) and deep learning (DL)-based Profiling Attacks [25, 35] for being the most widely used options in practice [23, 50].

2.1 Template attacks

Template attacks are the first type of profiling attacks introduced and they involve building a multivariate model of the probability distribution of the leakage. The practice is to use an extensive set of power traces taken from the DUT when it is manipulating some intermediate value $v = f(p, k)$. As long as v is related to a known variable (usually plaintext p or ciphertext c) and the secret key k, guessing v allows the attacker to disclose the secret key.

First, in the *profiling phase*, the attacker employs a set of n_p profiling traces ($\mathbf{T}_p$) to build a Gaussian multivariate model, for each possible v, creating the so-called *templates* (pairs of mean vectors and a covariance matrices ($\mathbf{m}$, $\mathbf{C}$)). Note that, as each v depend on d and k, each key hypothesis suggests a template. Finally, a *discriminant score* $D(k \mid \mathbf{t}_i)$ is computed for each key hypothesis k_j and all key hypothesis are graded in decreasing order of probability, creating a key guessing vector. In SCA, it is common to work with a metric called Guessing Entropy (ge), which is the average rank of the correct candidate k^* after several attacks.

In practice, TAs pose several limitations such as complexity issues or the need for dimensionality reduction [11]. The latter is often fixed by choosing just a few time samples of the power traces (POIs selection [36]), or employing a more complex technique like principal component analysis (PCA) or Fisher's linear discriminant analysis (LDA) [42]). Note that, with EDA-based PA, the POI selection is performed automatically by the algorithm [40].

2.2 DL-based SCA

In recent times, multiple papers related to DL-based SCA have been published. The approach is the same as for template attacks, but the model building and classification are performed using neural networks. Most of these works rely on two typical deep learning architectures: multilayer perceptron (MLP) and convolutional neural networks (CNN). The early architecture used in DL-SCA for its simplicity was MLP. The first proposal was applying regression to characterize leakage [46] but the approach rapidly evolved to use MLP as a classifier for intermediate values of the traces [25, 27]. After this, CNNs also began to be used because its spatial invariance property provides robustness against data distortions like environmental noise, desynchronization and countermeasures [3, 25, 35]. Other studies have examined the performance of various PAs [3, 35, 48, 50].

As stated above, one of the major drawbacks of classical SCA is the need for pre-processing and POI selection, as this part is strongly dependent on human engineering. DL-SCA claims to overcome those struggles, since the features are selected automatically from traces by the neural network. In any case, note that although this part of the problem may be mitigated, working with neural networks is a complex process that also requires human interaction for architecture selection, tuning and training of the neural networks to operate correctly.

2.3 Masking protected implementations

SCA countermeasures try to obfuscate the dependency between the power consumption of the DUT and the intermediate values of the implemented cryptographic algorithm. One of the most popular ones is masking. Masking (also known as secret sharing) comprise concealing each intermediate value v with a random value m (mask), which is different for each execution and unknown by the attacker, such as $v_m = v * m$. If correctly implemented, this ensures pairwise independence between masked values, unmasked values and the masks. Consequently, a classical (first-order) attack would fail.

Although theoretically sound, implementing masking incorrectly can be fatal for a device's security. Close manipulation of the shares can provoke unintended interactions between values in the microcontroller, principally caused by transitional effects [2] and glitches [10, 26]. These phenomena can halve the security order, making the first order masking (with two shares or mask and masked intermediate value) vulnerable even to first-order attacks [2]. This is important since this is the most widely used scheme in practice, because of the complexities involved in higher-order masking [41].

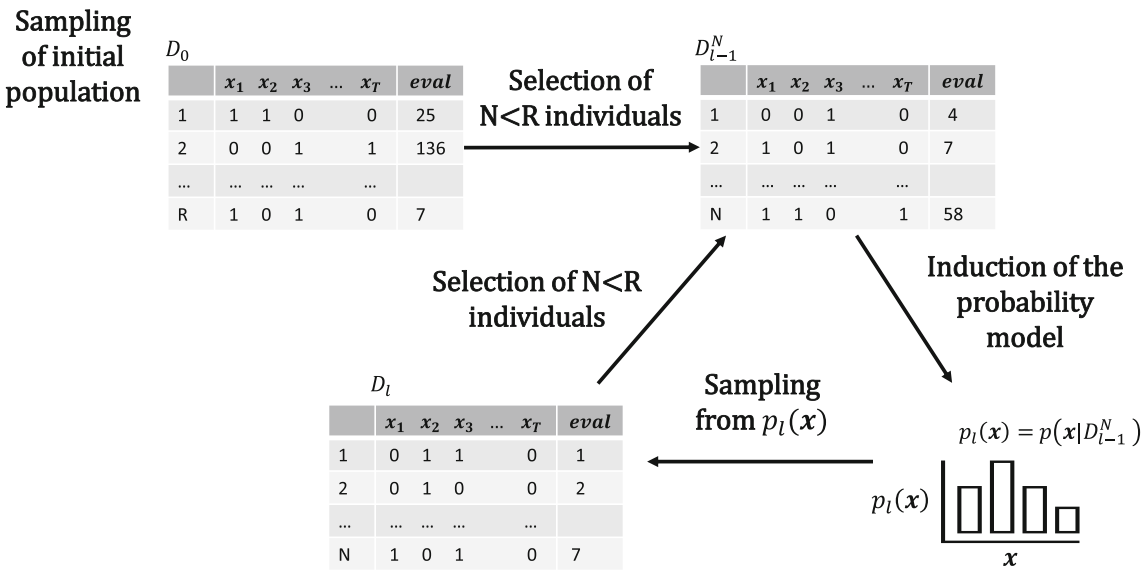

Fig. 2 Illustration of a generic EDA-based PA

3 Related work

In this section, we present several previous related works which are relevant for this paper.

3.1 EDA-based SCA

The usage of estimation of distribution algorithms (EDAs) in the SCA context was introduced in [40] as an alternative to adjust the POI selection and perform template attacks over embedded devices in an automated way. EDAs are a class of population based optimization heuristics that explore potential results by forming explicit probabilistic models of promising candidates. The approach is to seek in the space shaped for all groupings of POIs for the best ones. Instead of an exhaustive enumeration of all possible combinations, this method performs a search based on a quality measure combined with EDAs.

Figure 2 summarizes the process. In a nutshell, in the first place, an initial population of R individuals (POI selection candidates) is generated. This population can be generated at random or according to some criterion (i.e. assigning a higher probability to samples that present a stronger correlation with the processed intermediate values [40]). After this, R attacks are performed with the R candidates and the candidates are rated according to their performance. Then, the best N candidates are chosen ($N < R$) and the probability distribution $p(\mathbf{x})$ of potential candidates is estimated from them. The process is repeated until a stop condition is reached (see [40] for more details).

Although they present several advantages against DL, the authors of [39, 40] have contrasted it with other classical (manual) POI selection techniques but have never compared it directly with DL-based attacks. Thus, in this paper, we compare both techniques on various datasets, in terms of performance and complexity.

3.2 Profiling attacks on masking

On the one hand, although PAs are a derivative of DPA attacks, many papers claim that (first-order) DL-based SCA attacks can deal with masking countermeasures [3, 18, 25]. The claim is that, in these attacks, the network is trained without mask information (the traces are labelled with the unmasked intermediate value v). Despite this, as deep neural networks can implement highly complex functions, they might be able to guess the correct key without needing this information, and consequently, bypassing masking [32].

This is remarkable since, with classical TAs, when the mask value is unknown to the attacker during the profiling step, the leakages associated with a key follow a multimodal distribution. This leads to assumption errors whether the adversary exploits Gaussian template attacks, as confirmed in [23]. For this reason, some previous works apply TAs combined with second-order techniques and template-based DPA attacks with extra calculation considering the masks to succeed [28]. When the attacker has such strong capabilities (i.e. knows the key and the masks during profiling), it is considered a worst-case scenario [1].

Nonetheless, as mentioned above, if masking is not properly implemented the implementation can be vulnerable to first-order attacks: It is important to ensure that the two parts of the same secret (mask and masked intermediate value) are not handled too closely [41]. For instance, authors in [50]

claim that when the mask leakage is included in the observation time window, (first-order) TAs can relate the dependence between the mask and the masked variable leakage. Many other related works perform TAs over masked implementation without mask information [3, 21, 39, 40]. This allows for a more "realistic" (real-world) attack.

To the best of our knowledge, most of the results obtained with DL perform the attack in a weak setting (mask leakage in the attacked window and/or unintended interactions). This is partly because most of the results on DL-based SCAs are based on the ASCAD [3] dataset, in which there exist mask leakage in the targeted window, indicating possible unintended interactions. Thus, it is not clear that state-of-the-art CNNs have any advantage over TAs in these conditions, since both can circumvent masking. It is also unclear whether the attack works because of the presence of mask leakage or unintended interactions.

In any case, the authors of [39], published a dataset containing traces from masking implementations with and without mask leakage, but they only use EDA-based attacks. In this paper, we perform DL-based attacks on that dataset trying to determine whether DL-based attacks can bypass masking on both conditions or, on the contrary, current CNNs do not have any advantage in this scenario.[3]

3.3 Portability of profiling attacks

The original concept of profiling attacks comprises having two devices: the targeted device and an identical hardware copy that we can entirely control for building the leakage model. The idea is that even if the attacker has limited control of the target device, he can still carry out the attack.

However, although it is not a very real use case, researchers usually employ traces from the same device to perform both profiling and attack phases [3, 9, 21, 23, 36]. The problem is that, in practice, applying a power consumption model built with one device to a different hardware copy (concept typically called portability) is not trivial. In real life, variations happen between experimental setups and "identical" devices, as some previous works underline [4, 12, 15, 37]. There are two main reasons. One is that slight differences in the construction of a device or ageing can cause behavioral deviations between devices. The second is that subtle changes between experimental setups may provoke the same phenomenon. We are talking about environmental changes, glitches, interferences caused by communication interfaces, resonance due to LC and RC oscillators, the influence of the past state, variations in the magnetic field penetration, etc.

To the best of our knowledge, not many works on the topic take portability into account. In [15] Elaabid et al. introduce the portability issue and proposes waveform realignment and acquisition campaigns normalization to improve the performance in these conditions. Choudary et al. also studied portability and performed template attacks on different copies of the same device [11, 12]. Some other papers have employed different copies of the same device to perform template attacks [8, 19, 22, 40], but they are a minority. Recently, Bronchain et al. estimated the attack complexity of an adversary that profiles one device and attacks another device [5]. They compute the ratio between the perceived information(PI) when attacking a different device and the PI when attacking the same profiled device. Besides, several ML-based attacks have recently emerged that take portability into account. Bhasin et al. makes a comparison between different machine learning techniques using portable profiling attacks in [4]. In [13, 14, 20] authors successfully perform several DL-based cross-device SCA attacks. They also propose several preprocessing steps to improve the cross-device performance. In [47], authors propose an approach that uses meta-transfer learning to transfer DL networks between target devices by extracting information from a profiling device even using different SCA sources. Other DL-based approaches use transfer learning [43] or unsupervised domain adaptation [7] to perform cross-device attacks. However, these DL approaches are quite complex.

In this paper, we show how EDA-based PAs represent a simple alternative to DL-based attacks that can reach (and even beat) their performance in a more straightforward way (see Sect. 4.3), even in this portable scenario.

4 Experimental comparison

In this section, we compare the performance of EDAs and DL-based attacks on two open datasets: ASCAD [3] and AES_RA [39]. In addition, we also perform experiments considering the portability of the obtained models in our newly introduced AES_PTv2 dataset.[4]

4.1 ASCAD random keys

ASCAD [3] was the early open database for DL-SCA, and has become a standard for experimentation with DL in SCA, with many papers using it appearing every year [38, 39, 44, 48]. The DUT in this data set is an 8-Bit AVR microcontroller (ATmega8515), and includes EM emanation traces of the device implementing a masked AES-128 cipher [16]. The dataset is divided into two parts, fixed key and variable key. Although many works employ the fixed key version because it is an easier problem, for this comparison, we focus on the variable key subset because it is a more challenging and

[3] Note that a second-order attack (combining the leakage of two bytes of the key at a time to remove the mask) is feasible in both situations [39].

[4] AES_PTv2 Dataset: https://github.com/urioja/AESPTv2.

Table 1 Top results with PAs on ASCAD random

ASCAD random	[31] Best CNN	[45] Best CNN	[38] Best CNN	[38] Best CNN (RS)	[39] Best EDA
Trainable Param.	N/A	2 076 744	70 492	3 298	1 400
$\bar{Q}_{t_{ge}}$	105	1 568	490	1 018	150

realistic experiment. The traces represent a window of 1400 relevant samples per trace, corresponding to the third byte of the first round masked S-Box operation. As the sensitive intermediate value it is common to use an S-box output:

$$Y^{(i)}\left(k^*\right) = \text{Sbox}[P_3^{(i)} \oplus k^*]$$

For this dataset there exist published papers using both DL [31, 38, 45] and EDAs [39]. Thus, we have focused on these works without performing additional experiments, allowing an objective and unbiased view. Table 1 summarizes the best published attacks against ASCAD, in terms of trainable parameters and ge. As usual in the field, for ge we utilize $\bar{Q}_{t_{ge}}$, or the average number of traces needed to obtain a ge of 0.

Overall, we can see how despite being easier to execute, EDA-based PAs are able to provide better results (smaller $\bar{Q}_{t_{ge}}$) of than most CNNs on ASCAD (random key), being a more practical and simpler option for evaluators.

In terms of trainable parameters, the EDA-Based TA is also less demanding (one per time sample). Note that apart from the number of trainable parameters, with DL there is a huge number of possible architectures and hence hyperparameters to select and adjust. Conversely, the EDA-based approach has a far narrower number of parameters to tune (# iterations, population size and evaluation function). Finally, the complexity of the "training" itself is again lighter in an EDA-based attack: The time complexity of the EDA (UMDA) is $O(n)$ [40], whereas the time complexity of a backpropagation algorithm is much larger ($O(n^5)$ according to [17]).

4.2 AES_RA

AES_RA[5] was introduced in [39] and contains traces from two distinct embedded systems which use microcontrollers from the same family (Piñata training board from Riscure[6] and STM32F411E-DISCO development board[7]).

With each device, authors acquire traces from three AES implementations: an unprotected software AES implementation and two masking schemes, resulting in six different setups. Thus, this dataset is divided into two parts: power consumption traces from the Piñata board and capacitor EM power traces from the STM32F411E-Discovery Board, whereas power traces correspond to clean (noise-free) measurements, EM traces represent a more challenging problem (due to noise).

Table 2 Considered DL architectures

Name	Architecture
CNN1 [3]	C(64,11,2), P(2,2), C(128,11,1), P(2,2), C(256,11,1), P(2,2), C(512,11,1), P(2,2), C(512,11,1), P(2,2), FLAT, FC(4096), FC(4096), SM(256)
CNN2 [48]	C(4,1,1), BN, P(2,2), FLAT, FC(10), FC(10), SM(256)
CNN3 [30]	C(4,1,1), BN, P(2,2), FLAT, FC(10), FC(10), FC(10), SM(256)
CNN4 [38]	C(128,3,1), P(75,75), FLAT, FC(30), FC(2), SM(256)
CNN5 [38]	C(4,1,1), P(100,75), FLAT, FC(30), FC(10), FC(2), SM(256)
CNN6 [38]	C(2,50,1), BN, P(50,2), C(16,3,1), P(25,7), C(64,25,1), P(7,7),C(64,3,1), BN, P(4,4), GAP, FC(20), FC(15), FC(4), SM(256)
CNN7 [38]	C(4,50,1), P(50,4), C(2,50,1), P(2,2), C(2,50,1), P(50,50), GAP, FC(4), SM(256)

Regarding the protected AES implementations, the authors show how masked scheme 1 (MS1 from now on) is completely weak against (first-order) PAs, as the mask leaks in the same time window as the masked intermediate value (SBox output). Conversely, masked scheme 2 (MS2) contains only intermediate value leakage in the targeted time window, and hence, this implementation is robust against (first-order) PAs. For more details, we refer to the original paper [39].

In this work, we perform DL-based experiments on the dataset, confirming that the same conclusions are obtained as in [39] with EDAs on the different masking schemes and configurations.

4.2.1 Experimental configuration

We have performed several DL-based attacks on each protected implementation using some (pre-defined) CNN architectures. These architectures were not created for this specific dataset, and hence, good outcomes are not guaranteed. Nevertheless, all were designed for ASCAD, which represents a problem analogous to MS1 (mask leakage on the targeted

[5] AES_RA Dataset: https://github.com/AES-RA/AES_RA.

[6] Piñata board brochure: https://www.riscure.com/uploads/2017/07/pi%C3%B1ata_board_brochure.pdf.

[7] STM32F411VET6 Datasheet: https://www.alldatasheet.com/datasheet-pdf/pdf/929991/STMICROELECTRONICS/STM32F411VET6.html.

window). Thus, some CNNs shall succeed, especially considering that MS1 is particularly weak on Piñata (secret key can be recovered with about 5 traces using TAs). This also gives us a clue about how difficult it is to attack a similar dataset with a pre-defined architecture. Regarding trainable parameters, we have selected complex and simple CNNs in order to have a wide range of results (see Table 3).

More precisely, we have used the architecture for ASCAD random key suggested in the original paper [3], four CNNs from [38], the architecture from [48] for ASCAD fixed key and an improved version introduced in [30]:

- CNN1: From the original paper [3].
- CNN2: From [48], for ASCAD Fixed.
- CNN3: From [30], for ASCAD Fixed.
- CNN4: From [38], ASCAD Random (HW)
- CNN5: From [38], ASCAD Random RS (HW)
- CNN6: From [38], ASCAD Fixed (HW)
- CNN7: From [38], ASCAD Fixed RS (HW)

For all of them, we have used the hyper-parameters as in the original articles, except for the number of epochs. We have repeated the experiments with 50 and 75 epochs to ensure that poor results are not obtained because of underfitting/overfitting. Table 2 summarizes the DL architectures employed for our experiments. We follow the same notation as in [38]:

- Convolutional layer: C(filters, kernel_size, strides)
- Batch normalization: BN
- Average pooling: P(size, stride)
- Global average pooling: GAP
- Flatten: FLAT
- Fully-connected layer: FC(size)
- SoftMax: SM(classes)

Regarding the number of training traces, we are using the same number as in [39]. That is, 20,000 and 50,000 for pinata and 50,000 and 100,000 for the STM32F4 board (MS1 and MS2, respectively). The rest of hyperparameters are constant for all CNNs:

- Pooling type: average pooling
- SoftMax initializer: glorot uniform
- Initializer for other layers: he uniform
- Activation function: SeLU
- Optimizer: adam
- Train Epochs: 50 and 75
- Learning rate: one cycle policy

4.2.2 Experimental results

Figures 3, 4, 5 and 6 show the averaged ge of the correct key byte for all CNNs and setups (for 50 and 75 epochs respectively), and also the results using EDA-based PAs. This averaged ge represents the average of 10 attacks using the same model, cross-validated to avoid bias. Also, Table 3 shows the number of trainable parameters of each architecture, and the $\bar{Q}_{t_{ge}}$ of successful attacks. Note that although MS1 is especially weak on Piñata, some neural networks could not disclose the key.

In addition, the results with DL-based SCA are more variable in general. To show this, Figs. 3, 4, 5 and 6 also include Box Plots. These plots represent the variation of the final ge values of each one of the 10 attacks (for each model). In them, the dispersion and the median of the final values can be easily identified, besides clearly distinguishing whether there are outliers. This helps us to determine how precise is the model.

4.3 AES_PTv2

Recently, the authors of the AES_RA dataset have extended it with traces from different hardware copies of one of the targeted devices (STM32F411-DISCO board[8]), creating the AES_PTv2 dataset.[9] Thus, we employ this improved version of the dataset to compare the performance of EDA and DL approaches in a portable scenario (i.e. building the model with one device and attacking another "identical" copy). The dataset includes traces from four STM32F4 boards (called D1, D2, D3, and D4 from now on), with the same masking schemes explained above (MS1 and MS2). In other words, the same acquisition process employed with the STM32F4 board in AES_RA dataset has been repeated with three additional copies of the board. Details about the signal acquisition process and the dataset organization can be found in the AES_PTv2 GitHub.[10]

4.3.1 Experimental configuration

We repeat the same experiments as in the previous section, but in a more realistic setup. In other words, we perform the same battery of attacks, but profiling in D1 and performing the attack on D1, D2, D3 and D4. We focus on MS1, as long as is the scheme weak against profiling attacks.

[8] STM32F411VET6 Datasheet: https://www.alldatasheet.com/datasheet-pdf/pdf/929991/STMICROELECTRONICS/STM32F411VET6.html.

[9] AES_PTv2 Dataset: https://github.com/urioja/AESPTv2.

[10] AES_PTv2 Dataset: https://github.com/urioja/AESPTv2.

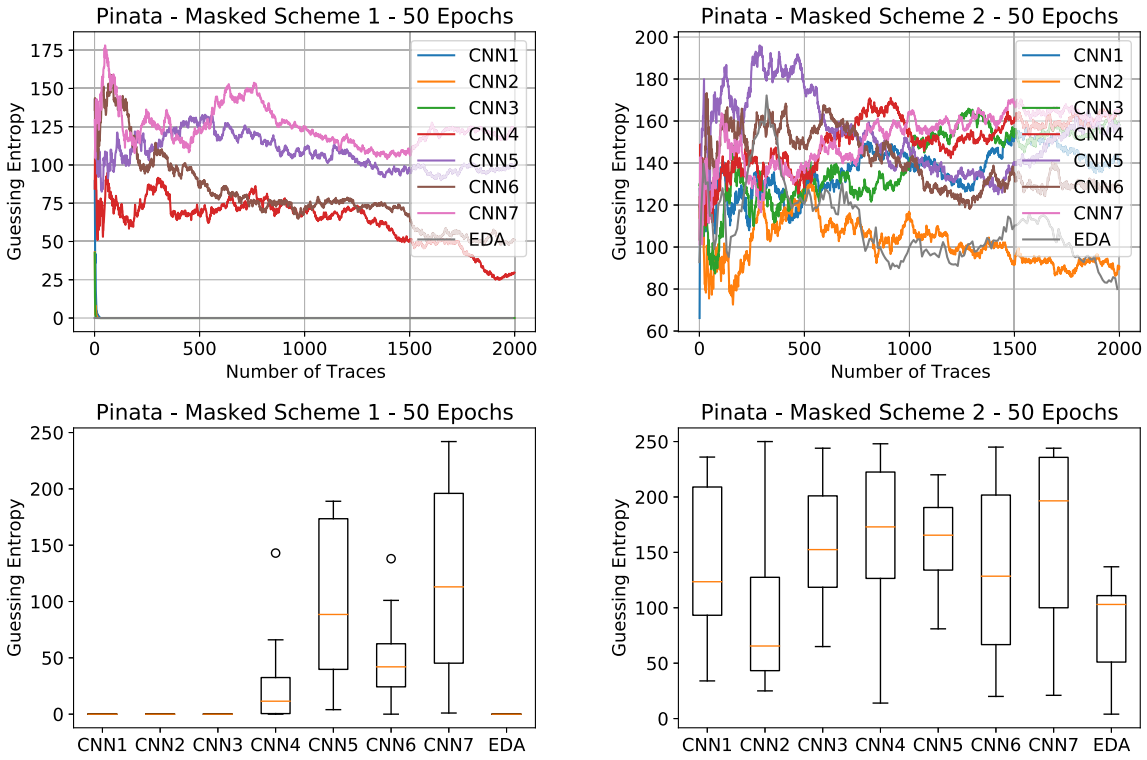

Fig. 3 Experimental results (averaged *ge* and box plot of final *ge* values) on AES_RA - Piñata [50 Epochs]

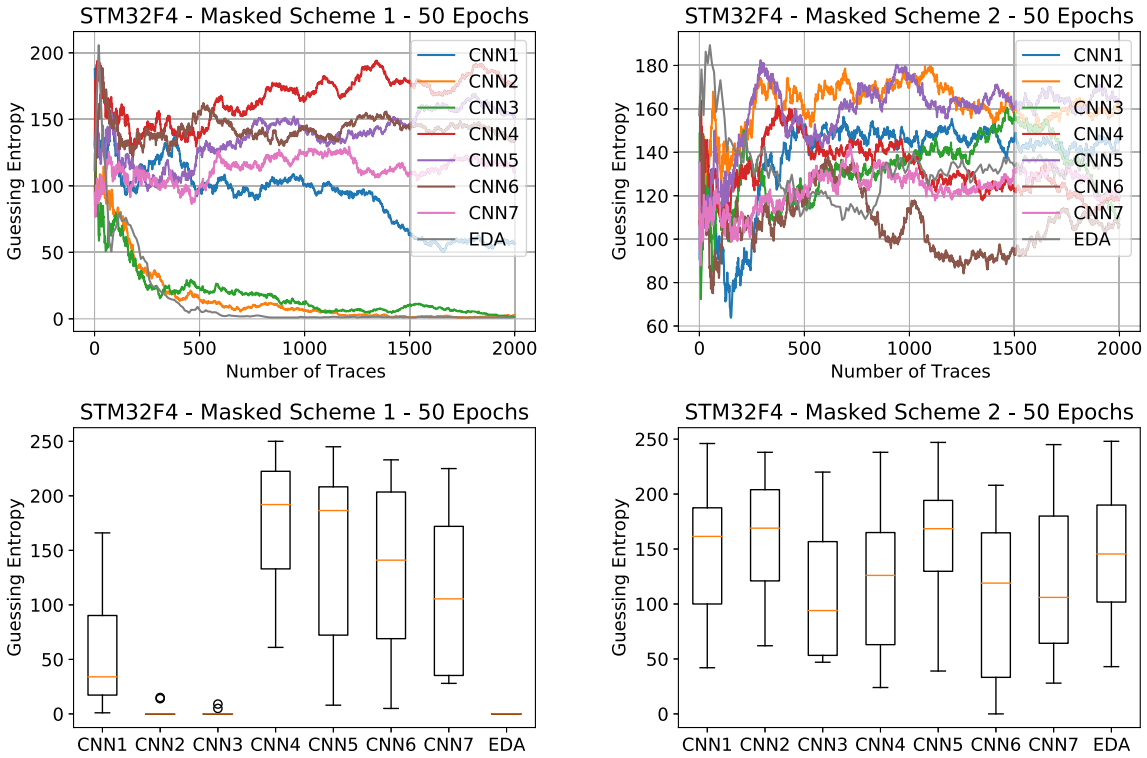

Fig. 4 Experimental results (averaged *ge* and box plot of final *ge* values) on AES_RA - STM32F4 [50 Epochs]

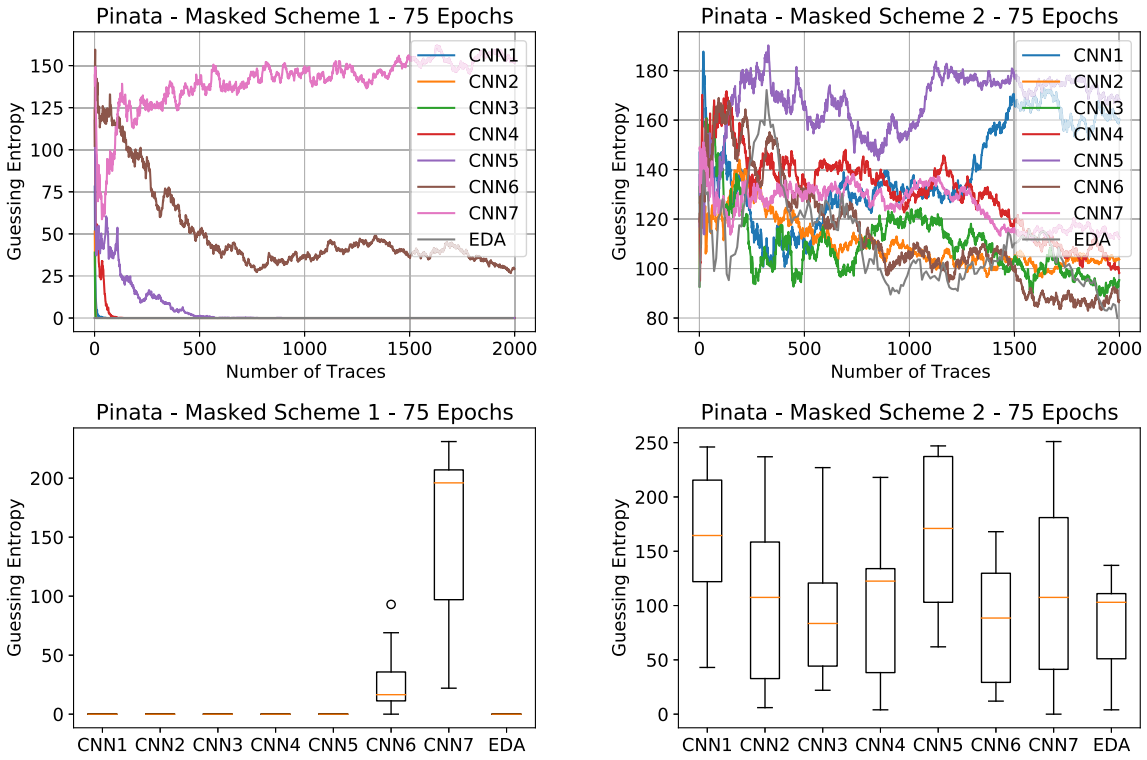

Fig. 5 Experimental results (averaged *ge* and box plot of final *ge* values) on AES_RA - Piñata [75 Epochs]

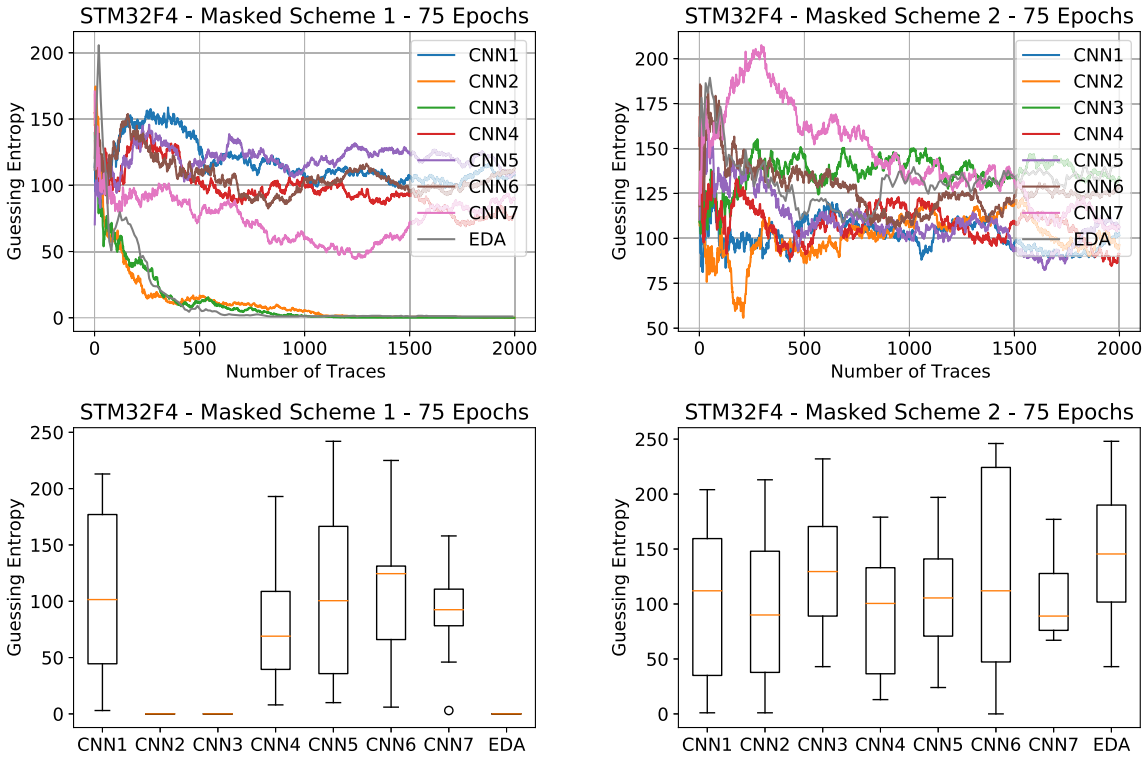

Fig. 6 Experimental results (averaged *ge* and box plot of final *ge* values) on AES_RA - STM32F4 [75 Epochs]

Table 3 Top results with PAs on AES_RA

AES_RA	CNN1	CNN2	CNN3	CNN4	CNN5	CNN6	CNN7	EDA
Trainable Param.	70 846 848	32 960	33 070	78 172	3 418	3 471	1 394	1 500
$\bar{Q}_{t_{ge}}$(Piñata-MS1)	27	8	6	113	34	–	–	5
$\bar{Q}_{t_{ge}}$(STM32-MS1)	–	1 867	1 222	–	–	–	–	800

4.3.2 Experimental results

In Fig. 7 the guessing entropy of each one of the attacks can be observed. Each subplot represents the attack over the four devices using one of the models (CNN1, CNN2, CNN3, CNN4, CNN5, CNN6, CNN7 and EDAs). We only represent the attacks with 75 epochs, as results were slightly better (as in the previous use case).

The results are in line with those of the non-portable scenario. Only CNN2 and CNN3 were able to recover the secret key. The rest of CNNs show a poor performance. Conversely, the EDA-based TA is able to obtain slightly better results than CNN2 and CNN3. This can be evidenced more clearly in Table 4. Note that, although we have not represented it due to space restrictions, similar results are obtained by using another copy to perform the profiling (e.g. profiling in D2 and attacking D1, D2, D3 and D4).

In conclusion, we can observe how EDAs and (some of the considered CNNs) perform well in a portable scenario. This shows how, even in a more challenging setup, the EDA-based approach is able to reach (and even beat) the performance of state-of-the-art CNNs.

5 Discussion

In terms of ge, both methods can achieve similar outcomes. Nevertheless, it should be noticed that we have only been able to achieve the results reached with EDAs with some CNNs. In addition, as the Box Plots show, the outcomes are in general more variable when using CNNs in our experiments. Also note that although these CNNs were designed for a similar dataset, they usually require human engineering to succeed.

Our comparison shows that EDA-based attacks have some advantages over the DL approach, mainly:

- Simplicity: fewer parameters to tune (i.e. hyperparameters), trainable parameters, and lighter algorithmic complexity.
- Generalized solution: no need to re-tune the hyperparameters when targeting new devices, implementations, datasets, etc.

On the other hand, we have also seen how a well trained DL model can obtain results equivalent to those of EDA-based TAs. In addition, training a neural network today is a highly optimized process that can be performed relatively quickly (thanks to the use of parallelization using GPUs) while EDA-based attacks have a long way to go to reach these levels of optimization. However, EDA-based PAs are far less complex than DL in terms of algorithmic complexity. This, together with the fact that several ways of optimizing them have already been identified [40] (e.g. attack parallelization, attack computation optimization, etc.), means that they could become as efficient or even more efficient than DL. Conversely, TAs work with a Gaussian assumption, whereas ML models do not assume the probability density function of the data [24]. Nevertheless, note that ML models could also be employed with the EDA approach, although authors chose TA for demonstrating their approach in [40].

Besides, a known disadvantage of TAs, and hence EDA-based TAs, versus the DL branch is that CNNs deal better with noise and small misalignments in the signal [6, 32]. Conversely, since the EDA approach is an automatic optimization of TAs it suffers from the same disadvantages as the TAs themselves, being commonly necessary to realign the traces to achieve success. This can be observed in [39] where an EDA-based TA is applied on misaligned traces, showing very poor results. In any case, it is important to note that in most of the related works using DL, a small window of clean and aligned traces is used [33]. In fact, several works show that DL, far from bypassing the need to filter the noise and align the signals, simply mitigates it, since the results are greatly improved with preprocessing [29, 49]. In conclusion, although DL may offer certain minor advantages over (EDA-based) TAs in terms of preprocessing, in terms of guessing entropy, both methods can obtain equivalent results, which can also be observed in some related works [3, 21, 33].

6 Conclusions and future work

From this analysis, we draw the same conclusions about MS1 and MS2 as in [39]: we recovered the secret key on MS1 using several predefined CNN architectures but not on MS2. This shows that both TAs and DL can circumvent masking in schemes like ASCAD or MS1, as some previous works have also shown [3, 21, 50]. However, determining whether more complex CNNs or (EDA-based) PAs can actually bypass masking under unfavourable conditions (no mask leakage

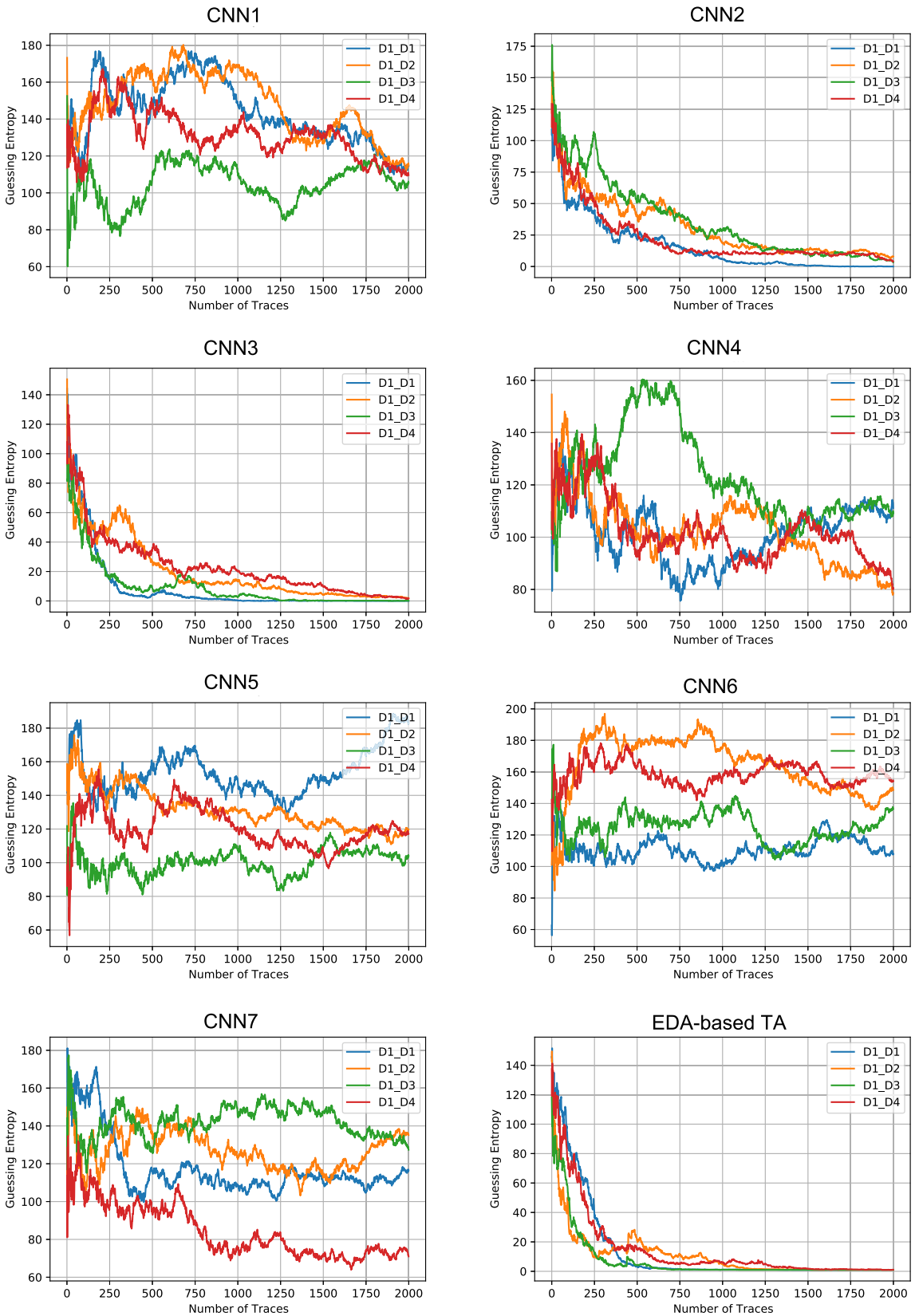

Fig. 7 Experimental results (averaged ge) on AES_PTv2 [75 Epochs]

Table 4 Top results with PAs on AES_PTv2

AES_PTv2	CNN1	CNN2	CNN3	CNN4	CNN5	CNN6	CNN7	EDA
Trainable Param.	70846 848	32960	33070	78172	3418	3471	1394	1500
$\bar{Q}_{t_{ge}}$(D1)	–	1 600	1 000	–	–	–	–	800
$\bar{Q}_{t_{ge}}$(D2)	–	>2 000	>2 000	–	–	–	–	1250
$\bar{Q}_{t_{ge}}$(D3)	–	>2 000	1 250	–	–	–	–	1500
$\bar{Q}_{t_{ge}}$(D4)	–	>2 000	>2 000	–	–	–	–	800

in the attacked window and/or no unintended interactions) is beyond the scope of this paper, we believe it is an interesting research question for future work. Besides, we have also shown how EDAs and (some of the considered) CNNs can perform well in a portable scenario. In addition, although investigating the influence of assuming the probability distribution of the data on the results is out of the scope of this paper, we believe it is an interesting avenue for further work.

Concluding, as we intend to show in this paper, both alternatives can provide similar results in terms of ge, with EDA-based attacks being a more straightforward alternative that can represent a very efficient and interpretable shortcut for evaluators.

Acknowledgements We are thankful to the anonymous referees for their constructive comments that help us to improve the paper. This work was partially supported by the *Ayudas Cervera para Centros Tecnológicos* grant of the Spanish Centre for the Development of Industrial Technology (CDTI) under the project EGIDA (CER-20191012), and by the Basque Country Government under the ELKARTEK program, project REMEDY - Real Time Control And Embedded Security (KK-2021/00091).

Author Contributions UR: conceptualization, methodology, software, formal analysis, validation, visualization, investigation, writing original draft, Writing, review and editing. LB: supervision, writing review and editing. IA: supervision, writing review and editing. JLF: advicing, software, writing review and editing.

Declarations

Conflicts of interest We have a conflict of interest to disclose, as one of the guest editors of the special issue shares an affiliation with some authors (Stjepan Picek, Radboud University, The Netherlands).

References

1. Azouaoui, M., Bellizia, D., Buhan, I., Debande, N., Duval, S., Giraud, C., Jaulmes, E., Koeune, F., Oswald, E., Standaert, F.-X., Whitnall, C.: A systematic appraisal of side channel evaluation strategies. IACR Cryptol. ePrint Arch (2020)
2. Balasch, J., Gierlichs, B., Grosso, V., Reparaz, O., Standaert, F.-X.: On the cost of lazy engineering for masked software implementations, Vol 8968, pp. 64–81. Joye, Marc, Springer (2014)
3. Benadjila, Ryad, Prouff, Emmanuel, Strullu, Rémi., Cagli, Eleonora, Dumas, Cécile.: Deep learning for side-channel analysis and introduction to ASCAD database. J. Cryptogr. Eng. **10**, 06 (2020)
4. Bhasin, S., Chattopadhyay, A., Heuser, A., Jap, D., Picek, S., Shrivastwa, R.R.: Mind the portability: a warriors guide through realistic profiled side-channel analysis. In: NDSS Symposium (2020)
5. Bronchain, Olivier, Standaert, François-Xavier.: Breaking masked implementations with many shares on 32-bit software platforms: or when the security order does not matter. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(3), 202–234 (2021)
6. Cagli, E., Dumas, C., Prouff, E.: Convolutional neural networks with data augmentation against jitter-based countermeasures. In: Fischer, W., Homma, N. (eds.) CHES 2017, pp. 45–68, Springer (2017)
7. Cao, Pei, Zhang, Chi, Xiangjun, Lu., Dawu, Gu.: Cross-device profiled side-channel attack with unsupervised domain adaptation. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**, 27–56 (2021)
8. Carbone, Mathieu, Conin, Vincent, Cornélie, Marie-Angela., Dassance, François, Dufresne, Guillaume, Dumas, Cécile., Prouff, Emmanuel, Venelli, Alexandre: Deep learning to evaluate secure RSA implementations. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(2), 132–161 (2019)
9. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Cryptographic hardware and embedded systems—CHES 2002, pp. 13–28. Springer (2002)
10. Chen, Z., Haider, S., Schaumont, P.: Side-channel leakage in masked circuits caused by higher-order circuit effects. In: Park, J. H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T., Yeo, S.-S. (eds.), Advances in information security and assurance, pp. 327–336, Berlin, Heidelberg (2009)
11. Choudary, M.O., Kuhn, M.G.: Efficient, portable template attacks. IEEE Trans. Inf. Forensics Secur. **13**(2), 490–501 (2018)
12. Choudary, O., Kuhn, M.G.: Template attacks on different devices. In: Constructive side-channel analysis and secure design, COSADE, pp. 179–198. Springer (2014)
13. Danial, J., Das, D., Golder, A., Ghosh, S., Raychowdhury, A., Sen, S.: EM-X-DL: efficient cross-device deep learning side-channel attack with noisy EM signatures. CoRR. arXiv:2011.06139 (2020)
14. Das, D., Golder, A., Danial, J., Ghosh, S.K., Raychowdhury, A., Sen, S.: X-deepsca: Cross-device deep learning side channel attack*. In: 2019 56th ACM/IEEE Design Automation Conference (DAC), pp 1–6 (2019)
15. Elaabid, Sylvain, Abdelazizand Guilley, M.: Portability of templates. J. Cryptogr. Eng. **2**(1), 63–74 (2012)
16. Federal Information Processing Standard. FIPS 197: Announcing the advanced encryption standard (AES) (2001). https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf (2001)
17. Kasper F.: Computational complexity of neural networks. https://kasperfred.com/series/introduction-to-neural-networks/computational-complexity-of-neural-networks (2020)
18. Gilmore, R., Hanley, N., O'Neill, M.: Neural network based attack on a masked implementation of AES. In: Proceedings of the 2015 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2015, pp. 106–111 (2015)

19. Gohr, A., Jacob, S., Schindler, W.: CHES 2018 side channel contest CTF–solution of the AES challenges. IACR Cryptol. ePrint Arch (2019)
20. Golder, Anupam, Das, Debayan, Danial, Josef, Ghosh, Santosh, Sen, Shreyas, Raychowdhury, Arijit: Practical approaches toward deep-learning-based cross-device power side-channel attack. IEEE Trans. Very Large Scale Integr. VLSI Syst. **27**(12), 2720–2733 (2019)
21. Kim, Jaehun, Picek, Stjepan, Heuser, Annelie, Bhasin, Shivam, Hanjalic, Alan: Make some noise unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(3), 148–179 (2019)
22. Kim, Taewon, Kim, Kwonyoup, Kim, Tae Hyun, Ryu, Sangryeol: AES wireless keyboard: Template attack for eavesdropping. In: Black Hat Asia, Singapore (2018)
23. Lerman, Liran, Poussier, Romain, Markowitch, Olivier, Standaert, François-Xavier.: Template attacks versus machine learning revisited and the curse of dimensionality in side-channel analysis: extended version. J. Cryptogr. Eng. **8**, 11 (2018)
24. Maghrebi, Houssem: Deep learning based side-channel attack: a new profiling methodology based on multi-label classification. IACR Cryptol. ePrint Arch (2020)
25. Maghrebi, H., Portigliatti, T., Prouff, E.: Breaking cryptographic implementations using deep learning techniques. In: SPACE 2016 (2016)
26. Mangard, S., Popp, T., Gammel, B.M.: Side-channel leakage of masked CMOS gates. In: Menezes, A. (ed.), Topics in Cryptology—CT-RSA 2005, pp. 351–365, Springer, Berlin (2005)
27. Martinasek, Z., Malina, L.: Comparison of profiling power analysis attacks using templates and multi-layer perceptron network (2014)
28. Oswald, E., Mangard, S.: Template attacks on masking-resistance is futile. In: CT-RSA 2007: Topics in Cryptology, vol. 4377, pp. 243–256 (2007)
29. Paguada, Servio, Batina, Lejla, Armendariz, Igor: Toward practical autoencoder-based side-channel analysis evaluations. Comput. Netw. **196**, 108230 (2021)
30. Paguada, S., Rioja, U., Armendariz, I.: Controlling the deep learning-based side-channel analysis: a way to leverage from heuristics. In: ACNS Workshops (2020)
31. Perin, Guilherme, Chmielewski, Lukasz, Picek, Stjepan: Strength in numbers: improving generalization with ensembles in machine learning-based profiled side-channel analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2020**(4), 337–364 (2020)
32. Perin, G., Ege, B.: Lowering the bar: deep learning for side-channel analysis (whitepaper). In: Proc. BlackHat, pp. 1–15 (2018)
33. Perin, Guilherme, Lichao, Wu., Picek, Stjepan: Exploring feature selection scenarios for deep learning-based side-channel analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2022**(4), 828–861 (2022)
34. Picek, S., Heuser, A., Jovic, A., Batina, L.: A systematic evaluation of profiling through focused feature selection. IEEE Trans. Very Large Scale Integr. (VLSI) Syst., pp. 1–14 (2019)
35. Picek, S., Samiotis, I. P., Kim, J., Heuser, A., Bhasin, S., Legay, A.: On the performance of convolutional neural networks for side-channel analysis. In: Chattopadhyay, A., Rebeiro, C., Yarom, Y. (eds.) Security, Privacy, and Applied Cryptography Engineering, pp. 157–176, Springer, Cham (2018)
36. Rechberger, C., Oswald, E.: Practical template attacks. In: Lim, C. H., Yung, M. (eds.) Information Security Applications, pp. 440–456, Springer, Berlin (2005)
37. Renauld, M., Standaert, F.-X., Veyrat-Charvillon, N., Kamel, D., Flandre, D.: A formal study of power variability issues and side-channel attacks for nanoscale devices. In: Advances in Cryptology—EUROCRYPT 2011, pp. 109–128. Springer, Heidelberg (2011)
38. Rijsdijk, Jorai, Lichao, Wu., Perin, Guilherme, Picek, Stjepan: Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(3), 677–707 (2021)
39. Rioja, U., Batina, L., Armendariz, I., Flores, J.L.: Towards human dependency elimination: AI approach to SCA robustness assessment. IEEE Trans. Inf. Forensics Secur. (2022)
40. Rioja, Unai, Batina, Lejla, Flores, Jose Luis, Armendariz, Igor: Auto-tune pois: estimation of distribution algorithms for efficient side-channel analysis. Comput. Netw. **198**, 108405 (2021)
41. Shelton, M., Samwel, N., Batina, L., Regazzoni, F., Wagner, M., Yarom, Y.: Rosita: towards automatic elimination of power-analysis leakage in ciphers. In: NDSS Symposium (2021)
42. Standaert, F.-X., Archambeau, C.: Using subspace-based template attacks to compare and combine power and electromagnetic information leakages. In: Oswald, E., Rohatgi, P. (eds.) Cryptographic Hardw Embed Syst—CHES 2008, pp. 411–425. Springer, Heidelberg (2008)
43. Thapar, D., Alam, M., Mukhopadhyay, D.: Deep learning assisted cross-family profiled side-channel attacks using transfer learning. In: 2021 22nd International Symposium on Quality Electronic Design (ISQED), pp. 178–185 (2021)
44. Wouters, Lennert, Arribas, Victor, Gierlichs, Benedikt, Preneel, Bart: Revisiting a methodology for efficient CNN architectures in profiling attacks. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2020**(3), 147–168 (2020)
45. Wu, L., Perin, G., Picek, S.: I choose you: automated hyperparameter tuning for deep learning-based side-channel analysis. IACR Cryptol. ePrint Arch., p. 1293 (2020)
46. Yang, S., Zhou, Y., Liu, J., Chen, D.: Back propagation neural network based leakage characterization for practical security analysis of cryptographic implementations, pp. 169–185 (2011)
47. Yu, H., Shan, H., Panoff, M., Jin, Y.: Cross-device profiled side-channel attacks using meta-transfer learning. In: 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 703–708 (2021)
48. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Methodology for efficient CNN architectures in profiling attacks. IACR Trans. Cryptogr. Hardw. Embed. Syst., Vol. 2020 (2019)
49. Zhou, Yuanyuan, Standaert, François-Xavier.: Deep learning mitigates but does not annihilate the need of aligned traces and a generalized ResNet model for side-channel attacks. J. Cryptogr. Eng. **10**, 04 (2020)
50. Zotkin, Y., Olivier, F., Bourbao, E.: Deep learning vs template attacks in front of fundamental targets: experimental study. IACR Cryptol. ePrint Arch., pp. 1213 (2018)