Enhancing Scale and Shift Invariance in Deep Learning-based Side-channel Attacks through Equivariant Convolutional Neural Networks

David Perez[1,4], Sengim Karayalcin[2], Stjepan Picek[1,3] and Servio Paguada[4]

[1] Radboud University, Netherlands
[2] Leiden University, Netherlands
[3] University of Zagreb, Republic of Croatia
[4] Ikerlan, Spain

Abstract. Deep learning-based side-channel analysis (DLSCA) has demonstrated remarkable performance over the past few years. Even with limited preprocessing and feature engineering, DLSCA is capable of breaking protected targets, sometimes requiring only a single attack trace. In the DLSCA context, the commonly investigated countermeasures are Boolean masking and desynchronization. While the exact mechanisms of how DLSCA breaks masking are less understood, the core idea behind handling desynchronization is simple. Convolutional neural networks (CNNs) are shift invariant, allowing them to overcome desynchronization. However, considering the importance and practicality of desynchronization countermeasures, we know remarkably little about the limits of CNNs or how to enhance their capabilities when dealing with desynchronization.

In this work, we begin with the theoretical foundations of shift and temporal scale equivariance. Afterward, we build a neural network model allowing such equivariance and test it against several commonly considered targets. Our results demonstrate that equivariant CNNs are robust, easy to design, and achieve excellent attack performance. More precisely, we showcase how such a simple model can even outperform recent transformer-based neural networks. Finally, we demonstrate the practical relevance of scale equivariance by showing how an equivariant CNN can learn leakage from a device operating at one clock frequency and generalize to a device with a different clock frequency, a result not previously demonstrated in DLSCA.

Keywords: Side-channel Analysis · Invariance · Equivariance · Neural Networks

1 Introduction

Side-channel analysis (SCA) has received significant attention from the research community in the last three decades. In such attacks, the relationship between secret-dependent computations and (physical) leakages (e.g., timing and power consumption) from cryptographic implementations is exploited to extract secret information [Koc96, KJJ99]. Today, there are many variations of side-channel attacks, but a common distinction is between direct and profiling attacks. In direct attacks, one could need millions of traces from the device under attack to obtain secret information. Profiling attacks, while depending on a stronger threat model, can also result in significantly more powerful attack performance than direct attacks [CRR02]. Indeed, in profiling attacks, if the attacker makes a good profile

E-mail: david.perezperez@ru.nl (David Perez), karayalcins@vuw.leidenuniv.nl (Sengim Karayalcin), stjepan.picek@ru.nl (Stjepan Picek), slpaguada@ikerlan.es (Servio Paguada)

(template, machine learning model), the attack could require only a single trace from the device under attack. More recently, the SCA community has devoted significant efforts to profiling attacks based on machine (deep) learning (see [MPP16, ZBHV20, PWP22]), as such attacks could deliver optimal attack performance[1] with only limited effort in mounting the attack. In deep learning-based SCA, the most common approach today is to use a convolutional neural network (CNN) to build a profiling model [PPM+23]. One of the main advantages of CNNs over other deep learning architectures (e.g., multilayer perceptrons) is the fact that they are shift invariant, which allows them to (partially) defeat the desynchronization countermeasures. As such, the notion of shift invariance is often mentioned in the DLSCA literature [HSAM22, HCM24, KWPP24]. Interestingly, very little effort has been made to understand what this notion truly means in the context of SCA.

Clearly, using CNNs without further consideration of invariance (and equivariance) is insufficient. Numerous studies have shown that larger desynchronization levels are more difficult to overcome, and sufficiently large desynchronization levels can also prevent the attack [CDP17, PPM+23, HSAM22]. The main approach to address this issue is to use data augmentation techniques to induce robustness during training [CDP17], but this limits the invariance to what the levels of desynchronization the model has seen during training [KWPP24]. Another direction to resolve this issue is to consider more recent developments in deep learning, where transformer-based architectures, combined with data augmentation, can handle longer measurements and larger desynchronization [HSAM22, HCM24]. In this work, we aim to determine whether understanding the notions of invariance and equivariance with respect to SCA can help design adapted CNNs that are robust to such transformations.

We begin by establishing the relevant theoretical foundations of invariance and equivariance in the context of SCA. We note that while these aspects are known in the general literature, to our knowledge, they have not been properly discussed in the DLSCA context. Afterward, we design a neural network architecture that is equivariant, and we demonstrate its robustness to various transformations. More precisely, we conduct an extensive experimental analysis on five datasets and three neural network models, showcasing how an equivariant convolutional neural network can outperform even a transformer network. Our main contributions are:

- We establish a DLSCA design approach by utilizing equivariant and invariant operations with respect to a predetermined set of transformations against which we want the models to be robust. Based on this approach, we propose a CNN that is equivariant to relevant transformations, which we denote by EquivSCA.[2] More precisely, we propose a convolutional model that is robust against shift changes and resilient to signal temporal scale changes.
- We conduct experimental analysis on five datasets and three neural network models, and demonstrate that equivariant CNN can outperform even recent transformer-based architectures on datasets ASCADv1, AES_RD, and ESHARD. We also tested the robustness of the models against desynchronization and time dilation transformations, showing how EquivSCA was generally the best-performing model.
- By experimentally comparing EquivSCA and other CNN-based models against traces of varying scales, we show that EquivSCA's ability to extract multi-scale features leads to more effective attacks.
- To assess the robustness of models to changes in device clock-frequency, we create datasets Nucleo_96MHz and Nucleo_168MHz[3] to be used by the SCA community. With those, we performed a transfer of learning experiment, showing how EquivSCA

[1] Optimal performance is breaking a target with a single attack trace.

[2] The EquivSCA code can be found at https://shorturl.at/lxW7K.

[3] Nucleo datasets can be found at https://shorturl.at/iV1zH

was able to attack traces generated with a clock frequency different from the one involved in its training.

2 Group Equivariant Deep Learning

Group equivariant deep learning is a branch of geometric deep learning [BBCV21] initiated by Cohen and Welling [CW16], trying to define convolutional models that are naturally robust to transformations such as translations, rotations, and reflections of images. This was achieved by using models composed of a sequence of equivariant layers, followed by an invariant pooling operation with respect to the transformations that the model should take into account.

These transformations are defined as a group $(G, *)$, where G is a set and $*$ is its binary operation under which the elements of G are closed. The equivariance of a mapping $f : X \to Y$ is formalized using a group G and two G-sets[4] X and Y. A mapping f is said to be equivariant if and only if

$$f(g * x) = g' * f(x), \tag{1}$$

for all $x \in X$ and $g, g' \in G$ that fulfill $g = g'$. When conditions are met, the function is called G-equivariant. If g' is the identity, the function is G-invariant.

The function f is generally not injective when it is G-invariant. This occurs as a consequence of f being equivalent to the composition of mappings $\rho : X/G \to Y$ and natural map[5] $\eta_G : X \to X/G$, where X/G is the quotient space $\{G * x : x \in X\}$. The presence of η_G implies that the input space is compressed into the quotient space whose cardinality is always smaller than or equal to the cardinality of X. On the other hand, G-equivariant function f does not compress the input space as it produces independent outputs for x and $g \cdot x$, while maintaining a well-defined relationship between them.

Observe that desynchronization can be seen as a translation element $H_\theta \in G$, acting over an input f as $H_\theta \cdot f(x) = f(H_\theta^{-1} * x) = f(x - \theta)$, so architectures incorporating translation equivariant or invariant layers have shown advantages [BB21, LCY14] in developing models with built-in invariance to translations. $H_\theta \cdot f(x)$ represents the group action of H_θ over the function f. By using and extending these design ideas from group equivariant deep learning, models that are naturally robust to translations can be created.

Cohen and Welling [CW16] generalized the convolution operation with the concept of group convolution. A common convolution operation used in deep learning is defined as $(f \tilde{*} \psi)(y) = \int dx f(x)\psi(x - y)$, and it is known for being translation-equivariant [GBC16]. Notice that it can be rewritten using group theory, leading to the translation-group convolution, defined as $(f \tilde{*} \psi)(H_y) = \int d\mu(H_x)\ f(H_x)\psi(H_y^{-1} \circ H_x)$. This formula is obtained by realizing that $x = 0 + x \equiv H_x * 0$, where $H_x \in G$ is a translation transformation of size x. This leaves $f(x)$ as $f(0 + x) = f(H_x * 0)$. For simplicity, it can be rewritten as $f(H_x)$. Following this, we have that $\psi(x - y) = \psi(0 + x - y) = \psi((H_y^{-1} \circ H_x) * 0) = \psi(H_y^{-1} \circ H_x)$, where $H_y^{-1} \circ H_x$ is the composition of translations $H_x, H_y^{-1} \in G$. Finally, $dx \equiv d\mu(H_x)$, where $d\mu(H_x)$ is the Haar measure over the group of translations. This allows the definition of the integral in terms of translations.

With the rewritten convolution formula in terms of translations, it can be further generalized to using any element $g \in G$ in the convolution. The convolution operation defined over a general G is known as the group convolution, shown in Eq. (2), where $g, h \in G$.

$$(f \tilde{*} \psi)(g) = \int d\mu(h)\ f(h)\psi(g^{-1} \circ h). \tag{2}$$

[4] A G-set is a mathematical object consisting of a set S and a group action of G on S.

[5] A natural map (or canonical map) is a function that arises directly from the mathematical structure of the objects involved, typically mapping an element to its equivalence class under a given relation

2.1 Temporal-Scale Equivariance

Temporal-scale or time dilation transformations are elements $T_\alpha \in G$ that act on a function f as $T_\alpha \cdot f(x) = f(T_\alpha^{-1} * x) = f(\alpha^{-1}x)$, where $\alpha \in \mathbb{R}^+$ is a positive real value known as the time dilation factor. Changing the clock's frequency of a cryptographic device results in the same operations being performed in a shorter or longer amount of time. If $f(x)$ is the trace generated, it is related to its version obtained after changing the clock's frequency by $f(x) \sim f(\alpha^{-1}x) = T_\alpha \cdot f(x)$. This results in time dilations being theoretically equivalent to changing the clock's frequency. In this situation, a convolution is not enough to analyze traces that differ in the frequency used to generate them. Since the convolutions are not equivariant to time dilations [AW19], as shown in Eq. (5), a common CNN model will require training and data augmentation to build the time dilation robustness required to attack traces without depending on the clock's frequency used to generate them.

$$(T_\alpha \cdot f \tilde{*} \psi)(y) = \int dx\ T_\alpha \cdot f(x)\psi(x-y) = \int dx\ f(\alpha^{-1}x)\psi(x-y) \Rightarrow \tag{3}$$

$$\text{next, substitution is applied: } \alpha x' = x. \tag{4}$$

$$\Rightarrow \alpha \int dx' f(x')\psi\left(\alpha x' - y\right) \neq (f \tilde{*} \psi)(\alpha^{-1}y) = T_\alpha \cdot (f \tilde{*} \psi)(y). \tag{5}$$

Group equivariant deep learning community defines another way to approach this problem, through the use of time-scale equivariant layers [RBTH20, WW19, ANS+23, SMS21]. These models use a sequence of time dilation equivariant layers followed by a time dilation invariant layer. This ensures built-in robustness against time dilation transformations of the input. As changing the clock's frequency results in theoretically equivalent traces to the ones obtained after applying a time dilation transform, the equivariant model would have an advantage over the non-equivariant one.

Given that the input analyzed by the model will consist of 1-D temporal signals, two types of time dilation equivariant layers are needed: the Lifting and the Time-Scale group convolutions. Both of them have their origin in the group convolution defined in Section 2. The Lifting convolution (Λ) [RBTH20] is a mapping that takes a function as input and returns its convolved version. It is used as the first convolutional layer in the model, and defined as a time-scale equivariant layer that takes as input a trace $f(x)$ and lifts it to a higher-dimensional vector space that includes the convolved $f(x)$ along with the time dilated convolved versions of it. This results in an output with a time dimension and a time dilation dimension. It is defined as shown in Eq. (6).

$$\Lambda(f)(y,s) = \int dx f(x) \frac{1}{s} \psi\left(\frac{x-y}{s}\right). \tag{6}$$

In Eq. (6), s is the time dilation factor and $\frac{1}{s}$ has its origin in the Haar measure being invariant to time dilations T_s. Thus, $dx = d\mu(x) = d\mu(T_s * x) = d\mu(sx) = d(sx)/s = dx = d\mu(x)$. Notice that ψ represents the trainable kernels used in the layer.

The Lifting convolution is equivariant to time dilation transformations $T_\alpha \in G$ as shown in Eq. (9).

$$\Lambda(T_\alpha \cdot f)(y,s) = \int dx f(\alpha^{-1}x) \frac{1}{s} \psi\left(\frac{x-y}{s}\right) \Rightarrow \tag{7}$$

$$\text{when: } \alpha x' = x. \tag{8}$$

$$\Rightarrow \int dx' f(x') \frac{1}{\alpha^{-1}s} \psi\left(\frac{x' - \alpha^{-1}y}{\alpha^{-1}s}\right) = T_\alpha \cdot \Lambda(f)(y,s). \tag{9}$$

The Time-Scale group convolution (Ω) [RBTH20] is a mapping that takes a function as input and returns its convolved version. It differs from the Lifting convolution in that the input and output have both time dimension (x, y) and time dilation dimension (p, s). This contrasts with the Lifting convolution that takes as input an object with only a time dimension. Its formula is given in Eq. (10).

$$\Omega(f)(y,s) = \int dxdpf(x,p)\frac{1}{s^2}\psi\left(\frac{x-y}{s}, \frac{p}{s}\right). \tag{10}$$

In Eq. (10), similarly to the case of Lifting convolution, $\frac{1}{s^2}$ has its origin in the Haar measure being invariant to time dilations T_s. Then, $dxdp = d\mu(x,p) = d\mu(T_s * (x,p)) = d\mu(sx, sp) = d(sx)d(sp)/s^2 = dxdp = d\mu(x,p)$ where ψ represents the trainable kernels.

The Time-Scale group convolution is equivariant to time dilation transformations $T_\alpha \in G$ as shown in Eq. (13).

$$\Omega(T_\alpha \cdot f)(y,s) = \int dxdpf(\alpha^{-1}x, \alpha^{-1}p)\frac{1}{s^2}\psi\left(\frac{x-y}{s}, \frac{p}{s}\right) \Rightarrow \tag{11}$$

$$\text{next, substitutions are applied: } \alpha x' = x,\ \alpha p' = p. \tag{12}$$

$$\Rightarrow \int dx'dp'f(x',p')\frac{1}{(\alpha^{-1}s)^2}\psi\left(\frac{x'-\alpha^{-1}y}{\alpha^{-1}s}, \frac{p'}{s}\right) = T_\alpha \cdot \Omega(f)(y,s). \tag{13}$$

To be able to dilate with s, the kernels ψ used in the Lifting and Time-Scale group convolution, it is required that they are continuous. The Lifting convolution's kernel is defined as:

$$\psi(x) = \sum_i^{TK} w^i B(x - \tau^i). \tag{14}$$

The Time-Scale group convolution's kernel is defined as:

$$\psi(x,p) = \sum_m^{TSK}\sum_i^{TK} w^{m,i} B(x - \tau^i)\delta(p - \beta^m). \tag{15}$$

Eqs. (14) and (15) show how the kernel ψ is defined as the weighted sum of a group of bell-shaped B-splines of second order. In the case of the Time-Scale group convolution, the functions are also multiplied by Dirac delta functions. The polynomial approximation used for the B-spline is shown in Figure 1. $\tau \in \mathbb{Z}$ represents the centers (in the temporal dimension) of the B-splines. $\beta^m \in \mathbb{Z}$ represents the centers (in the time dilation dimension) of the Dirac functions. $i, m \in \mathbb{Z}$ are the indices used in the sums, with cardinality equal to TK and TSK, respectively. TK and TSK are hyperparameters that define the number of trainable weights w used to obtain a feature map with the Lifting and the Time-Scale group convolution layers. Notice in Figure 1 that for $|x| > 3$, the function returns undesired noise. Therefore, we clamp the function to 0 for every $|x| > 3$.

Given the continuous definitions of Lifting and Time-Scale group convolutions, it is necessary to discretize them. This can be done by using $\int dxf(x) \approx \sum_i \Delta x^i f^i$, where $\Delta x^i = x^i - x^{n-1} = 1$ and $\Delta y^j = y^j - y^{j-1} = 1$. The time dilation factor s and dimension p are discretized in powers of a common base S, resulting in $s = S^n$, $p = S^m$ and $dp \approx \Delta p^m = S^m - S^{m-1} = (S-1)S^{m-1}$. $i, j, n, m \in \mathbb{Z}$ are the indices used, and $S \in \mathbb{R}^+$ is a positive real hyperparameter. These leave the Lifting and Time-Scale group convolution as follows.

$$\Lambda(f)^{j,n} = \sum_i f^i \frac{1}{S^n}\psi^{i-j,n}. \tag{16}$$

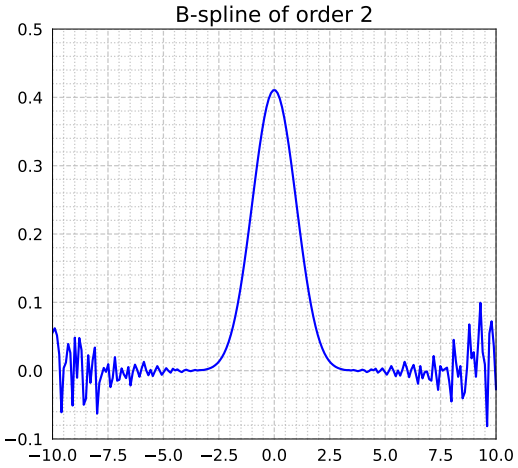


Figure (1) – B-spline function of second order.

$$\Omega(f)^{j,n} = \sum_{i,m} f^{i,m} \frac{(S-1)S^{m-1}}{S^{2n}} \psi^{i-j,m-n}. \tag{17}$$

When dealing with the time dilation elements s, p, it is necessary to define a finite domain for them. As the number of time dilation transformations is infinite, it is compulsory to define a finite number of them to be used in the computations. This, along with $s = S^n$, $p = S^m$, provokes the establishment of maximum and minimum values for n and m. To minimize the effects of aliasing when sampling the dilated kernels, the minimum values for n and m are set to 0. TSK defines the maximum of the time dilation dimension of the kernels, while the maximum value of n is denoted by n_scales.

The Lifting and Time Dilation group convolution use time dilated kernels with different time dilation factors. This use of kernels of different sizes in the same model should provoke benefits similar to the ones obtained in [HZG+24]. Apart from these two advantages, the Lifting and Time-Scale group convolution should have two extra benefits. First, in comparison to the model of He et al. [HZG+24], the combined use of feature maps obtained with kernels of different sizes is executed in each Time Dilation group equivariant convolution, instead of just before the feed-forward network. This should allow a richer analysis by the model. Second, Young's convolution inequality [Bog07], shown in Eq. (18), tells us that the unit norm of the convolution needs to be less than or equal to the product of the unit norms of the functions involved. Because of this, changing the unit norm of the kernel changes the maximum limit of the unit norm of the convolution. As shown in Eq. (19), modifying the kernel size results in changes over its unit norm. Consequently, using larger kernels can lead to larger values in their feature maps. These values could receive more importance from the model because of the size of the kernels used.

$$\int dy |(f \tilde{*} \psi)(y)| \leq \left(\int dx |f(x)| \right) \left(\int dx |\psi(x-y)| \right). \tag{18}$$

$$\int dx \, |\psi\left(\frac{x-y}{s}\right)| \xrightarrow{sx'=x-y} s \int dx' |\psi(x')|. \tag{19}$$

Contrarily, the unit norm of the kernels used in the Lifting and Time Dilation group convolutions is invariant to the change in their size. This is shown in Eqs. (20) and (21).

$$\int dx \, |\frac{1}{s}\psi(\frac{x-y}{s})| \xrightarrow{\alpha x'=x-y} \int dx' |\psi(x')|. \tag{20}$$

$$\int dxdp \; |\frac{1}{s^2}\psi(\frac{x-y}{s}, \frac{p}{s})| \xrightarrow[p=sp']{sx'=x-y} \int dx'dp'|\psi(x', p')|. \tag{21}$$

3 Related Work

Since the introduction of deep learning to SCA in 2016 [MPP16], a large number of works have investigated improving profiling methods [PPM+23]. One of the main advantages of using DL for SCA is that networks can be resilient to the effects of transformations in the trace triggered by hiding countermeasures. Such transformations can be in the amplitude domain where Gaussian noise is added to the traces [KPH+19], and in the temporal domain where traces are misaligned due to, e.g., clock-jitter or random delays [CDP17].

In DLSCA, dealing with desynchronized traces is a common scenario that motivates numerous investigations, see, e.g., [KWPP24, HSAM22]. This is related to the desynchronization of traces being a common consequence of side-channel countermeasures, such as random delay insertion and dummy operation insertion. A common approach to creating desynchronization-robust models involves using data augmentation techniques [LZW+21], which increases the computational cost of training. Cagli et al. [CDP17] first considered using data augmentation to mitigate the effects of clock jitter, indicating that convolutional networks can effectively handle some misalignment in the traces. With the introduction of the ASCAD database in 2018 [BPS+20], which included artificially desynchronized versions, the robustness of networks to these desynchronizations became a standard benchmark for evaluating novel model architectures and methodologies.

More recently, there has been interest from the DLSCA community in the consequences for the leakage when changing the clock frequency of a cryptographic device [LCCR22]. This happens when the same device is used with different clock frequencies, because of the design or the environment in which it operates (thermal throttling). This is a consequence of the direct relation between frequency, power consumption, and heat generated. Concurrently, efforts have been made searching for optimal time-scales of the kernels used in CNN-based SCA [ZBHV20, PA20] and how features learned at different time-scales can be combined to create more efficient attacks [HZG+24].

Zhou and Standaert [ZS20] showed that deep learning does not fully annihilate the need for re-synchronization and introduced a ResNet architecture to address desynchronization. Wu and Picek [WP20] showed that autoencoders can be trained to remove the effects of desynchronization (and other hiding countermeasures) if an attacker can capture a "clean" set of traces without these effects. Finally, Krček et al. [KWPP24] explored the robustness of CNNs to shifts by testing how various levels of data augmentation affect the robustness of convolutional networks to shifts.

The effectiveness of a CNN-based SCA is significantly influenced by the size and number of trainable parameters of the kernels used. As shown by Zaid et al. [ZBHV20], using small filters resulted in models whose intermediate values were less affected by the noise and redundancy of the features in the traces. On the other hand, Paguada and Armendariz [PA20] showed how the use of dilated kernels with large receptive fields allowed the convolutional layers to reduce overfitting. This reduction is achieved by analyzing at the same time more global features, composed of the leakage of the intermediate values of AES and the masking countermeasure. He et al. [HZG+24] demonstrated how using convolutional layers with kernels of different sizes simultaneously resulted in increased model efficiency.

More recently, transformer-based models [VSP+17] have received attention from the SCA community due to their ability to handle long-range dependencies. Jin et al. [JZHY20] introduced attention-based modules inside convolutional blocks. Hajra et al. [HSAM22] introduced transformer-based architectures that are invariant to shifts in traces. In

subsequent work, the same authors introduced the EstraNet model [HCM24], which significantly improves the computational efficiency of attention mechanisms in these networks. This network could be considered as a state-of-the-art in DLSCA for transformer architectures. The level of robustness to shifts for both convolutional and transformer models often depends on the level of shifts occurring in the training dataset. However, Hajra et al. [HSAM22] demonstrated how TransNet was able to overcome this dependency, successfully attacking traces with larger shifts than those used in the training dataset. They linked this behavior with the use of a global average pooling. We will examine this aspect further in the context of group equivariant deep learning. In this area, works like [CW16, RBTH20, WW19] study deep learning models that are naturally robust to a selected group of transformations over the input.

4 The EquivSCA Model

Next, we introduce the EquivSCA Model, a classification model developed based on the principles of group equivariant deep learning. It consists of a sequence of layers, detailed in Table 1. In Table 1, *c_out* denotes the number of output feature maps for convolutional layers or the output dimensionality for dense layers. *ks* specifies the temporal size of the pooling kernel used on the average pooling, applied over the time dimension. *pred* represents the number of classes used to classify. *pred* depends on the leakage model used. If it is the Identity of the byte selected, then $pred = 256$. On the other hand, if the leakage model is the Hamming Weight of the same byte, then $pred = 9$. Finally, TK and TSK represent the sizes of the non-time dilated kernels over the time and time dilation dimensions, respectively. Note that the architecture has only four convolutional blocks (convolution + pooling layer) and three fully connected layers. The Lifting and Time-Scale group convolution Layers are both equivariant to temporal translation $H_\theta \in G$, along with time dilations $T_\alpha \in G$, as shown in Eqs. (24) and (28). H_θ does not change the time-scale of the signal, so p is invariant. This, along with the use of global average pooling as the invariant function, makes the EquivSCA model theoretically robust to both temporal translation and time dilations.

$$\Lambda((T_\alpha \circ H_\theta) \cdot f)(y, s) = \int dx f\left(\frac{x-\theta}{\alpha}\right) \frac{1}{s} \psi\left(\frac{x-y}{s}\right) \Rightarrow \tag{22}$$

$$\text{then, substitution is applied: } \alpha x' = x - \theta. \tag{23}$$

$$\Rightarrow \int dx' f(x') \frac{1}{\alpha^{-1} s} \psi\left(\frac{x' - \alpha^{-1}(y-\theta))}{\alpha^{-1} s}\right) = (T_\alpha \circ H_\theta) \cdot \Lambda(f)(y, s). \tag{24}$$

$$\Omega((T_\alpha \circ H_\theta) \cdot f)(y, s) = \int dx dp f(\alpha^{-1}(x-\theta), \alpha^{-1} p) \frac{1}{s^2} \psi\left(\frac{x-y}{s}, \frac{p}{s}\right) \Rightarrow \tag{25}$$

$$\text{with the following substitutions applied: } \alpha x' = x - \theta, \alpha p' = p. \tag{26}$$

$$\Rightarrow \int dx' dp' f(x', p') \frac{1}{(\alpha^{-1} s)^2} \psi\left(\frac{x' - \alpha^{-1}(y-\theta)}{\alpha^{-1} s}, \frac{p'}{s}\right) = \tag{27}$$

$$= (T_\alpha \circ H_\theta) \cdot \Omega(f)(y, s). \tag{28}$$

As demonstrated by He et al. [HZG+24], using preprocessing techniques over the input can help the model reach better performance. In our case, we require the preprocessing technique to be equivariant to both time dilations and time translations, similar to Lifting and Time Dilation group convolutions. The preprocessing technique, if not equivariant,

Table (1) – EquivSCA model architecture.

Position	Layer	Hyperparameters
0º	BatchNorm1D	
1º	Lifting Convolution	c_out= 32, $TK = 13$
2º	BatchNorm2D	
3º	Activation Function	GELU
4º	Average Pooling	$ks = 2$
5º	Time-Scale Group Convolution	$c_out = 64$, $TK = 15$, $TSK = 3$
6º	BatchNorm2D	
7º	Activation Function	GELU
8º	Average Pooling	$ks = 25$
9º	Time-Scale Group Convolution	$c_out = 128$, $TK = 13$, $TSK = 3$
10º	BatchNorm2D	
11º	Activation Function	GELU
12º	Average Pooling	$ks = 2$
13º	Time-Scale Group Convolution	$c_out = 128$, $TK = 9$, $TSK = 3$
14º	BatchNorm2D	
15º	Activation Function	GELU
16º	Global Average Pooling	
17º	Dense Layer	$c_out = 20$, SELU
18º	Dense Layer	$c_out = 20$, SELU
19º	Dense Layer	$c_out = 20$, SELU
20º	Prediction Layer	$c_out = pred$

would break the equivariance of the following convolution layers, thereby blocking the built-in robustness of the entire model. Motivated by this fact, we used BatchNorm1D as the first layer. This is because it normalizes the input using a mean $\mu \in \mathbb{R}$ and deviation $\sigma \in \mathbb{R}$. As the mean and the deviation used by the BatchNorm1D are calculated with the entire input, they are robust against time dilations ($T_\alpha \in G$) and time translations ($H_\theta \in G$). This leads to the operation being equivariant. Techniques such as MinMaxScaler and StandardScaler from the scikit-learn library use vectors of means $\vec{\mu} \in \mathbb{R}^n$ and deviations $\vec{\sigma} \in \mathbb{R}^n$. Each value of the vectors is calculated for a specific position in the input signal, making it not equivariant to time dilation and time translations.

Another difference EquivSCA has from other CNNs used in DLSCA is the use of GELU activation function [HG23] in the convolutional blocks. GELU is a nonlinear, infinitely differentiable function whose formula can be found in Eq. (29).

$$GELU(x) = \frac{x}{2}\left(1 + tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right). \tag{29}$$

Regarding the hyperparameters of Lifting and Time-Scale group convolution, we set the default base of the time dilation factor S to 1.2. n_scales is set to 7 for every experiment done in Section 5. Lastly, TSK equals 3. The primary reasons for selecting these values are related to a trade-off between the computational cost of the model and its robustness against time dilation. Increasing the S, TSK, TK, and n_scales ends up with convolutions performed with larger (and therefore, more costly) kernels. On the other hand, S is the base of the time dilation factor used in the equivariant convolutions,

and n_scales defines the size of the time dilation dimension. These define the space in which our model is equivariant to transformations. This is a result of the equivariance being obtained through having kernels as time dilated as the signal analyzed. If the model receives as input a trace more time dilated than the maximum dilated kernel, then the equivariant convolutional layers will not be able to give an equivariant output, making the model not robust to that transformation. As the maximum time dilation applied to the kernels is S^{n_scales}, the model will be able to recover learned features from traces time dilated with $S^{-n_scales} \leq \alpha < S^{n_scales}$. The code for the EquivSCA architecture can be found at https://shorturl.at/lxW7K.

5 Experimental Results

5.1 Datasets

ASCADV1. ASCAD datasets represent the most common target for evaluating the performance of DLSCA. The first edition of the dataset (ASCADv1) contains two versions: one with a fixed key for the profiling set and the other with random keys for the profiling set. We denote the former as ASCADf and the latter as ASCADr. The ASCADf dataset[6] contains measurements from an 8-bit AVR microcontroller. The AES-128 implementation is protected with first-order Boolean masking, except for the first two bytes, which are left unprotected for testing purposes. The acquisition window covers only the first round of AES, and each trace consists of 100 000 sample points. The dataset contains 60 000 traces overall, which are split into 50 000 for the profiling phase and 10 000 for the attack. All encryption operations in both the profiling and attack groups are performed using the same fixed key. Following the recommendations from the datasets' authors, we consider the preselected interval of 700 features for our experiments. Note that this is the OPOI cases as discussed in [PWP22].

The ASCADr dataset[7] has a similar setup to ASCADf concerning the measurements source. The main difference is that the traces in the ASCADr profiling group use random keys to perform the encryption instead of a fixed key. The traces are also longer as they consist of 250 000 samples. In total, there are 200 000 traces for the profiling phase and 100 000 for the attack phase. The encryption for the attack group is still performed using a fixed key. We again work with a preselected window of features, which for this dataset contains 1 400 features. For both ASCAD datasets, we consider the Identity leakage model, and we attack the third key byte, as this is the first masked key byte.

ESHARD. The ESHARD dataset[8] contains EM emanations of a first-order boolean masked AES-128 implementation implemented on an STM32F4 microcontroller (Cortex-M4) and was introduced by Vasselle et al. [VTM23]. The datasets includes 100 000 traces with a fixed key, where we take 90 000 traces as the profiling set and 10 000 traces as the attack set. Each trace contains a window of 1 400 samples per trace, consisting of the loading of the mask and the 16 masked S-box operations. Note that we consider the non-shuffled version of the traces. In the case of attacking the ESHARD dataset, the leakage model selected is the Hamming weight of the first byte of the intermediate value, similarly to [WRAp+24].

AES_RD. The AES_RD dataset[9] is a protected software implementation of AES-128 on an 8-bit Atmel AVR microcontroller. The protection is a random delay countermeasure as described by Coron and Kizhvatov [CK09]. This countermeasure causes the misalignment

[6] https://github.com/ANSSI-FR/ASCAD/tree/master/ATMEGA_AES_v1/ATM_AES_v1_fixed_key

[7] https://github.com/ANSSI-FR/ASCAD/tree/master/ATMEGA_AES_v1/ATM_AES_v1_variable_key

[8] https://gitlab.com/eshard/nucleo_sw_aes_masked_shuffled/-/blob/main/Nucleo_AES_masked_non_shuffled.ets

[9] https://github.com/ikizhvatov/randomdelays-traces

of important features, making the attack more difficult to conduct. We attack the first AES key byte, targeting the first S-box operation. The dataset consists of 50 000 traces of 3 500 features each. The leakage model selected is the value of the first byte of the intermediate value, as in other works, see, e.g., [ZBHV20].

Nucleo. To investigate the knowledge transferability of a model against changes in the clock frequency of the SCA target device, we acquired Nucleo datasets. These datasets include Nucleo_96MHz.h5 and Nucleo_168MHz. The datasets were obtained from a Nucleo-F429RZI board with a nucleo Cortex-M4, 256 KB of RAM, and 2 MB of flash memory. The device executes AES-128 without any countermeasure against side-channel attacks. We use a probe to measure the electromagnetic emissions emitted by the device, which were then sampled and saved, forming the traces with an Lecroy WaveRunner9404M oscilloscope with 1 GS/s of sample frequency. The equipment allowed us to obtain the traces for the clock's frequencies of 96 MHz and 168 MHz, resulting in Nucleo_96MHz.h5 and Nucleo_168MHz.h5, respectively. As discussed in Section 2.1, the operations that need to be done to execute AES are still the same. Consequently, the biggest effect of changing the clock's frequency is that the time needed to perform the algorithm varies. It is also important to note that even though, ideally, changing the frequency should provoke the time dilation of the reference traces, most electronic components used on the devices exhibit behavior changes with the frequency of the electric signal. This is the case of capacitors and inductors whose impedance depends on the frequency. This dependence causes that in real-world scenarios, the relation between traces obtained for different clock frequency values differs in more than one time dilation transformation. The datasets include profiling traces obtained for different plaintexts and keys. The attack traces were obtained for a fixed key and different plaintexts. Nucleo_96MHz and Nucleo_168MHz include the raw traces measured, with 90 000 and 50 000 samples, respectively. Both datasets include 100 000 profiling traces and 50 000 attack traces. The Nucleo datasets are publicly available at Zenodo.[10]

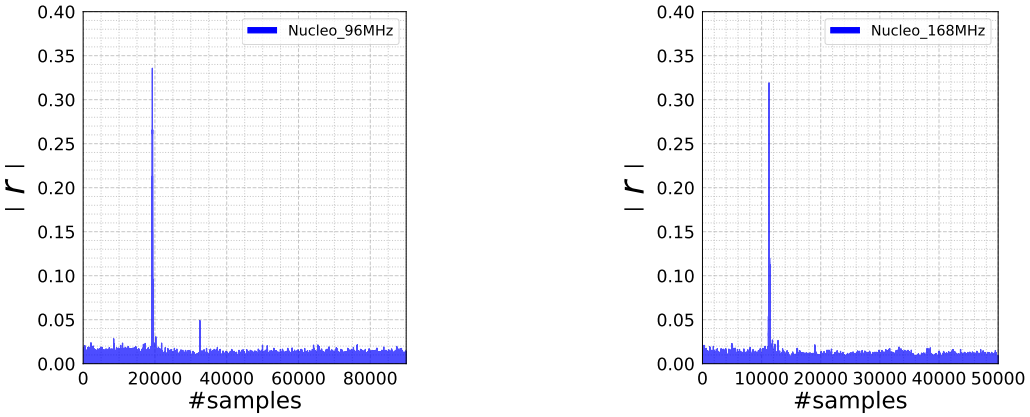
We calculated the absolute value of the Pearson correlation coefficient $|r|$ of the samples in the profiling traces and the first byte in the intermediate value. The result is displayed in Figure 2. In each graph of Figure 2, a main peak with a correlation magnitude exceeding 0.3 can be seen. The samples of the peaks leak more information about the target byte than the rest of the trace. Notice also how the positions of the samples that leak the most information for each dataset are related. As we mentioned, the number of operations required to execute AES (N_{op}) remains the same across the datasets. Thus, knowing that the clock's frequencies are used to define the speed at which operations are performed, the positions with maximum leakage, 11 275 for Nucleo_168MHz and 19 295 for Nucleo_96MHz, are related by $N_{op} = 11275 \cdot 168 \approx 19295 \cdot 96$.

5.2 Experimental Settings and Methodology

The training setup consists of 100 epochs of training, using a batch size of 256 traces. The training dataset contains the entire profiling dataset. We employed the ADAM optimizer and the CrossEntropy loss function. Finally, the learning rate was managed by the OneCyleLR scheduler. It is configured with a maximum learning rate equal to 5E-3, and the warm-up process occupies 40% of the training process.

The guessing entropy (GE) was used to evaluate the effectiveness achieved by the model trained for every epoch. This way, after training for 100 epochs, we can observe the evolution of GE with the training. We select as the best version of each model the one that obtained the lowest number of traces needed to reach GE equal to 0 ($N_{GE=0}$). If GE did not reach 0 for any epoch, then we select as the best version of the model the one that reaches the lowest value of GE for the maximum number of attacking traces used. We

[10] https://shorturl.at/iV1zH



(a) Nucleo_96MHz. (b) Nucleo_168MHz.

Figure (2) – Absolute value of the Pearson correlation coefficient between the traces and the first byte of the intermediate value, for datasets Nucleo_168MHz and Nucleo_96MHz.

repeat the training runs 5 times, obtaining from each run the best versions of the models. Using these models, we calculate the average guessing entropy of the model. This serves to reduce the dependence of the results on the initialization of the model. We compare our results with EffCNN from Zaid et al. [ZBHV20] and EstraNet from Hajra et al. [HCM24]. If possible, taking into account the dataset used, we employed the configuration given by the authors of the models. If not, we used a configuration from another compatible dataset. For EstraNet, we selected the configuration given for ASCADr. In the case of EffCNN, we used the configuration given in [PA20] for ASCADr for the ESHARD dataset. For EstraNet, we utilized the training configuration specified by the authors, as they highlighted its criticality for proper model operation.

5.3 Study of the Dilation Factor S

The EquivSCA model is defined by a time dilation factor, S, which is used to generate dilated versions of the model's kernels. This parameter is expected to influence the model's effectiveness against time dilated side-channel traces. To investigate this influence and justify our choice of $S = 1.2$ for all subsequent experiments, we trained a set of EquivSCA models with varying S values on the ASCADr dataset. During training, we saved a snapshot of the model after each epoch. To identify the best performance for each value of S, we then evaluated every saved snapshot at performing an attack with 1000 traces, which were artificially time dilated by a factor α ranging from 0.5 to 1.5. To resample the traces, we have used the function resample[11] from scipy. The metric reported is the lowest Guessing Entropy (GE (1000)) value achieved across all epochs. The results of this analysis are presented in Figure 3.

Figure 3 provides two main findings. First, a model with $S = 1.0$, which implies no kernel dilation, performs poorly, yielding a guessing entropy of approximately 40 for all tested values of α. Second, models configured with $S = 1.2$ and $S = 1.3$ achieve the lowest GE values across the entire range of α. Thus, we selected $S = 1.2$ for our experiments, as it provides a similar level of robustness to time dilation as $S = 1.3$ but at a lower computational cost. This cost-saving is because a larger S value results in larger, more computationally expensive kernels.

[11] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.resample.html

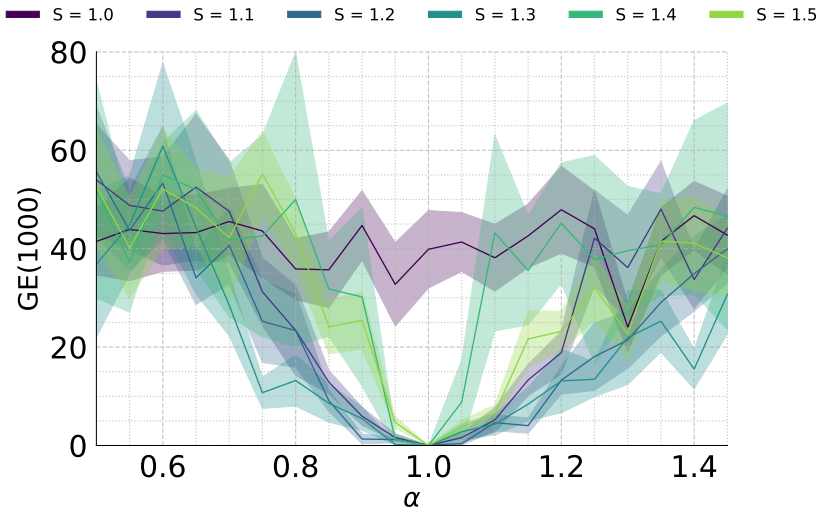


Figure (3) – Evaluation of the minimum guessing entropy reached by EquivSCA during the training in relation to the value S used in the architecture and the α factor of the traces.

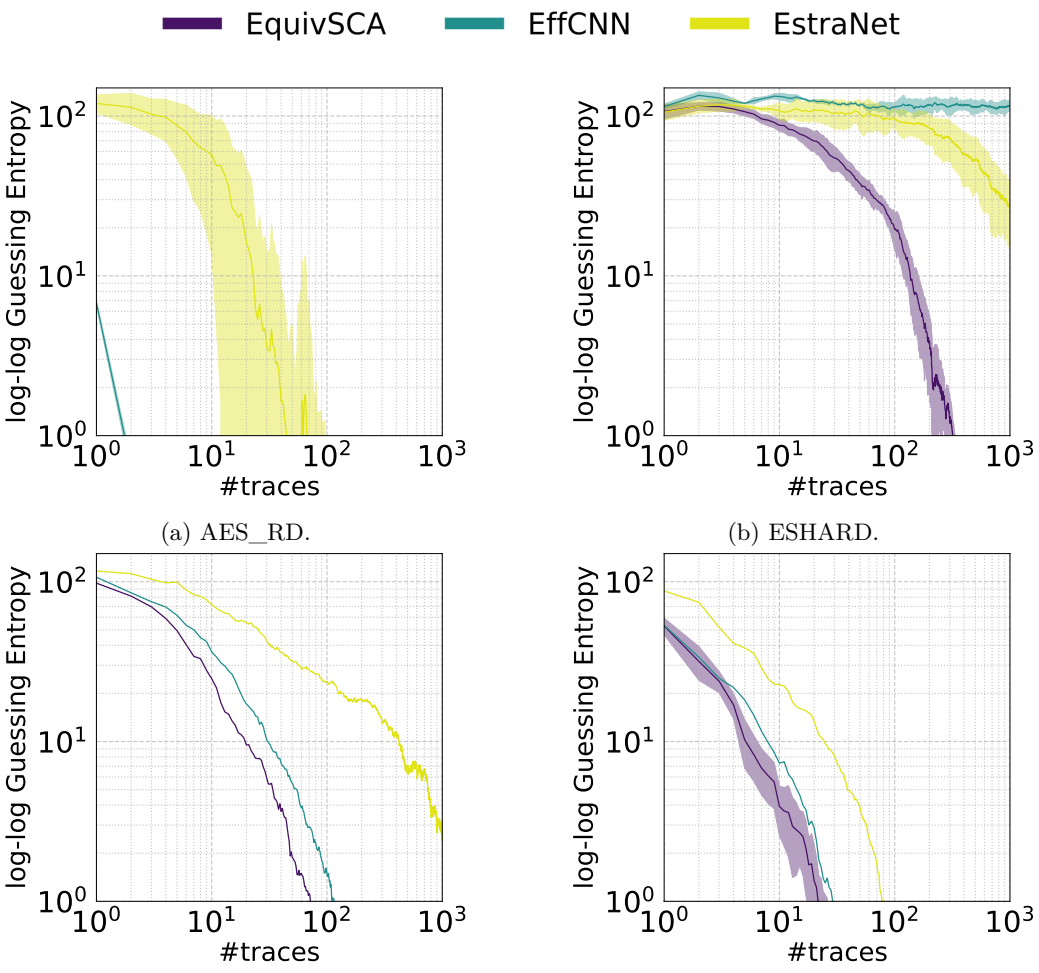
5.4 Performance Evaluation

We train the models to attack the datasets AES_RD, ESHARD, ASCADf, and ASCADr. We provide the results in Figure 4 and Table 2.

Table (2) – Results of the guessing entropy value obtained with 1000 traces and the average value of traces needed to reach GE=0 for the different datasets.

Case	Model	GE (1000)	avg $N_{GE=0}$
AES_RD	*EquivSCA*	0	1
	EffCNN	0	2
	EstraNet	0	45
ESHARD	*EquivSCA*	0	319
	EffCNN	114	None
	EstraNet	26.8	None
ASCADf	*EquivSCA*	0	72
	EffCNN	0	116
	EstraNet	2.51	None
ASCADr	*EquivSCA*	0	22
	EffCNN	0	29
	EstraNet	0	78

The results show that EquivSCA achieved GE of 0 using only a single trace on the AES_RD dataset. As such, this results in EquivSCA not appearing in the log-log plot of Figure 4. For the ESHARD dataset, EquivSCA was the only model among those evaluated to perform a successful attack. Furthermore, for the ASCADf and ASCADr datasets, EquivSCA demonstrated the most rapidly decreasing GE curves when compared to EstraNet and EffCNN. In terms of computational cost, EquivSCA is more demanding



(a) AES_RD.

(b) ESHARD.

(c) ASCADf.

(d) ASCADr.

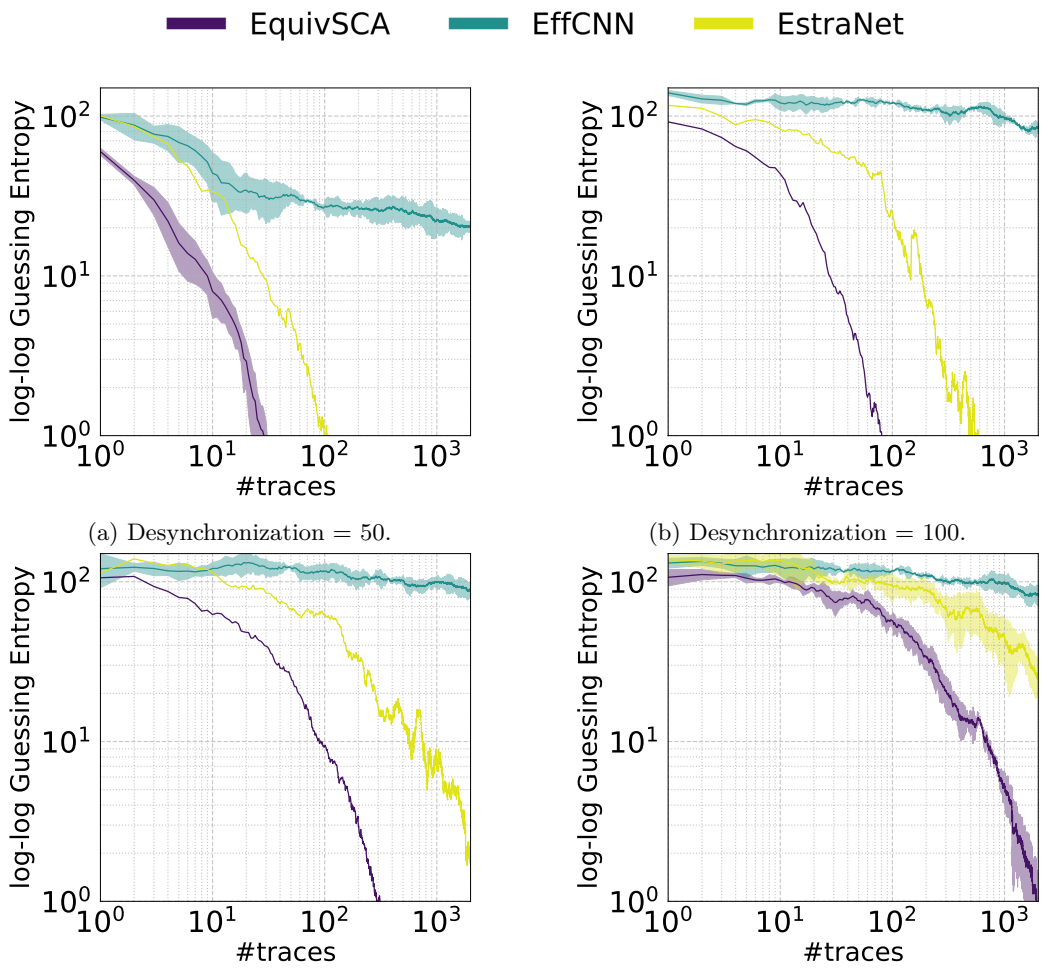
Figure (4) – Log-log representations of the average guessing entropy function obtained by each model attacking different datasets.

to train than EstraNet or EffCNN. Training on the ASCADv1r dataset with one NVIDIA RTX A5000 required 198 minutes for EquivSCA, while EstraNet and EffCNN took 138 and 42 minutes, respectively.

5.5 Temporal Shift Robustness

EquivSCA's architecture is defined using an equivariant-based methodology that should help create a built-in robustness of the model against desynchronization. To test and compare this natural robustness, we trained the models to attack ASCADr with desynchronization 50. Then, the trained models are evaluated with the same dataset but with different values of desynchronization. We selected this dataset because of the availability of its raw traces, which permitted us to define the desynchronization of the traces. For ASCADr, the results are in Figure 5 and Table 3.

We can see how EquivSCA presents itself as the most robust model against desynchronization. Notice how EquivSCA is considerably less affected by larger offsets than those used in its training, in comparison with EffCNN. This is a consequence of using BatchNorm1D as the preprocessing technique, rather than the MinMaxScaler used in



(a) Desynchronization = 50.

(b) Desynchronization = 100.

(c) Desynchronization = 200.

(d) Desynchronization = 400.

Figure (5) – Log-log representation of the average guessing entropy function obtained by each model at an attack over the desynchronized versions of ASCADr.

EftfCNN. The use of global average pooling instead of the flatten layer also yields an edge in terms of creating a natural robustness to time translations of the input traces.

Compared to EstraNet, for the case of ASCADr, both architectures present a natural robustness to desynchronization. If we compare the results of EquivSCA and EstraNet from Table 3, EquivSCA requires for every value of desynchronization a lower number of traces to attack. Moreover, if we calculate the increment in average $N_{GE=0}$ with each value of desynchronization, EquivSCA presents smaller increments with the increase of the offset. The robustness of EstraNet against desynchronization could be the consequence of the attention mechanism and the Softmax-attention layer presented in the model. Attention mechanism is known for being permutation equivariant [HSAM22]. As the time translations can be seen as a composition of permutations, the attention mechanism is then equivariant to time translations. Softmax-attention calculates the attention of the input with a set of trainable queries, acting as the shift-invariant operation of EstraNet.

Table (3) – Results of the guessing entropy value obtained with 2000 traces and the average value of traces needed to reach GE=0 for the attacks over the desynchronized versions of ASCADr.

Case	Model	GE (2000)	avg $N_{GE=0}$
Desynchronization = 50	*EquivSCA*	0	29
	EffCNN	20.3	None
	EstraNet	0	99
Desynchronization = 100	*EquivSCA*	0	81
	EffCNN	84.9	None
	EstraNet	0	497
Desynchronization = 200	*EquivSCA*	0	291
	EffCNN	89.3	None
	EstraNet	1.81	None
Desynchronization = 400	*EquivSCA*	0.8	1929
	EffCNN	82.2	None
	EstraNet	26.2	None

5.6 Time Dilation Robustness

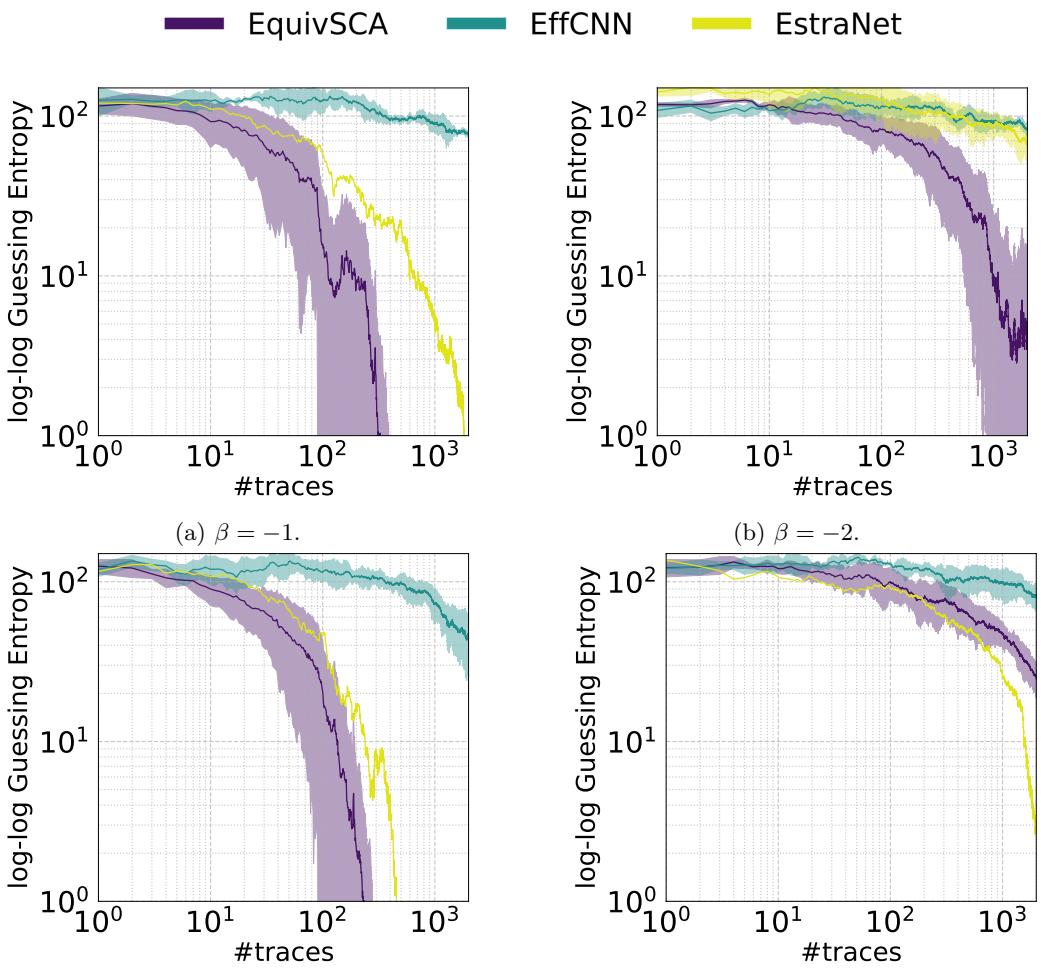
Following the analysis of time translation robustness, we now evaluate the models' robustness against time dilations. This property is a primary motivation for our proposed EquivSCA model, which is based on time dilation-equivariant convolutions. To our knowledge, this is the first work in DLSCA to assess the performance of deep learning models on traces with varying time scales. Therefore, we compare EquivSCA (with and without data augmentation) against EstraNet and EffCNN trained with data augmentation. This comparison benchmarks the performance of a model with built-in invariance against a conventional approach with just data augmentation.

We begin by studying the models' built-in robustness. For this purpose, we reuse the models previously trained in Section 5.5 to attack the ASCADr dataset with desynchronization 50. However, the original EffCNN model cannot be used directly. Its architecture lacks a global average pooling (GAP) layer before the flatten layer, which means the input size of its first dense layer depends on the trace length. This dependency makes the model incompatible with variable-length inputs, a requirement for our time dilation experiment. To resolve this, we modified the EffCNN architecture by inserting a GAP layer immediately before the flatten layer.

The evaluation traces were transformed using dilation factors α defined as powers of 1.2 (i.e., $\alpha = 1.2^{\beta}$). This choice is motivated by the internal dilation factor $S = 1.2$ used in EquivSCA. As EquivSCA uses kernels dilated by factors of S^{β}, evaluating with corresponding α values allows assessing the upper limits of the model's effectiveness. The results for models trained without data augmentation are given in Figure 6 and Table 4.

Next, we retrained the models but we also added data augmentation. Specifically, the original ASCADr training set was augmented with two additional copies, which were resampled with dilation factors of $\alpha = S$ and $\alpha = S^{-1}$ (where $S = 1.2$). The results for the models trained on this augmented dataset are presented in Figure 7 and Table 4.

The results in Figures 6 and 7 show that EquivSCA outperforms both EstraNet and, most notably, EffCNN, which fails to achieve a successful attack in any tested scenario. However, when trained with data augmentation, EstraNet's performance becomes comparable to that of EquivSCA. This suggests that for EstraNet, the combination of attention mechanisms with small (size 3) convolutional kernels can yield a model robust to time dilations, provided it is trained with an augmented dataset. In contrast, the EquivSCA model is inherently robust to this type of input transformation due to its



(a) $\beta = -1$.

(b) $\beta = -2$.

(c) $\beta = 1$.

(d) $\beta = 2$.

Figure (6) – Log-log representation of the average guessing entropy function obtained by each model at an attack over the time dilated versions of ASCADr with desynchronization 50.

equivariant design, requiring no attention layers. While this robustness is a built-in feature, it is further enhanced by data augmentation.

5.7 Transfer Learning and Different Clock Frequencies

The clock frequency of cryptographic devices is commonly manipulated for reasons such as reducing power consumption, mitigating side-channel attacks via clock jitter, or managing heat through thermal throttling. As previously discussed, altering the clock frequency theoretically results in time dilated power traces, since the sequence of cryptographic operations remains identical while the time to execute them changes.

Given that EquivSCA has demonstrated robustness to time dilations, we now investigate its transfer learning capabilities. Specifically, we study the ability of a model trained on traces from a device operating at one clock frequency to successfully attack traces from the same device operating at another frequency. To this end, we use two datasets, Nucleo_96MHz and Nucleo_168MHz. The theoretical time dilation factor, α, relating these two datasets can be calculated from their respective clock frequencies: $\alpha = 168/96 = 1.75$. Therefore, to attack the Nucleo_168MHz dataset using a model trained only

Table (4) – Results of the guessing entropy value obtained with 2000 traces and the average value of traces needed to reach GE=0 for the attacks over the time dilated versions of ASCADr with desynchronization 50. The results are formatted as without data augmentation/with data augmentation.

Case	Model	GE (2000)	avg $\mathbf{N_{GE=0}}$
$\beta = -2$	*EquivSCA*	3.52/1.34	None/None
	EffCNN	83.8/80	None/None
	EstraNet	65.4/17.7	None/None
$\beta = -1$	*EquivSCA*	0.13/0	312/99
	EffCNN	76.9/37.1	None/None
	EstraNet	0/0	1810/99
$\beta = 1$	*EquivSCA*	0/0	230/71
	EffCNN	44.2/8.16	None/None
	EstraNet	0/0	455/117
$\beta = 2$	*EquivSCA*	24.7/0.31	None/1015
	EffCNN	80.6/86.6	None/None
	EstraNet	3.03/0	None/362

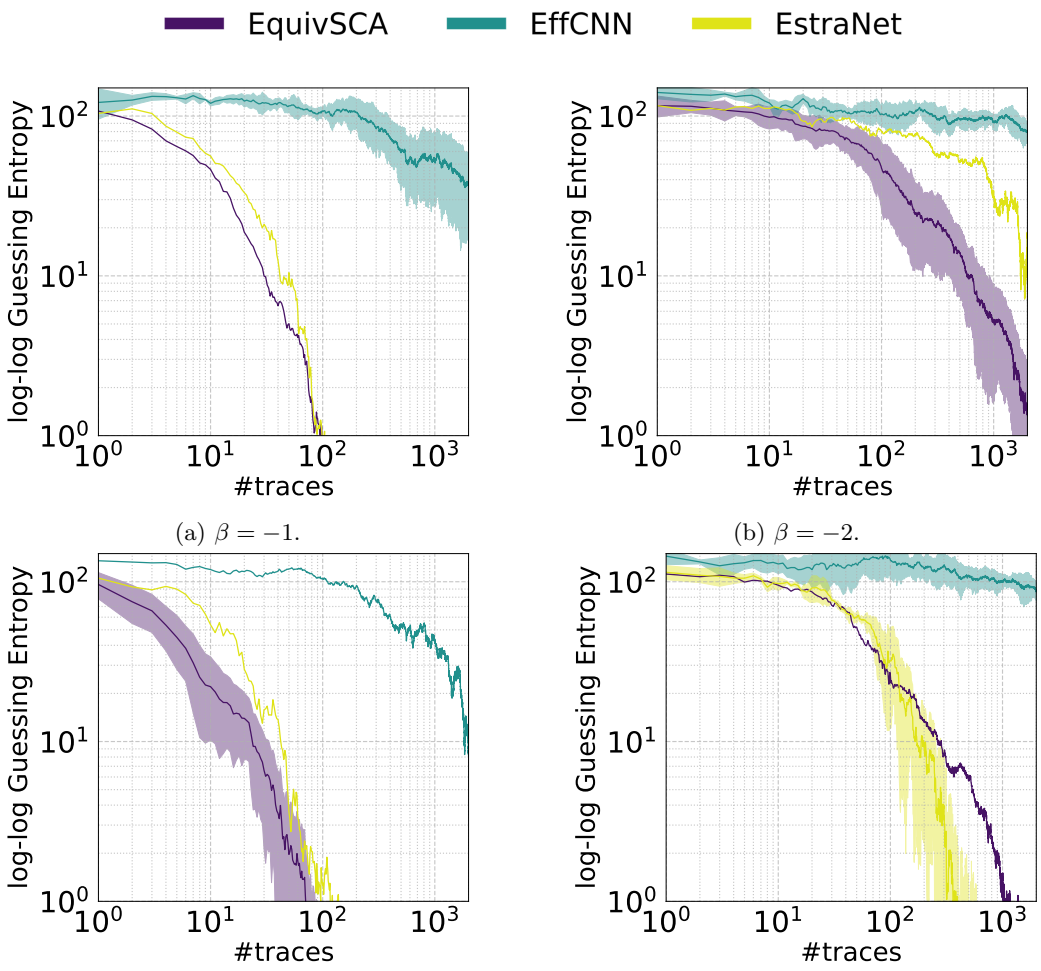
on Nucleo_96MHz traces, a good dilation factor for the EquivSCA model would be $S = 1.75$. However, the large computational cost associated with this value makes us select $S = \sqrt{1.75} \approx 1.3$. This choice is motivated by the fact that the second-level dilated kernels in EquivSCA (i.e., those dilated by S^2) will then match the theoretical dilation factor between the datasets. Moreover, we aim to showcase whether our neural network can still perform well even if the dilation factor used is not exactly the same as the one obtained through the theoretical calculations.

We adopt the OPOI scenario [PWP22] for this experiment. The models are trained on the maximum-correlation window $[18000, 20625)$ from the Nucleo_96MHz dataset, targeting the first byte of an intermediate value as the leakage source. Following the same protocol as in Section 5.6, we train EquivSCA (with $S = 1.3$), EstraNet, and EffCNN, both with and without data augmentation (which also uses $S = 1.3$). After training, we use the model snapshot from each epoch to attack the $[11000, 12500)$ window of the Nucleo_168MHz dataset. The results are presented in Figure 8 and Table 5.

The results show that only the EquivSCA model, when trained with data augmentation, successfully attacked the Nucleo_168MHz dataset; all other configurations failed. Clearly, having different clock frequencies represent a challenging scenario with difficulty above common desynchronization countermeasures or Boolean masking. This also demonstrates that neural network manage to learn well what is given in training set, but may struggle to adapt to changes given in the test set.

5.8 Limitations

While our experiments clearly show the advantages of equivariant CNNs, there are also some limitations. One consideration is that, in our experiments, we use a single hyperparameter and training setup derived from [ZBHV20]. In the original works, both EstraNet and the EffCNN are adapted to specific targets, allowing for performance improvements. As



(a) $\beta = -1$.

(b) $\beta = -2$.

(c) $\beta = 1$.

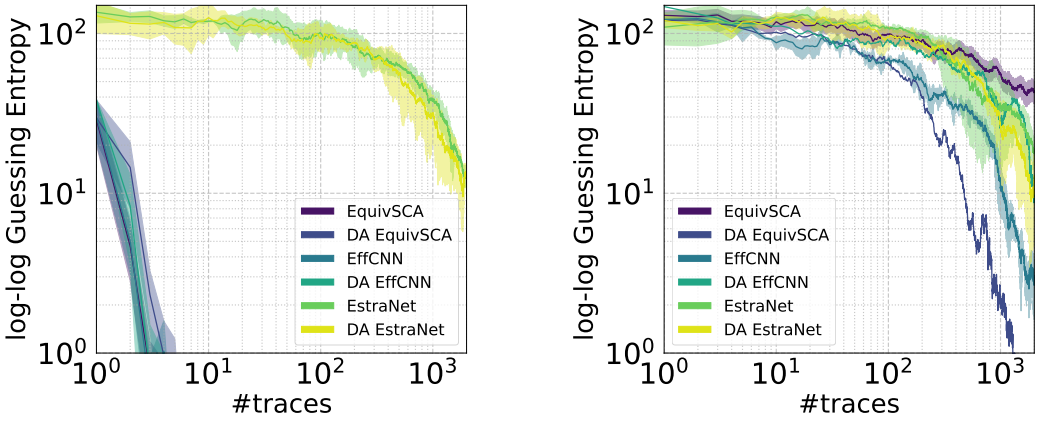
(d) $\beta = 2$.

Figure (7) – Log-log representation of the average guessing entropy function obtained by each model at an attack over the time dilated versions of ASCADr with desynchronization 50, trained with data augmentation.

these do not provide a specifically adapted model for some (EffCNN) or all (EstraNet) scenarios, the obtained performance is not representative of what could be possible for these architectures. Tuning the models to each target should result in further performance improvements. However, we aimed for consistent training setups across experiments, rather than specific improvements, to showcase 'average' performance when applying the method to a new target.

6 Conclusions and Future Work

In this work, we propose EquivSCA, a model based on group equivariant neural networks. More precisely, we propose a model with built-in robustness to transformations such as time translations and time dilations, As such, the EquivSCA model can identify patterns in feature maps generated using kernels of varying sizes. EquivSCA has been demonstrated to be a state-of-the-art model when it comes to executing attacks over the datasets AES_RD, ASCADv1, and ESHARD. The model is capable of handling desynchronized traces with



(a) Nucleo 96MHz.

(b) Nucleo 168MHz.

Figure (8) – Log-log representation of the guessing entropy function obtained by each model at an attack over the datasets Nucleo_96MHz and Nucleo_168MHz, having been trained with just Nucleo_96MHz.

Table (5) – Results of the guessing entropy value obtained with 2000 traces and the average number of traces needed to reach GE=0 for the attacks over Nucleo_96MHz and Nucleo_168MHz.

Case	Model	GE (2000)	avg $N_{GE=0}$
Nucleo_96MHz	EquivSCA	0	3
	EffCNN	0	3
	EstraNet	12.4	None
	DA EquivSCA	0	4
	DA EffCNN	0	3
	DA EstraNet	13.7	None
Nucleo_168MHz	EquivSCA	42.7	None
	EffCNN	2.69	None
	EstraNet	17.7	None
	DA EquivSCA	0	1288
	DA EffCNN	17.7	None
	DA EstraNet	12.9	None

offsets larger than those found in the training traces. EquivSCA also showed to be naturally robust against time dilation transformations. This robustness could be increased by using a data augmentation technique that involves training the model with resampled copies of the original traces. Moreover, EquivSCA showed the capability of being trained with traces obtained from a device with one clock frequency and performing a successful attack over traces obtained from the same device but with a different clock frequency.

For future work, one direction could be related to mixing attention layers with the time dilation equivariant convolutional layers. This is based on the attention layer being permutation equivariant, which makes it compatible with the temporal translation equivariant layers. Moreover, as shown in Section 5.6, the EstraNet model trained with data augmentation achieved results similar to those obtained by EquivSCA. Since EquivSCA does not have an attention layer, adding it could lead to better results.

Acknowledgment

This work was supported by the Provincial Council of Bizkaia through the CyberAI2Grid Project (2019/0097).

References

[ANS+23] Thomas Altstidl, An Nguyen, Leo Schwinn, Franz Köferl, Christopher Mutschler, Björn M. Eskofier, and Dario Zanca. Just a matter of scale? reevaluating scale equivariance in convolutional neural networks. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*, pages 1–8. IEEE, 2023. doi:10.1109/IJCNN54540.2023.10191724.

[AW19] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *J. Mach. Learn. Res.*, 20:184:1–184:25, 2019. URL: https://jmlr.org/papers/v20/19-519.html.

[BB21] Valerio Biscione and Jeffrey S. Bowers. Convolutional neural networks are not invariant to translation, but they can learn to be. *J. Mach. Learn. Res.*, 22:229:1–229:28, 2021. URL: https://jmlr.org/papers/v22/21-0019.html.

[BBCV21] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021. URL: https://arxiv.org/abs/2104.13478, arXiv:2104.13478.

[Bog07] Vladimir I. Bogachev. *Measure Theory I*. Springer, Berlin, 2007. doi:10.1007/978-3-540-34514-5.

[BPS+20] Ryad Benadjila, Emmanuel Prouff, Rémi Strullu, Eleonora Cagli, and Cécile Dumas. Deep learning for side-channel analysis and introduction to ASCAD database. *J. Cryptogr. Eng.*, 10(2):163–188, 2020. URL: https://doi.org/10.1007/s13389-019-00220-8, doi:10.1007/S13389-019-00220-8.

[CDP17] Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 45–68. Springer, 2017. doi:10.1007/978-3-319-66787-4_3.

[CK09] Jean-Sébastien Coron and Ilya Kizhvatov. An efficient method for random delay generation in embedded software. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, volume 5747 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 2009. doi:10.1007/978-3-642-04138-9_12.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*,

volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002. doi:10.1007/3-540-36400-5_3.

[CW16] Taco Cohen and Max Welling. Group equivariant convolutional networks. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2990–2999. JMLR.org, 2016. URL: http://proceedings.mlr.press/v48/cohenc16.html.

[GBC16] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016. URL: http://www.deeplearningbook.org/.

[HCM24] Suvadeep Hajra, Siddhartha Chowdhury, and Debdeep Mukhopadhyay. Estranet: An efficient shift-invariant transformer network for side-channel analysis. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2024(1):336–374, 2024. URL: https://doi.org/10.46586/tches.v2024.i1.336-374, doi:10.46586/TCHES.V2024.I1.336-374.

[HG23] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023. URL: https://arxiv.org/abs/1606.08415, arXiv:1606.08415.

[HSAM22] Suvadeep Hajra, Sayandeep Saha, Manaar Alam, and Debdeep Mukhopadhyay. Transnet: Shift invariant transformer network for side channel analysis. In Lejla Batina and Joan Daemen, editors, *Progress in Cryptology - AFRICACRYPT 2022: 13th International Conference on Cryptology in Africa, AFRICACRYPT 2022, Fes, Morocco, July 18-20, 2022, Proceedings*, volume 13503 of *Lecture Notes in Computer Science*, pages 371–396. Springer Nature Switzerland, 2022. doi:10.1007/978-3-031-17433-9_16.

[HZG+24] Pengfei He, Ying Zhang, Han Gan, Jianfei Ma, and Hongxin Zhang. Side-channel attacks based on attention mechanism and multi-scale convolutional neural network. *Comput. Electr. Eng.*, 119:109515, 2024. URL: https://doi.org/10.1016/j.compeleceng.2024.109515, doi:10.1016/J.COMPELECENG.2024.109515.

[JZHY20] Minhui Jin, Mengce Zheng, Honggang Hu, and Nenghai Yu. An enhanced convolutional neural network in side-channel attacks and its visualization. *CoRR*, abs/2009.08898, 2020. URL: https://arxiv.org/abs/2009.08898, arXiv:2009.08898.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999. doi:10.1007/3-540-48405-1_25.

[Koc96] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996. doi:10.1007/3-540-68697-5_9.

[KPH+19] Jaehun Kim, Stjepan Picek, Annelie Heuser, Shivam Bhasin, and Alan Hanjalic. 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. URL: https://doi.org/10.13154/tches.v2019.i3.148-179, doi:10.13154/TCHES.V2019.I3.148-179.

[KWPP24] Marina Krček, Lichao Wu, Guilherme Perin, and Stjepan Picek. Shift-invariance robustness of convolutional neural networks in side-channel analysis. *Mathematics*, 12(20), 2024. URL: https://www.mdpi.com/2227-7390/12/20/3279, doi:10.3390/math12203279.

[LCCR22] Chen Liu, Abhishek Chakraborty, Nikhil Chawla, and Neer Roggel. Frequency throttling side-channel attack. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 1977–1991. ACM, 2022. doi:10.1145/3548606.3560682.

[LCY14] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL: http://arxiv.org/abs/1312.4400.

[LZW+21] Zhimin Luo, Mengce Zheng, Ping Wang, Minhui Jin, Jiajia Zhang, and Honggang Hu. Towards strengthening deep learning-based side channel attacks with mixup. In *20th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2021, Shenyang, China, October 20-22, 2021*, pages 791–801. IEEE, 2021. URL: https://doi.org/10.1109/TrustCom53373.2021.00114, doi:10.1109/TRUSTCOM53373.2021.00114.

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In Claude Carlet, M. Anwar Hasan, and Vishal Saraswat, editors, *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*, volume 10076 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2016. doi:10.1007/978-3-319-49445-6_1.

[PA20] Servio Paguada and Igor Armendariz. The forgotten hyperparameter: - introducing dilated convolution for boosting cnn-based side-channel attacks. In Jianying Zhou, Mauro Conti, Chuadhry Mujeeb Ahmed, Man Ho Au, Lejla Batina, Zhou Li, Jingqiang Lin, Eleonora Losiouk, Bo Luo, Suryadipta Majumdar, Weizhi Meng, Martín Ochoa, Stjepan Picek, Georgios Portokalidis, Cong Wang, and Kehuan Zhang, editors, *Applied Cryptography and Network Security Workshops - ACNS 2020 Satellite Workshops, AIBlock, AIHWS, AIoTS, Cloud S&P, SCI, SecMT, and SiMLA, Rome, Italy, October 19-22, 2020, Proceedings*, volume 12418 of *Lecture Notes in Computer Science*, pages 217–236. Springer, 2020. doi:10.1007/978-3-030-61638-0_13.

[PPM+23] Stjepan Picek, Guilherme Perin, Luca Mariot, Lichao Wu, and Lejla Batina. Sok: Deep learning-based physical side-channel analysis. *ACM Comput. Surv.*, 55(11):227:1–227:35, 2023. doi:10.1145/3569577.

[PWP22] Guilherme Perin, Lichao Wu, and Stjepan Picek. Exploring feature selection scenarios for deep learning-based side-channel analysis. *IACR Trans. Cryptogr.*

Hardw. Embed. Syst., 2022(4):828–861, 2022. URL: https://doi.org/10.46586/tches.v2022.i4.828-861, doi:10.46586/TCHES.V2022.I4.828-861.

[RBTH20] David W. Romero, Erik J. Bekkers, Jakub M. Tomczak, and Mark Hoogendoorn. Wavelet networks: Scale equivariant learning from raw waveforms. *CoRR*, abs/2006.05259, 2020. URL: https://arxiv.org/abs/2006.05259, arXiv:2006.05259.

[SMS21] Ivan Sosnovik, Artem Moskalev, and Arnold W. M. Smeulders. How to transform kernels for scale-convolutions. In *IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, QC, Canada, October 11-17, 2021*, pages 1092–1097. IEEE, 2021. doi:10.1109/ICCVW54120.2021.00128.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[VTM23] Aurélien Vasselle, Hugues Thiebeauld, and Philippe Maurine. Spatial dependency analysis to extract information from side-channel mixtures: extended version. *J. Cryptogr. Eng.*, 13(4):409–425, 2023. URL: https://doi.org/10.1007/s13389-022-00307-9, doi:10.1007/S13389-022-00307-9.

[WP20] Lichao Wu and Stjepan Picek. Remove some noise: On pre-processing of side-channel measurements with autoencoders. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(4):389–415, 2020. URL: https://doi.org/10.13154/tches.v2020.i4.389-415, doi:10.13154/TCHES.V2020.I4.389-415.

[WRAp+24] Lichao Wu, Azade Rezaeezade, Amir Ali-pour, Guilherme Perin, and Stjepan Picek. Leakage model-flexible deep learning-based side-channel analysis. *IACR Communications in Cryptology*, 1(3), 2024. doi:10.62056/ay4c3txol7.

[WW19] Daniel E. Worrall and Max Welling. Deep scale-spaces: Equivariance over scale. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7364–7376, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/f04cd7399b2b0128970efb6d20b5c551-Abstract.html.

[ZBHV20] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli. Methodology for efficient CNN architectures in profiling attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(1):1–36, 2020. URL: https://doi.org/10.13154/tches.v2020.i1.1-36, doi:10.13154/TCHES.V2020.I1.1-36.

[ZS20] Yuanyuan Zhou and François-Xavier Standaert. 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(1):85–95, 2020. URL: https://doi.org/10.1007/s13389-019-00209-3, doi:10.1007/S13389-019-00209-3.