Conditional Variational AutoEncoder based on Stochastic Attack

Bridging Deep Learning and Profiled Side-Channel Attacks

Gabriel Zaid[1*], Lilian Bossuet[2], Mathieu Carbone[1], Amaury Habrard[2], and Alexandre Venelli[3]

[1] Thales ITSEF, Toulouse, France, firstname.lastname@thalesgroup.com
[2] Univ Lyon, UJM-Saint-Etienne, CNRS Laboratoire Hubert Curien UMR 5516 F-42023, Saint-Etienne, France, firstname.lastname@univ-st-etienne.fr
[3] NXP Semiconductors, France, firstname.lastname@nxp.com

Abstract. Over the recent years, the cryptanalysis community leveraged the potential of research on Deep Learning to enhance attacks. In particular, several studies have recently highlighted the benefits of Deep Learning based Side-Channel Attacks (DLSCA) to target real-world cryptographic implementations. While this new research area on applied cryptography provides impressive result to recover a secret key even when countermeasures are implemented (e.g. desynchronization, masking schemes), the lack of theoretical results make the construction of appropriate models a notoriously hard problem. In this work, we propose the first solution that bridges DL and SCA. Based on theoretical results, we develop the first generative model, called *Conditionnal Variational AutoEncoder based on Stochastic Attacks* (cVAE-SA), designed from the well-known *Stochastic Attacks*, that have been introduced by Schindler *et al.* in 2005. This model reduces the black-box property of DL and eases the architecture design for every real-world crypto-system as we define theoretical complexity bounds which only depend on the dimension of the (reduced) trace and the targeting variable over $\mathbb{F}_2^n$. We validate our theoretical proposition through simulations and public datasets on wide-range of use-cases, including multi-task learning, curse of dimensionality and masking scheme.

Keywords: Side-Channel Attacks · Deep Learning · Discriminative Models · Generative Models · Stochastic Attacks · Variational AutoEncoder

1 Introduction

Context. Side-Channel Analysis (SCA) is a class of cryptographic attack in which an adversary tries to exploit the vulnerabilities of a real-word crypto-system for key recovery by analyzing its physical characteristics via side-channel

* Work done when the author was a PhD candidate at CNRS Laboratoire Hubert Curien.

traces like power consumption or electromagnetic emissions. During the execution of an algorithm into a crypto-system, side-channel traces record the intermediate variable being processed. A *sensitive value* denotes an intermediate variable that depends on small pieces of the secret key, namely *subkeys*. Side-channel observables related to sensitive variables are referred as *traces* and can be exploited in order to recover secret subkeys. One of the most powerful types of SCA, referred to as *profiled* SCA, is defined as a two-stage process. The underlying problem solved by this type of SCA relies on the classification task based on estimation of conditional (class-oriented) *probability distributions* related to each secret subkey. The first profiled SCA so-called *template attacks* was introduced by [4] considering that a real leakage model coincides exactly with the deterministic part of the leakage and gaussian noise assumption. Then, Schindler *et al.* proposed the so-called *stochastic attacks* that refine the approximation of real leakage model [18] characterizing the leakage as a pseudo-boolean function over a monomial basis.Both approaches constitute the classical profiled SCA in the current state-of-the-art and are based on the construction of a *generative model*.

Two separated worlds. Widely developed in the Machine Learning field, a *generative* model has the particularity of being able to generate new data from estimated *Probability Density Functions* (PDFs) (which are close to the real unknown one) by a sampling approach. Contrary to the classical profiled SCA, the state-of-the-art of Deep Learning based Side-Channel Analysis (DLSCA) only considers the *discriminative* models for key recovery objective. This solution differs from the generative model because it only learns boundaries between classes while the classical profiled SCA attempts to model the PDF of individual classes. Hence, generative models solve a more general problem than just guessing which subkey leaks the most. While the classical profiled SCA can be easily justified from a theoretical point of view, the DLSCA models, provided by the state-of-the-art, are very difficult to design and interpret. Thus, there are no intuitions on how those models should be configured, even against unprotected implementations. Indeed, even if a wide-range of architectures have been studied in DLSCA context (e.g. fully-connected neural networks [13,14], ResNets [8], transformer neural network [7], Support Vector Machine (SVM) [14], Random Forest[14]), there are no practical rules to construct them due to a lack of understanding between DL and SCA paradigms. Consequently, the discriminative models force the adversary to consider the DLSCA approach as black-box tools. Thus, bridging the gap between DL and SCA is essential to define the limitations of the DLSCA and provide clear improvements in this field. From this new starting point, an adversary can bring additional DL features that reduce the limitations provided by the classical profiled SCA approaches while preserving the interpretability and the explainability provided by the generative models.

Contributions. This paper falls into the class of works on machine-learning based cryptanalysis targeting real-world crypto-systems. In particular, we bridge DL and SCA paradigms by proposing the very first generative architecture which is based from theoretical results provided by stochastic attacks [18]. **This new**

model clarifies the links between DL and classical SCA issues (*i.e.* dimensionality reduction, needs of synchronization, higher-order attacks, multi-task learning). It represents a seminal contribution for further investigations and developments. In particular, the contributions of our work can be summarised as follows:

- We establish the first link between DLSCA and classical profiled SCA. By bridging both techniques, each field can benefit from the advantages of the other. Explainability of classical SCA can be transfered to DLSCA that was considered more or less as a black-box tool.
- We propose the Conditional Variational AutoEncoder based Stochastic Attacks (cVAE-SA) as a new neural network architecture that lies between stochastic attacks and DLSCA. Our models benefit from the theoretical aspects of stochastic attacks, as well as their ability to estimate and reconstruct the targeted leakage models. This analogy is helpful to ease the construction of the neural network as well as its interpretation.
- Thanks to its analogy with the stochastic attacks, we define some theoretical bounds related to the neural network complexity. It suggests that shallow neural networks can be sufficient to exploit the sensitive information induced in a trace. This result is in accordance with the Universal Approximation Theorem [15].
- We develop a new attack strategy based on similarity measure that allows an adversary to specifically choose which samples the model should target to retrieve the sensitive information. This results in a more flexible solution than classical profiled side-channel attacks.
- We validate all our theoretical results through a wide-range of use-cases including the following challenges in SCA context namely multi-task learning, curse of dimensionality, targeting masking scheme.
- Through a detailed experimental comparison of our cVAE-SA proposition with classical profiled attacks (*i.e.* template and stochastic attacks) as well as multiple DLSCA models, we highlight the benefits and the drawbacks of cVAE-SA. This results in a perspective about a new typology of models specific to the SCA context.

This proposition opens-up further research directions where improvements from both fields could be further combined for enhancing the attack efficiency as well as the explainability of the results.

2 Preliminaries

2.1 Notation & terminology

Basics on probability theory. Let calligraphic letters $\mathcal{X}$ denote sets such that, if $\mathcal{X}$ is finite, its cardinality, denoted $|\mathcal{X}|$, defines its number of elements. The corresponding capital letters X (resp. bold capital letters) denote random variables (resp. random vectors $\mathbf{T}$). The lowercase x (resp. $\mathbf{t}$) denote the realization of X (resp. $\mathbf{T}$). The i^{th} entry of a vector $\mathbf{T}$ is defined as $\mathbf{T}[i]$. The probability

of observing an event X is denoted by $\mathsf{Pr}[X]$ such that a conditional probability of observing an event X knowing an event Y is denoted $\mathsf{Pr}[X|Y]$. The moments of a random variable X are quantities providing information about the shape and location of its *Probability Mass Function* (discrete case) or its *Probability Density Function* (continuous case). $\mathbb{E}[X]$ is used to denote the *expected value* of a random variable X and $\mathbb{E}_{X\sim\mathcal{D}}$ defines under which probability distribution it is computed. In addition, the second central moment, also known as the *variance*, of a random variable X is defined as $\mathbb{V}[X]$. The symbol $\mathbb{V}_X[f(X)]$ (resp. $\mathbb{E}_X[f(X)]$) denotes the variance (resp. expected value) of a function f of the random variable X, over the distribution of X. The *standard deviation* of a random variable X, denoted σ_X, is defined as the square root of its variance.

A side-channel measurement will be constructed as a random vector $\mathbf{T} \in \mathbb{R}^D$ where D defines the dimension of the related trace. The targeted sensitive variable, denoted $Y = f(X, k^*)$, depends on a cryptographic primitive $f : \mathcal{X} \times \mathcal{K} \to \mathbb{F}_2^n$, a public variable $X \in \mathcal{X}$ (e.g. plaintext or ciphertext) and a part of the secret key $k^* \in \mathcal{K}$ (e.g. byte) that the adversary tries to retrieve. We define $\mathbf{X} \sim \mathcal{N}_D(\boldsymbol{\mu}, \Sigma)$ to denote a random vector $\mathbf{X}$ that follows a multivariate Gaussian distribution of parameters $\boldsymbol{\mu} \in \mathbb{R}^D$ and $\Sigma \in \mathcal{M}_{D,D}(\mathbb{R})$. In the rest of this paper, $\Sigma_{\mathbf{X}}$ (resp. $\boldsymbol{\mu}_{\mathbf{X}}$) denotes the covariance matrix (resp. mean) of a random variable $\mathbf{X}$.

Given p_X and q_X two probability distributions on $\mathcal{X}$, the Kullback-Leibler (KL) divergence measures how p_X differs from q_X such that:

$$\mathcal{D}_{\mathrm{KL}}\left(p_X||q_X\right) = \sum_{x\in\mathcal{X}} p_X\left[x\right] \log\left(\frac{p_X\left[x\right]}{q_X\left[x\right]}\right).$$

Usually, p_X denotes the measured probability distribution while q_X defines the theoretical model. The KL-divergence is always non-negative and equals zero if and only if $p_X = q_X$.

SCA terminology. SCA usually apply a divide-and-conquer strategy which consists in separately recovering differents parts of the N-bit (global) secret key $k^* = \|_{i=1}^{\frac{N}{n}} k_i^*$ considering n-bit subkeys $k_i^* \in \mathcal{K}$. For the rest of this paper, we will consider only attacking a subkey (*i.e.* $n = 8$), hence using k^* instead of k_i^* and referencing subkey as key in the rest of the paper. Given a variable Y and an independent noise $\mathbf{Z}$, a traces $\mathbf{T}$ is a D-dimensional random vector $\{\mathbf{T}[0], \ldots, \mathbf{T}[D-1]\}$ where $\mathbf{T}[i]$ represents the leakage of time sample i (for $0 \leq i < D$) satisfying the Gaussian Indepedent Noise Assumption[a]:

$$\mathbf{T} = \psi(Y) + \mathbf{Z} \tag{1}$$

where $\psi : \mathbb{F}_2^n \to \mathbb{R}$ is a pseudo-boolean function mapping a n-bit intermediate value Y which is generated from a cryptographic primitive $f : \mathcal{X} \times \mathcal{K} \to \mathbb{F}_2^n$. The latter corresponds to the deterministic part of the trace. In this work, we set $n = 8$. Let $\mathbf{Z} \sim \mathcal{N}_D(\boldsymbol{\mu}, \Sigma)$ corresponds to the noise which is characterized by a multivariate gaussian distribution parameterized by an unknown pair $(\boldsymbol{\mu}, \Sigma)$.

[a] This means that the Gaussian noise $\mathbf{Z}$ is independent of the variable Y.

2.2 Related works & limitations of discriminative models

Related works. Typically, to perform a DLSCA attack the adversary considers a discriminative approch which models the conditional posterior probabilities $\mathsf{Pr}[Y|\mathbf{T}]$ in order to discriminate and pick the most likely hypothetical candidate Y (*i.e.* sensitive information) given a trace $\mathbf{T}$. A discriminative model estimates a ϕ-parametric probability conditional distribution $\mathsf{Pr}[Y|\mathbf{T},\phi]$ that is as similar as possible to the true unknown joint probability distribution $\mathsf{Pr}[Y|\mathbf{T}]$. This approach is beneficial for directly solving a classification problem without modeling unnecessary information and thus, mitigating the impact of some countermeasures such as the desynchronization effect [3,7,21].This reason leads the side-channel community to investigate the DL approaches to improve the profiled SCA [1,3,9,21] relying on discriminative approach. More specifically, all the DLSCA models proposed in the state-of-the-art are based on such approach (e.g. fully-connected neural networks [13,14], ResNets [8], transformer neural network [7]). However, due to the lack of theoretical results, the discriminative models can be seen as black-box tools, and the design of models can be a real challenge even against unprotected cryptographic implementations. To reduce this issue, some solutions which automatically tune model hyperparameters have been investigated [1,11,17] but the related process is time-consuming, the range of the hyperparameters' values are randomly bounded such that a poor design of the model can be highly impacted by underfitting/overfitting issues. This paper reduces this issue by providing the first DL model based on SCA theoretical result in order to make the construction phase easier.

Generative approach. An alternative solution consists in considering a probabilistic generative approach which captures the interactions between all the variables considered by the resulted learning algorithm. To comply with this technical specification, this strategy builds a model that estimates the probability distribution of the traces. To fit with SCA context, the conditional probability distribution, $\mathsf{Pr}[\mathbf{T}|Y]$, has to be estimated such that, afterwards, the Bayes' theorem can be computed in order to retrieve the conditional posterior probabilities $\mathsf{Pr}[Y|\mathbf{T}]$ and pick the most likely label Y. More concretely, a generative model can be viewed as an estimation of a Θ-parametric conditional distribution $\mathsf{Pr}[\mathbf{T}|Y,\Theta]$ that is as similar as possible to the true unknown conditional distribution $\mathsf{Pr}[\mathbf{T}|Y]$. The classical profiled SCAs, such as the stochastic attacks [18], follow this approach by building a model, *i.e. leakage model*, that estimates the class-conditional probability distributions (*i.e.* $\mathsf{Pr}[\mathbf{T}|Y]$) for each possible value of a sensitive intermediate variable. One benefit of this method is the ability to explain and interpret the result provided by the model. The following section summarizes the stochastic attacks [18] introduced by Schindler *et al.* as well as the new model that we detail in Sec.3.

2.3 Overview of this work

A short description of stochastic attacks [18]. Given a trace $\mathbf{T}$ such that its i^{th} time sample can be defined as $\mathbf{T}[i] = \psi_i(f(X,k^*)) + \mathbf{Z}[i]$, the goal of the

stochastic attack is to find an approximation of the leakage model, denoted $\hat{\psi}_i$, as close as possible to the true unknown ψ_i. As ψ_i is assumed to be a pseudo-boolean function, ψ_i can be viewed as a linear combination of monomial basis' vectors $u \in \mathbb{F}_2^n$. Hence, there exists a set of real coefficients $(\alpha_u)_{u \in \mathbb{F}_2^n}$ such that, for a sensitive intermediate value $Y \in \mathbb{F}_2^n$, the *leakage model* (see Eq.1) is redefined as:

$$\hat{\psi}_{i,\alpha}(Y) = \sum_{u=(u[0],\ldots,u[n-1]) \in \mathbb{F}_2^n} \alpha_u[i] \cdot Y^u, \tag{2}$$

where Y^u denotes the *monomial basis* and characterizes the conjunction of all bits of Y such that $Y^u = \prod_{j=0}^{n-1} Y[j]^{u[j]}$ where $Y[j] \in \mathbb{F}_2$ defines the j^{th} bit of Y and the power notation is simply $Y[j]^0 = 1$ and $Y[j]^1 = Y[j]$. In other words, ψ_i can be approximated as a multivariate polynomial in the bit-coordinate $Y[j]$ with coefficients in $\mathbb{R}$. The degree d (s.t. $d \leq n$) of such monomial is defined as the maximal number of bits' interaction induced in $\hat{\psi}_{i,\alpha}(Y)$. In particular, this degree d can be viewed as logical operator (e.g. AND or XOR). The related subspace is denoted by $\mathcal{F}_{d+1}$. For the profiling phase, the stochastic attack mechanism consists firstly in choosing the degree d of the pseudo-boolean function $\hat{\psi}_\alpha$, and then in estimating the leakage model for each hypothetical key $k \in \mathcal{K}$. Given a set of N_p labeled traces $\mathcal{I}_p = \{(\mathbf{t}_0, y_0), \ldots, (\mathbf{t}_{N_p-1}, y_{N_p-1})\}$, the adversary estimates the leakage model $(\hat{\psi}_{i,\alpha}(Y))_{Y \in \mathbb{F}_2^n}$ by finding the best set of coefficients $(\alpha_u[i])_{u \in \mathbb{F}_2^n}$ through the application of the *ordinary least squares* (OLS) method. The set of coefficients $(\alpha_u[i])_{u \in \mathbb{F}_2^n}$ which minimizes the OLS are called the *OLS estimator* for ψ. More details on how to practically implement stochastic attacks can be found in [5]. While the basis choice is essential for efficient profiling phase, *i.e.* having a good approximation of the *leakage model*, the application of *gradient descent* method for minimizing the OLS method is an interesting alternative to the classical approach (see [18, Eq.13]) and will be explored in next sections. This leads to get a better intuition into how a DL model should be designed in order to extract the sensitive information and results in a more flexible solution during the exploitation phase.

A new generative strategy in DLSCA. In SCA context, we want to explicitly compute an approximation of the true unknown conditional probability distribution $\mathsf{Pr}[\mathbf{T}|Y]$ in order to retrieve the secret key that is manipulated by the targeted real-world crypto-system. In 2014, Kingma and Welling introduced the *Variational AutoEncoder* (VAE) [10] as a solution to this issue outside of the SCA context. In this paper, we develop a new usage of variational autoencoders for DLSCA and we present our main contribution: the *Conditional Variational AutoEncoder based on Stochastic Attacks* (cVAE-SA). In particular, this new approach models a Θ-parametric conditional distribution $\mathsf{Pr}[\mathbf{T}|Y, \Theta]$ through the design of two distinct parts referred as *encoder* and *decoder*.

First, an encoder characterizes the noise $\mathbf{Z}$ included in a trace (see Eq.1) by estimating a small set of samples that depend on the targeted sensitive variable, *i.e.* the Points of Interest (PoIs). To conduct such a characterization, a KL-

divergence loss is used in order to find the trainable parameters that fit the most with the true unknown solution.

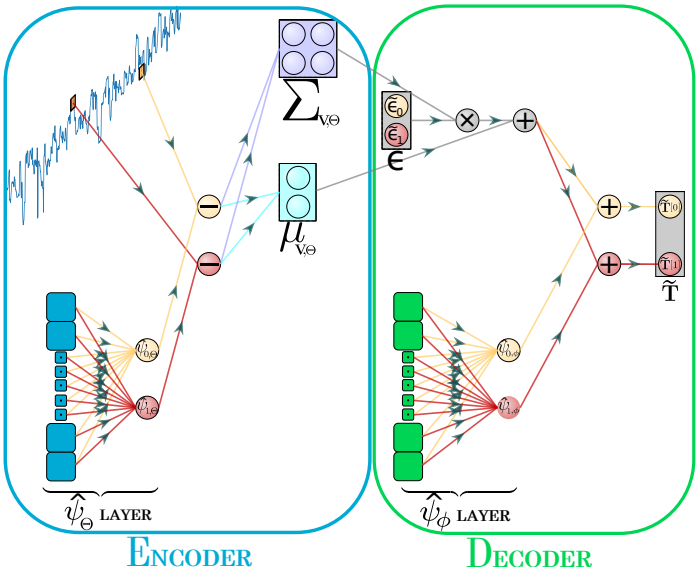


Fig. 1: cVAE-SA structure.

Then, a decoder is defined in order to generate a synthetic trace, based on a reconstruction loss, from the noise estimated by the encoder, and an approximation of the deterministic part $\psi(Y)$ defined in Eq.1. An overview of the model is provided in Fig.1, and the details of the cVAE-SA are defined in the following section.

Once the cVAE-SA is designed, it is automatically configured over a set of training traces in order to estimate the Θ-parametric conditional distribution $\mathsf{Pr}[\mathbf{T}|Y,\Theta]$ that should be as similar as possible to the true condition distribution $\mathsf{Pr}[\mathbf{T}|Y]$. Based on this configured model, the adversary can compute the maximum likelihood over a set of attack traces in order to retrieve the most likely subkey candidate over $\mathbb{F}_2^n$ such that $n = 8$.

3 Conditional Variational AutoEncoder based on Stochastic Attacks

Through this section, we explain the link between generative DL models and classical profiled SCA by building a new type of VAE. Sec.3.1 introduces the problem we want to solve and proposes a first link with SCA. Sec.3.2 explains our architecture and the theoretical link with the work provided by Schindler *et al.* [18], known as the stochastic attacks. Then, Sec.3.3 describes the training process of cVAE-SA and the relation with similarity measures. Finally, Sec.3.4 describes the attack phase and the theoretical architecture complexity bounds.

3.1 Generative latent variable models

We propose to contextualize Conditional VAE (cVAE) [19] into SCA in order to give a new perspective for generative models. Supported by theoretical aspects of stochastic attacks, this new approach can be considered as an alternative to classical discriminative models often used in DLSCA.

Problem statement. The cVAE aims at modeling a Θ-parametric conditional distribution $\mathsf{Pr}[\mathbf{T}|Y,\Theta]$ from two random variables $\mathbf{T} \in \mathbb{R}^D$ and $Y \in \mathbb{F}_2^n$. Suppose that a trace $\mathbf{T} \in \mathbb{R}^D$ is acquired by assuming that all the time samples are sequentially generated such that its assigned label only depends on a small set of time samples (*i.e.* PoIs). As the cVAE is a *latent variable model*, which suggests that the variability in the traces given a label Y can be captured by a small finite set of time samples, its applicability in the SCA context fits well. By designing such models for performing SCA, we thus want to capture the interactions between the time samples *via* the characterization of a latent space $\mathcal{V}$. In particular, a Θ-parametric latent variable model F_Θ, providing a Θ-parametric conditional distribution $\mathsf{Pr}[\mathbf{T}|Y,\Theta]$, is representative of the true unknown conditional distribution $\mathsf{Pr}[\mathbf{T}|Y]$, for every trace $\mathbf{T}$ and every given sensitive variable Y, if there is a representation of compressed data $\mathbf{V} \in \mathcal{V}$, also known as latent space representation, such that the marginal distribution is given by:

$$\mathsf{Pr}\left[\mathbf{T}|Y,\Theta\right] = \int_{\mathbf{v}\in\mathcal{V}} \mathsf{Pr}\left[\mathbf{T}|Y,\mathbf{v},\Theta\right] \mathsf{Pr}\left[\mathbf{v}|Y\right] d\mathbf{v}, \tag{3}$$

where $\mathbf{v}$ is the realization of a random variable $\mathbf{V}$ in a $D^{'}$-dimensional space $\mathcal{V}$, with a probability $\mathsf{Pr}[\mathbf{V}=\mathbf{v}]$ defined over $\mathcal{V}$, and $\mathsf{Pr}[\mathbf{V}=\mathbf{v}|Y]$ denotes the probability of observing $\mathbf{v}$ over the latent space $\mathcal{V}$ knowing Y. While the latent space characterizes the noise distribution defined in Eq.1 (*deeper details will be provided in the following paragraphs*), we can easily assume that $\mathsf{Pr}[\mathbf{V}|Y] = \mathsf{Pr}[\mathbf{V}]$ because $\mathbf{V}$ is independent of the label Y. In addition, we assume that $\mathsf{Pr}[\mathbf{V}]$ follows a multivariate Gaussian distribution of parameters $(\boldsymbol{\mu}, \Sigma)$. Through the conditional variational autoencoder, we wish to optimize Θ such that we can sample $\mathbf{V}$ from $\mathsf{Pr}[\mathbf{V}]$, and obtain, with high probability, a new generated trace $\tilde{\mathbf{T}}$ very similar to the true known $\mathbf{T}$.

Intractability. However, Eq.3 is unfortunately intractable as it should be computed for every latent representations induced by the latent space $\mathcal{V}$. Thus, the following part of the section proposes solutions to circumvent this issue. Hopefully, $\mathsf{Pr}[\mathbf{T}|Y,\Theta]$ may still be efficiently approximated thanks to the *Monte-Carlo* method. Hence, for a large number of samples $\{\mathbf{v}_0,\ldots,\mathbf{v}_{N_v}\}$, a trace $\mathbf{T}$ and a label Y, we can compute an estimation of $\mathsf{Pr}[\mathbf{T}|Y]$. As a consequence, for a given label Y and a latent variable $\mathbf{V}$, we can build a neural network that computes $\mathsf{Pr}[\mathbf{T}|Y,\mathbf{V},\Theta]$. This model, denoted $F_\Theta^{(dec)} : \mathbb{R}^{D^{'}} \times \mathbb{F}_2^n \to \mathbb{R}^D$, is named *the decoder*. Given a latent variable $\mathbf{V} \in \mathbb{R}^{D^{'}}$ and a sensitive variable $Y \in \mathbb{F}_2^n$, the decoder generates a new trace $\tilde{\mathbf{T}}$ as close as possible to the real observed trace $\mathbf{T}$.

However, to perfectly construct a new set of D-dimensional traces from $F_{\Theta}^{(dec)}$, we have to estimate the multivariate Gaussian distribution characterizing the latent space $\mathcal{V}$ by approximating the following probability distribution $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y]$. This probability is defined as $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y] = \frac{\mathsf{Pr}[\mathbf{T}|\mathbf{V},Y]\cdot\mathsf{Pr}[\mathbf{V}]}{\mathsf{Pr}[\mathbf{T}|Y]}$ and it is also intractable due to Eq.3. Consequently, a solution is to find a parametric model that approximates the true unknown posterior $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y]$. In statistics, the variational inference techniques can approximate such complex distributions. Given a trace $\mathbf{T}$ and a label $Y \in \mathbb{F}_2^n$, a Θ-parametric model can be constructed to estimate the parameters $\boldsymbol{\mu}_{\mathbf{V}} \in \mathbb{R}^{D^{'}}$ and $\Sigma_{\mathbf{V}} \in \mathcal{M}_{D',D'}(\mathbb{R})$ of the multivariate Gaussian distribution $\mathcal{N}_{D'}(\boldsymbol{\mu}_{\mathbf{V}}, \Sigma_{\mathbf{V}})$ such that the KL-divergence between the approximation and the targeted probability distribution $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y]$ is minimized. The Θ-parametric model, denoted $F_{\Theta}^{(enc)} : \mathbb{R}^D \times \mathbb{F}_2^n \rightarrow \mathbb{R}^{D^{'}}$, is called *the encoder*. In the rest of this paper, we denote as $F_{\Theta,\phi}$ the resulted cVAE-SA such that, for a given trace $\mathbf{T}$ and a given label Y, $F_{\Theta,\phi}(\mathbf{T}, Y) = F_{\phi}^{(dec)} \circ (F_{\Theta}^{(enc)}(\mathbf{T}, Y), Y)$. Furthermore, as the aim of this paper is to bridge the DL and the classical profiled SCA, no particular focus will be proposed on dimensionality reduction techniques. Thus, the following part assumes that $D^{'} = D$.

3.2 Latent space estimation and instances' generation

Through the description of the stochastic attack (see Sec.2.3), the adversary can construct a conditional variational autoencoder adapted for the SCA context.

Encoder. As mentioned in Sec.2.3), the encoder models a neural network $F_{\Theta}^{(enc)} : \mathbb{R}^D \times \mathbb{F}_2^n \rightarrow \mathbb{R}^D$ which returns an approximation of $\boldsymbol{\mu}_{\mathbf{V}} \in \mathbb{R}^D$ and $\Sigma_{\mathbf{V}} \in \mathcal{M}_{D,D}(\mathbb{R})$ such that it can be used to characterize an element in the latent space $\mathcal{V}$ of dimension D. This element describes the behavior of the targeted crypto-system (see on the left part of Fig.1). In this regard, we design the encoder $F_{\Theta}^{(enc)}$ such that it characterizes the leakage model $\psi(Y)$ and the random part $\mathbf{Z}$ of a trace $\mathbf{T}$ in order to fit with the stochastic attack process. To build a suited encoder, the related neural network should follow the structure defined in Sec.2.3 in order to extract the maximum amount of relevant information from $\mathbf{T}$. First, the adversary has to estimate the deterministic part of a trace $\mathbf{T}$ (*i.e.* leakage model ψ) that is defined by Eq.2. This modeling can be estimated by a fully-connected layer of D neurons such that each of them is linked with all elements of the monomial basis $(Y^u)_{u\in\mathbb{F}_2^n}$. Let $Y \in \mathbb{F}_2^n$ and $(Y^u)_{u\in\mathbb{F}_2^n}$ (resp. $\hat{\psi}_{i,\Theta}(Y)$) be the input (resp. output) of the i^{th} neuron such that:

$$\hat{\psi}_{i,\Theta}(Y) = \varrho\left(\sum_{u=(u[0],\ldots,u[n-1])\in\mathbb{F}_2^n} \Theta_u[i]\cdot Y^u\right),$$

where $\varrho(.)$ is a function (linear or non-linear) and $\Theta \in \mathcal{M}_{\sum_{i=0}^{d} 1+\binom{n}{i},D}(\mathbb{R})$ denotes the set of trainable parameters for a given degree d that characterizes the space

$\mathcal{F}_{d+1}$. While the goal of our work is to reduce the gap between deep learning and classical profiled SCA, we define $\varrho(.)$ as the identity function in order to satisfy $\hat{\psi}_{\Theta}[i] = \hat{\psi}_{\alpha}[i]$ and consider that the deterministic part of a trace at time sample i can be approximated by a single neuron. In the rest of this paper, this layer will be denoted as $\hat{\psi}_{\Theta}$ (see Fig.1).

Once the noise-free part $\hat{\psi}_{\Theta}$ is estimated, the next step is to deeply characterize the noise part $\mathbf{Z}$ using traces and the neurons of $\hat{\psi}_{\Theta}$ layer. In the cVAE-SA, we choose to deliberately force the subtraction of the traces at time sample i and the i^{th} neuron of $\hat{\psi}_{\Theta}$ layer in order to fit with Eq.1. Then, the encoder $F_{\Theta}^{(enc)}$ is trained to return a Θ-parametric mean vector $\boldsymbol{\mu}_{\mathbf{V},\Theta} \in \mathbb{R}^D$, and a Θ-parametric covariance matrix $\Sigma_{\mathbf{V},\Theta} \in \mathcal{M}_{D,D}(\mathbb{R})$ that describes the multivariate Gaussian noise for a given trace $\mathbf{T}$. Those approximations respectively estimate $\boldsymbol{\mu}_{\mathbf{V}}$ and $\Sigma_{\mathbf{V}}$. Thus, from these parameters, the adversary can compute $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$. That is, $F_{\Theta}^{(enc)}$ extracts all the information induced in a trace such that the latent variable $\mathbf{V} \in \mathcal{V}$ is characterized by its noise part $\mathbf{Z}$.

Decoder. Once the encoder is constructed, the adversary can capture the parameters $\boldsymbol{\mu}_{\mathbf{V},\Theta}$ and $\Sigma_{\mathbf{V},\Theta}$ which are needed to design the related Gaussian noise distribution $\mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$. From a new sample $\mathbf{V} \sim \mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$, the adversary designs a decoder $F_{\phi}^{(dec)} : \mathbb{R}^D \times \mathbb{F}_2^n \rightarrow \mathbb{R}^D$ (see on the right part of Fig.1) such that given a trace $\mathbf{T}$ and the subspace $\mathcal{F}_{d+1}$, containing all the pseudo-boolean functions of degree lower or equal to d, he wants to maximize the conditional probability distribution $\mathsf{Pr}[\mathbf{T}|Y, \mathbf{V}, \phi]$, *i.e.* building a new trace $\tilde{\mathbf{T}} \in \mathbb{R}^D$ as similar as possible to the related real trace $\mathbf{T}$ and defined as follows:

$$\tilde{\mathbf{T}}[i] = \underbrace{\sum_{u=(u[0],\ldots,u[n-1])\in\mathbb{F}_2^n} \phi_u[i] \cdot Y^u}_{\hat{\psi}_{i,\phi}} + \mathbf{V}[i]. \tag{4}$$

Note that a latent variable $\mathbf{V} \in \mathbb{R}^D$ is initially sampled from the prior distribution $\mathsf{Pr}[\mathbf{V}]$ such that the dimension of $\mathbf{V}$ should correspond with the dimension of the latent space estimated by the encoder. However, performing the training process in such configuration can be arduous. Indeed, during the training process, the backpropagation cannot be performed because the adversary has to compute the gradient of the loss function with respect to samples (*i.e.* latent variable $\mathbf{V} \in \mathcal{V}$), which is inherently non-differentiable. To circumvent this issue, the reparametrization trick [10] proposes to rewrite $\mathbf{V}$ such that the derivative can be computed with respect to the parametric distributions (*i.e.* $\boldsymbol{\mu}_{\mathbf{V},\Theta}$ and $\Sigma_{\mathbf{V},\Theta}$) that are differentiable. Instead of generating samples from $\mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$, sampling is performed from $\epsilon \sim \mathcal{N}_D(\mathbf{0}, \mathbf{I}_D)$, followed by the computation of $\mathbf{V} = \boldsymbol{\mu}_{\mathbf{V},\Theta} + \Sigma_{\mathbf{V},\Theta}^{\frac{1}{2}} \times \epsilon$.

Once $\mathbf{V}$ is constructed, the adversary has to approximate the deterministic part of the leakage model, namely $\hat{\psi}_{\phi}$. As already mentioned for the encoder, its estimation can be made with a fully-connected layer such that the input of size D is characterized by $(Y^u)_{u\in\mathbb{F}_2^n}$ for a given $Y \in \mathbb{F}_2^n$. Because the adversary

wants to characterize all the input time samples, the number of nodes in the $\hat{\psi}_\phi$ layer depends on the dimensionality of the latent space, *i.e.* dimension of $\mathcal{V}$ (see Fig.1). Based on $\hat{\psi}_\phi$ and $\mathbf{V}$, the adversary can then build a new trace $\tilde{\mathbf{T}}$ following Eq.4.

Then, to adequately find the trainable parameters Θ and ϕ, the adversary has to consider some learning metrics that aims at approximating $\mathsf{Pr}[\mathbf{T}|Y]$.

3.3 Similarity maximization

As defined in Sec.3.2, our generative model has to optimize a set of parameters ϕ and Θ in order to maximize the marginal log-likelihood $\log(\mathsf{Pr}[\mathbf{T}|Y,\phi])$.

Theorem 1. *For any choice of encoder $F_\Theta^{(enc)}$ and trainable parameters Θ, the conditional marginal log likelihood $\log(\mathsf{Pr}[\boldsymbol{T}|Y,\phi])$ can be defined as:*

$$\begin{aligned}\log\left(\mathsf{Pr}\left[\boldsymbol{T}|Y,\phi\right]\right) = \mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}}\left[\log\left(\mathsf{Pr}\left[\boldsymbol{T},\mathbf{v}|Y,\phi\right]\right) - \log\left(\mathsf{Pr}\left[\mathbf{v}|\boldsymbol{T},Y,\Theta\right]\right)\right] \\ + \mathcal{D}_{KL}\left(\mathsf{Pr}\left[\boldsymbol{V}|\boldsymbol{T},Y,\Theta\right]||\mathsf{Pr}\left[\boldsymbol{V}|\boldsymbol{T},Y,\phi\right]\right).\end{aligned} \tag{5}$$

Unfortunately, due to the intractability of $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\phi]$ (see Sec.3.1), Eq.5 cannot be solved in practice. Hence, we have to define a function such that $\log(\mathsf{Pr}[\mathbf{T}|Y,\phi])$ can be approximated through an optimization algorithm. In [10], Kingma and Welling propose a variational lower bound on the marginal likelihood which was generalized by Sohn *et al.* on conditional marginal likelihood [19].

Theorem 2. *[19] For any choice of encoder $F_\Theta^{(enc)}$ and trainable parameters Θ, the variational lower bound of $\log(\mathsf{Pr}[\boldsymbol{T}|Y,\phi])$ is defined as:*

$$\begin{aligned}\log\left(\mathsf{Pr}\left[\boldsymbol{T}|Y,\phi\right]\right) \geq -\mathcal{D}_{KL}\left(\mathsf{Pr}\left[\boldsymbol{V}|\boldsymbol{T},Y,\Theta\right]||\mathsf{Pr}\left[\boldsymbol{V}\right]\right) \\ + \mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}}\left[\log\left(\mathsf{Pr}\left[\boldsymbol{T}|Y,\mathbf{v},\phi\right]\right)\right].\end{aligned} \tag{6}$$

The equality between Eq.5 and Eq.6 holds if and only if the encoder $F_\Theta^{(enc)}$, which approximates the parameters $\boldsymbol{\mu}_{\mathbf{V},\Theta}$ and $\Sigma_{\mathbf{V},\Theta}$ that are needed to compute $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\Theta]$, is able to perfectly predict $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\phi]$. In such configuration, the adversary exactly captures the random part induced in a trace $\mathbf{T}$. Based on Eq.6, we define the empirical risk that we minimize to train the cVAE-SA.

Definition 1 (Empirical risk combined with Evidence Lower BOund (ELBO) Loss). *Given a latent space $\mathcal{V}$, a set of N_p labeled traces $\mathcal{I}_p = \{(\boldsymbol{t}_0, y_0), \ldots, (\boldsymbol{t}_{N_p-1}, y_{N_p-1})\}$, we define the empirical risk optimizing $F_{\Theta,\phi}$, that approximates the generative distribution $\mathsf{Pr}[\boldsymbol{T}|Y]$, as follows:*

$$\begin{aligned}\hat{\mathcal{R}}\left(\mathcal{L}_{ELBO}, F_{\Theta,\phi}\right) = \frac{1}{N_p}\sum_{i=0}^{N_p-1}\underbrace{\mathcal{D}_{KL}\left(\mathsf{Pr}\left[\boldsymbol{V}|\boldsymbol{t}_i,y_i,\Theta\right]||\mathsf{Pr}\left[\boldsymbol{V}\right]\right)}_{\textit{KL-Divergence Loss}} \\ \underbrace{-\,\mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}}\log\left(\mathsf{Pr}[\boldsymbol{t}_i|y_i,\mathbf{v},\phi]\right)}_{\textit{Reconstruction Loss}},\end{aligned}$$

such that $(\mathsf{Pr}[\boldsymbol{V}|\boldsymbol{t}_i, y_i, \Theta])_{0\leq i<N_p}$ *is computed from* $\boldsymbol{\mu}_{\boldsymbol{V},\Theta}$ *and* $\Sigma_{\boldsymbol{V},\Theta}$ *provided by the encoder* $(F_\Theta^{(enc)}(\boldsymbol{t}_i))_{0\leq i<N_p}$ *and* $(\mathsf{Pr}[\boldsymbol{t}_i|y_i, \mathbf{v}, \phi])_{0\leq i<N_p}$ *is obtained from* $F_\phi^{(dec)}(\mathbf{v})$.

Sampling $\mathbf{v}$ from the learned posterior $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$ knowing the trace $\mathbf{T}$, the related label Y and the multivariate Gaussian distribution $\mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$, can be seen as encoding $\mathbf{T}$ into $\mathbf{v}$, while $F_\phi^{(dec)}$ seeks to reconstruct $\mathbf{T}$ from $\mathbf{v}$. Classically used for training a variational autoencoder, the loss function defined in Def.1 can be decomposed into two terms: the reconstruction and the KL-divergence terms. To minimize the reconstruction loss, the embedding means $\boldsymbol{\mu}_{\mathbf{V},\Theta}$ are pushed far away from each other and embedding standard deviations $\Sigma_{\mathbf{V},\Theta}$ are pulled toward zero. To get smaller $\mathcal{D}_{\mathrm{KL}}(\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]||\mathsf{Pr}[\mathbf{V}])$, the embedding means are pulled toward zero and the embedding standard deviations are pulled toward one. While the KL-divergence term is opposed to the reconstruction loss, it can be seen as a regularization term. Indeed, putting a lot of information about $\mathbf{T}$ in $\mathbf{V}$ makes reconstruction trivial, but the penalization induced by the regularization term is non-negligible. Therefore, the regularization term acts as an information bottleneck, so a balance between both terms must be found. If necessary, the KL-divergence loss can be monitored by a hyperparameter β. In the state-of-the-art, these models are called β-*Variational AutoEncoders*. However, the impact of the β-parameter on the resulted learning algorithm is considered as out of the scope of this paper.

Reconstruction loss. This term, denoted by $\mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}} \log(\mathsf{Pr}[\mathbf{T}|Y, \mathbf{v}, \phi])$, is linked with the decoder $F_\phi^{(dec)}$ introduced in Sec.3.2. It defines the probability of constructing $\mathbf{T}$ given the label Y and a sample $\mathbf{v}$ of the latent space $\mathcal{V}$ of size D. Hence, the reconstruction loss tends to maximize the log likelihood in order to construct traces that are correlated with the true unknown leakage model ψ and the noise $\mathbf{Z}$ related to $\mathbf{T}$. Thus, it encourages the decoder to learn how a trace can be reconstructed from a given noise representation defined by a latent variable $\mathbf{V}$. The reconstruction loss optimizes the parameters ϕ to retrieve the correct coefficients associated with each vector of the monomial basis $(Y^u)_{u\in\mathbb{F}_2^n}$ such that the probability distribution in latent space $\mathcal{V}$ is defined by a multivariate Gaussian distribution $\mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$. Typically, if we only consider the case where no interaction between the time samples of $\mathbf{T}$ occurs, then, the covariance matrix $\Sigma_{\mathbf{V},\Theta}$ can be simplified to a diagonal matrix such that its vector representation can be described as $\sigma^2_{\mathbf{V},\Theta} = [\Sigma_{\mathbf{V},\Theta}[0,0], \Sigma_{\mathbf{V},\Theta}[1,1], \ldots, \Sigma_{\mathbf{V},\Theta}[D,D]]$. In such configuration, we do not expect to capture the time samples' interaction related to the constructed trace $\tilde{\mathbf{T}} \in \mathbb{R}^D$. Thus, the reconstruction loss can be computed as follows:

$$\mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}} -\log\left(\mathsf{Pr}\left[\mathbf{T}|Y, \mathbf{v}, \phi\right]\right) = \mathbb{E}_{\mathbf{v}\sim F_\Theta^{(enc)}}\left[\sum_{i=0}^{D-1} \frac{1}{2}\log\left(2\pi\sigma^2_{\tilde{\mathbf{T}}}[i]\right) + \frac{(\mathbf{T}[i] - \boldsymbol{\mu}_{\tilde{\mathbf{T}}}[i])^2}{2\sigma^2_{\tilde{\mathbf{T}}}[i]}\right], \tag{7}$$

where $\mu_{\tilde{\mathbf{T}}}[i]$ (resp. $\sigma^2_{\tilde{\mathbf{T}}}[i]$) indicates the i^{th} element of the mean (resp. variance) vector of generated traces $\tilde{\mathbf{T}}$ given a latent representation $\mathbf{v}$ and a deterministic part $\hat{\psi}_\phi$ which depends on $Y = f(X, k^*)$ (see Eq.4). However, assuming that $\Sigma_{\mathbf{V},\Theta}$ can be simplified to a diagonal matrix affects the ability of the generated trace $\tilde{\mathbf{T}}$ to capture the interaction between the time samples of $\mathbf{T}$. While this choice can be problematic from a performance perspective[b], the computation gain is non-negligible as the matrix inversion does not have to be computed in order to process the reconstruction loss. Then, we assume that the output distribution of the conditional variational autoencoder is an isotropic Gaussian (*i.e.* for all $\mathbf{v} \sim \mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \text{diag}(\Sigma_{\mathbf{V},\Theta}))$, we can define $\Sigma_{\tilde{\mathbf{T}}} = \sigma^2 \cdot \mathbf{I}_D$ where σ^2 is a scalar). While the *Mean Squared Error* (MSE) loss function can be written as $\mathbb{E}_{\mathbf{v}\sim F^{(enc)}_\Theta} ||\mathbf{T} - \boldsymbol{\mu}_{\tilde{\mathbf{T}}}||_2$, Eq.7 can be simplified as follows:

$$\mathbb{E}_{\mathbf{v}\sim F^{(enc)}_\Theta} - \log\left(\mathsf{Pr}\left[\mathbf{T}|Y, \mathbf{v}, \phi\right]\right) = \frac{1}{2}\log\left(2\pi\sigma^2\right) + \frac{\mathbb{E}_{\mathbf{v}\sim F^{(enc)}_\Theta} ||\mathbf{T} - \boldsymbol{\mu}_{\tilde{\mathbf{T}}}||_2}{2\sigma^2}. \quad (8)$$

Note that this solution is minimized if the scalar $\sigma^2 = \mathbb{E}_{\mathbf{v}\sim F^{(enc)}_\Theta} ||\mathbf{T} - \boldsymbol{\mu}_{\tilde{\mathbf{T}}}||_2 =$ MSE [20].

This loss is approximated via *Monte-Carlo* sampling, however, due to computation constraints, we consider only one sample $\mathbf{v}$ for computing Eq.8 during the training process. Consequently, for an estimated trace $\tilde{\mathbf{T}}$, we minimize its L^2-norm from the related true trace $\mathbf{T}$ in order to find the best parameters ϕ. In other words, through this solution, we attempt to find an estimated trace $\tilde{\mathbf{T}}$ as similar as the real one $\mathbf{T}$. Thus, the decoder $F^{(dec)}_\phi$ is only affected by the reconstruction loss and seeks to suitably reconstruct $\tilde{\mathbf{T}}$ based on a latent representation $\mathbf{V}$ and a deterministic part $\hat{\psi}_\phi$.

KL-divergence loss. However, to reduce the overfitting issue, a *regularization* term is added. In addition to the optimization of ϕ, the cVAE concurrently optimizes Θ to minimize the KL-divergence of the approximation $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$ from $\mathsf{Pr}[\mathbf{V}]$. In addition, the better $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$ approximates the true posterior distribution $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \phi]$, in terms of the KL divergence, the smaller the gap between $\hat{\mathcal{R}}(\mathcal{L}_{ELBO}, F_{\Theta,\phi})$ and the marginal log-likelihood $\log(\mathsf{Pr}[\mathbf{T}|Y, \phi])$. As remainder, both $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$ and $\mathsf{Pr}[\mathbf{V}]$ are assumed to be Gaussian, specifically, $\mathsf{Pr}[\mathbf{V}|\mathbf{T}, Y, \Theta]$ follows $\mathcal{N}_D(\boldsymbol{\mu}_{\mathbf{V},\Theta}, \Sigma_{\mathbf{V},\Theta})$ and $\mathsf{Pr}[\mathbf{V}]$ follows $\mathcal{N}_D(0, \mathbf{I}_D)$. The latter distribution is assumed as the traces are standardized, *i.e.* zero mean and unit variance, and such that no interactions are captured between the time samples. Indeed, as $\mathsf{Pr}[\mathbf{V}]$ characterizes the random part of $\tilde{\mathbf{T}}$ (see Eq.4), it has to follow the same distribution as the random part of the real trace $\mathbf{T}$ which is $\mathcal{N}(0, 1)$ for each non-informative time sample. Through this configuration, the

[b] To nuance this issue, Bruneau *et al.* [2, Fig.3] illustrate that the information induced by the covariance matrix is mainly brought by its diagonal.

KL-divergence can be computed as follows:

$$\begin{aligned} D_{\mathrm{KL}}(\mathsf{Pr}\left[\mathbf{V}|\mathbf{T},Y,\Theta\right]||\mathsf{Pr}\left[\mathbf{V}\right]) &= \mathbb{E}_{\mathbf{v}\sim F_{\Theta}^{(enc)}}\left[\log(\mathsf{Pr}\left[\mathbf{v}|\mathbf{T},Y,\Theta\right]) - \log(\mathsf{Pr}\left[\mathbf{v}\right])\right] \\ &= \frac{1}{2}\left(-\log\left(|\Sigma_{\mathbf{V},\Theta}|\right) - \mathrm{tr}(\mathbf{I}_D) + \boldsymbol{\mu}_{\mathbf{V},\Theta}^T \cdot \boldsymbol{\mu}_{\mathbf{V},\Theta} + \mathrm{tr}(\Sigma_{\mathbf{V},\Theta})\right) \\ &= -\frac{1}{2}\sum_{i=0}^{D-1}\left(1 + \log\left(\sigma^2_{\mathbf{V},\Theta}[i]\right) - \boldsymbol{\mu}^2_{\mathbf{V},\Theta}[i] - \sigma^2_{\mathbf{V},\Theta}[i]\right). \end{aligned}$$

The last simplification can be made as $\Sigma_{\mathbf{V},\Theta}$ can be rewritten as a vector $\sigma^2_{\mathbf{V},\Theta}$ such that each element of $(\sigma^2_{\mathbf{V},\Theta}[i])_{0\leq i<D}$ defines the i^{th} diagonal of $\Sigma_{\mathbf{V},\Theta}$. While the KL-divergence measures how the $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\Theta]$ distribution is different from $\mathsf{Pr}[\mathbf{V}]$, standardizing input traces is helpful to reduce the impact of irrelevant time samples.

To clearly explain the impact of the KL-divergence loss on the learning process, let us denote $\mathbf{T}$ a D-dimensional trace that has been standardized at each sample. Let $\{l_0,\ldots,l_{s-1}\}$ defines a set of indices where the sensitive information leaks (*i.e.* PoIs) such that,

$$\mathbf{T}[i] = \begin{cases} \psi(Y)[i] + \mathbf{Z}[i] \ \sim\ \mathcal{N}(0,1) & \text{if} \quad i \in \{l_0,\ldots,l_{s-1}\}, \\ \mathbf{Z}[i] \ \sim\ \mathcal{N}(0,1) & \text{otherwise.} \end{cases}$$

In this setting, we assume that the interactions between trace samples are negligible. Thus, if the trace $\mathbf{T}$ is standardized, each time sample has a zero mean and unit variance. Hence, if the i^{th} time sample of $\mathbf{T}$ has no deterministic part (*i.e.* $\psi(Y) = 0$), the related element of the latent variable $\mathbf{V}[i] = \mathbf{Z}[i] - \hat{\psi}_{\Theta}(Y)[i]$ follows $\mathcal{N}(0,1)$ which induces that $\hat{\psi}_{\Theta}[i]$ is negligible. Consequently, if $i \notin \{l_0,\ldots,l_{s-1}\}$, the related sample of the latent variable follows the same distribution as $\mathbf{Z}[i]$. In such scenario, the KL-divergence loss is negligible while the latent representation does not contain any information about the secret key k^*. Thus, considering these non-informative time samples do not affect the regularization term and are unsuitable for the decision process. This result encourages the adversary to only consider the time samples with a non-negligible deterministic part.

On the other hand, if $i \in \{l_0,\ldots,l_{s-1}\}$, then $\mathbf{V}[i] = \psi(Y)[i] + \mathbf{Z}[i] - \hat{\psi}_{\Theta}(Y)[i]$ follows $\mathcal{N}(0,1)$. Thus, it suggests that $\mathbf{Z}[i] \sim \mathcal{N}(\mathbb{E}[\hat{\psi}_{\Theta}(Y)[i] - \psi(Y)[i]], \mathbb{V}[\psi(Y)[i]] + \mathbb{V}[\hat{\psi}_{\Theta}(Y)[i]] + \mathbb{V}[\mathbf{V}[i]] - 2\cdot Cov[\psi[i], \hat{\psi}_{\Theta}[i]] - 2\cdot Cov[\psi[i], \mathbf{V}[i]] + 2\cdot Cov[\hat{\psi}_{\Theta}[i], \mathbf{V}[i]])$. However, due to the KL-divergence loss function involved during the training process, we force the latent variable $(\mathbf{V}[i])_{0\leq i<D}$ to follow $\mathcal{N}_D(0,\mathbf{I}_D)$. As defined in Sec.3.2, this latent variable characterizes an estimation of the noise $\mathbf{Z}$ induced in the trace $\mathbf{T}$. Thus, during the training process of the cVAE-SA, we penalize the model to tend $\mathbb{E}[\hat{\psi}_{\Theta}(Y)[i] - \psi(Y)[i]]$ towards 0 and $\mathbb{V}[\psi(Y)[i]] + \mathbb{V}[\hat{\psi}_{\Theta}(Y)[i]] + \mathbb{V}[\mathbf{V}[i]] - 2\cdot Cov[\psi[i], \hat{\psi}_{\Theta}[i]] - 2\cdot Cov[\psi[i], \mathbf{V}[i]] + 2\cdot Cov[\hat{\psi}_{\Theta}[i], \mathbf{V}[i]]$ towards 1 such that this solution is reached if and only if $\hat{\psi}_{\Theta} = \psi$. Consequently, when KL-divergence loss is computed, the conditional variational autoencoder

aims at optimizing the trainable parameters Θ of the encoder $F_{\Theta}^{(enc)}$ such that the regularization term equals 0 if and only if Θ is optimal. This justification suggests that the latent space should be only composed by PoIs. One solution is to consider dimensionality reduction techniques. In addition, when the Gaussian noise increases, the dependence between $\mathbf{T}[i]$ and $\psi[i]$ decreases. In this configuration, differentiating the sensitive information from the noise can be difficult as $\mathbf{Z}[i]$ approximately follows $\mathcal{N}(0,1)$ regardless of the information included in the time sample i. This observation confirms the benefits of the noise to reduce the efficiency of DLSCA approach.

Once the generative model $F_{\Theta,\phi}$ is trained, the adversary has to make a decision following the approximation of $\mathsf{Pr}[\mathbf{T}|Y]$ in order to fit with the stochastic attack approach. The following section describes this strategy.

3.4 Decision rule & network complexity

Typically, the inference phase of cVAE consists in generating a new set of data based on an input and a conditional known label. In SCA context, our goal is different and tends to find the conditional unknown label Y that fits the best for a given trace $\mathbf{T}$. The following part describes a new solution to retrieve the secret key k^* from the model previously defined.

Key recovery phase. During the training phase, we defined a function $F_{\Theta,\phi}$ that approximates $\log(\mathsf{Pr}[\mathbf{T}|Y,\phi])$ through an optimization algorithm (*i.e.* gradient descent-based algorithms) such that the generated trace $\tilde{\mathbf{T}}$, defined by the output of the decoder $F_{\phi}^{(dec)}$, is close to the real one $\mathbf{T}$ captured for a given label. Based on a new set of generated traces, the adversary can find the key hypothesis that approximates the marginal likelihood. Let $\mathcal{I}_a$ be a set of N_a attack traces such that $\mathcal{I}_a = \{\mathbf{t}_0, \ldots, \mathbf{t}_{N_a-1}\}$. For each of these attack traces, the adversary can compute the related label $Y = f(X,k)$ mixing the known plaintexts $X \in \mathcal{X}$ and a key hypothesis $k \in \mathcal{K}$. Then, the *Monte Carlo* method can be performed to get an estimation of $\mathsf{Pr}[\mathbf{T}|Y,\phi]$. Hence, for a set of N_v latent samples $\{\mathbf{v}_0, \ldots, \mathbf{v}_{N_v-1}\}$, we can compute an approximation of the marginal log-likelihood as follows:

$$\begin{aligned}\log\left(\mathsf{Pr}\left[\mathbf{t}_i|y_i,\phi\right]\right) \approx & -\mathcal{D}_{\mathrm{KL}}\left(\mathsf{Pr}\left[\mathbf{V}|\mathbf{t}_i,y_i,\Theta\right]||\mathsf{Pr}\left[\mathbf{V}\right]\right) \\ & -\frac{1}{2}\log\left(2\pi\cdot\sum_{j=0}^{D-1}\left(\mathbf{t}_i[j]-\frac{1}{N_v}\sum_{h=0}^{N_v-1}\tilde{\mathbf{t}}_h[j]\right)^2\right)-\frac{1}{2},\end{aligned} \tag{9}$$

where $\tilde{\mathbf{t}}_h$ is the h^{th} generated trace constructed from $\mathsf{Pr}[\mathbf{t}_i|y_i,\mathbf{v}_h,\phi]$ (see Eq.8). When the inferred posterior $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\Theta]$ deviates from the true unknown posterior $\mathsf{Pr}[\mathbf{V}|\mathbf{T},Y,\phi]$, the number of samples N_v increases in order to obtain an accurate approximation of $\mathsf{Pr}[\mathbf{T}|Y,\phi]$. If the profiling phase has been performed successfully, then $(\mathbf{t}_i - \frac{1}{N_v}\sum_{j=0}^{N_v-1}\tilde{\mathbf{t}}_j)^2$ should be minimized when $k = k^*$. Hence, the most likely candidate is defined through the maximum likelihood rule:

$$\hat{k} = \underset{k \in \mathcal{K}}{\arg\max} \left(\sum_{i=0}^{N_a - 1} \log \left(\mathsf{Pr} \left[\mathbf{t}_i | f(x_i, k), \phi \right] \right) \right).$$

To enhance the key extraction phase, the adversary can precisely define the PoIs' indexes *via* a leakage assessment once the profiling phase is performed. Indeed, if Θ and ϕ are correctly learned, the adversary can visualize them in order to properly select the PoIs (see Sec.4.2). Thus, during the attack phase, the adversary can only compute Eq.9 on the samples that are considered as relevant by the cVAE-SA.

Theoretical network complexity bounds. Based on the previous sections, we can efficiently find an architecture for a given implementation. Consequently, some theoretical network complexity bounds can be expressed following the adversary's knowledge. Indeed, our generative neural network (*i.e.* cVAE-SA) can be easily built for a given $Y \in \mathbb{F}_2^n$, a degree d of bits' interaction and a D-dimensional trace $\mathbf{T} \in \mathbb{R}^D$. First, for estimating $(\hat{\psi}_{\Theta}[i])_{0 \leq i < D}$ (resp. $(\hat{\psi}_{\phi}[i])_{0 \leq i < D}$), the encoder (resp. decoder) needs to optimize Θ (resp. ϕ) in order to retrieve the correct leakage model. Hence, for a given $Y \in \mathbb{F}_2^n$, the number of weights that has to be optimized is $((1 + \sum_{i=0}^{d} \binom{n}{i}) \cdot D)$ in both cases (s.t. $d \leq n$). Then, for estimating $(\mathbf{V}[i])_{0 \leq i < D}$ (resp. $(\tilde{\mathbf{T}}[i])_{0 \leq i < D}$), we have to link the i^{th} sample of the trace $\mathbf{T}$ (resp. the latent variable $\mathbf{V}$) with the related $\hat{\psi}_{\Theta}[i]$ (resp. $\hat{\psi}_{\phi}[i]$). Here, we decide to follow the classical stochastic attacks in order to easily extract the related noise. Hence, no weights are needed for this operation. Finally, to approximate $\boldsymbol{\mu}_{\mathbf{V},\Theta}$ (resp. $\Sigma_{\mathbf{V},\Theta}$), we need $(D \cdot (D+1))$ (resp. $D^2 \cdot (D+1)$) neurons. For the simplified diagonal case, $\Sigma_{\mathbf{V},\Theta}$ can be reduced to $\sigma^2_{\mathbf{V},\Theta}$, thus, only $D \cdot (D+1)$ neurons are needed in this configuration. To sum up the complexity measurement, the adversary needs to construct a generative model with $(D \cdot ((D+1)^2 + 2 \cdot (1 + \sum_{i=0}^{d} \binom{n}{i})))$ weights (resp. $(2D \cdot ((D+1) + 1 + \sum_{i=0}^{d} \binom{n}{i}))$ weights if $\Sigma_{\mathbf{V},\Theta}$ is reduced to $\sigma^2_{\mathbf{V},\Theta}$). Following those metrics, it can be noticed that the trace dimension D influences the most the network complexity.

However, a solution can be considered to improve the network complexity without altering the cVAE-SA performance. Indeed, following Sec.3.3, if the adversary detects s PoIs, he can construct a vector $\{l_0, \ldots, l_{s-1}\}$ of s indices such that l_i denotes the index related to the i^{th} point of interest. Based on this knowledge, he can build a cVAE-SA with lower complexity such that most of the relevant information, dedicated to the s PoIs, can be extracted from a trace. Instead of considering all the samples of the D-dimensional trace (s.t. $D \geq s$), he can construct a neural network with $(s \cdot ((s+1)^2 + 2 \cdot (1 + \sum_{i=0}^{d} \binom{n}{i})))$ weights (resp. $(2s \cdot ((s+1) + 1 + \sum_{i=0}^{d} \binom{n}{i}))$ weights if $\Sigma_{\mathbf{V},\Theta}$ is reduced to $\sigma^2_{\mathbf{V},\Theta}$). As a consequence, we drastically reduce the network complexity without altering the ability of the generative model to retrieve the secret key as suggested in Sec.3.3. For example, the network complexity of Fig.1 is about $1,040$ weights if all bits' interactions are considered (*i.e.* $d = 8$, $s = 2$ and $n = 8$). When black-box models (e.g. discriminative models) are considered, finding such complexity bounds is

known as an arduous task as no correlations are provided with classical profiled SCA.

One of the main benefits of the proposed variational autoencoder is its explainability and its interpretability regarding the side-channel context. In addition, our theoretical results suggest that its width does not have to be large no matter the dimension of the traces. This result is faithful with the Universal Approximation Theorem [15]. Through the following section, we validate these properties and broaden the attacks' spectrum on protected implementations considering the boolean making scheme.

4 Empirical investigations on cVAE-SA

Through this section, we validate all the theoretical observations provided in Sec.3 without optimizing the hyperparameters selection.

4.1 Settings

Hyperparameter selection. While classical DLSCA models need to tune a lot of hyperparameters (e.g. type of neural network, number of layers, number of nodes per layer, activation function, optimizer algorithms, learning rate, number of epochs, batch size), the configuration of the proposed cVAE-SA only deals with the optimizer algorithm, the batch size, the learning rate and the number of epochs. In this section, optimization is done using the *Adam* optimizer on batch size $\{8, 16, 32, 64, 128\}$ and the learning rate is set to $\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. We construct each model with the following number of epochs $\{10, 20, 30, 40\}$ and select the hyperparameters that provide the best ranking value. Finally, in this section, Nt_{rank} denotes the number of attack traces that are needed to reach a constant rank of 1. These traces are randomly shuffled and picked up from a set of attack traces $\mathcal{I}_a$ which is characterized by simulations that are described in the following. For a good estimation of Nt_{rank}, an average over 10 simulations, denoted $\bar{Nt}_{\text{rank}}$, is computed.

Simulations. To verify the benefits of the cVAE-SA, we simulate D-dimensional traces from a 8-bit sensitive variable Y. In this section, the simulated traces are built following two scenarios:

– **Scenario 1** – We assume the leakage model induces the maximum amount of interactions between bits (*i.e.* $\mathcal{F}_9$), such that all bits influencing the leakage model have the same weighting. Hence, the i^{th} time sample of the simulated

trace $\mathbf{T}$ is defined as follows:

$$\mathbf{T}[i] = \begin{cases} 1 \cdot Y[1] + 1 \cdot Y[3] + 1 \cdot Y[6] \\ + 1 \cdot \oplus_{b=0}^{1} Y[b] + 1 \cdot \oplus_{b=0}^{2} Y[b] + 1 \cdot \oplus_{b=0}^{3} Y[b] \\ + 1 \cdot \oplus_{b=0}^{4} Y[b] + 1 \cdot \oplus_{b=0}^{5} Y[b] + 1 \cdot \oplus_{b=0}^{6} Y[b] \\ + 1 \cdot \oplus_{b=0}^{7} Y[b] + \mathbf{Z}[i] & \text{if } i \in \{l_0, \ldots, l_{s-1}\}, \\ \\ \mathbf{Z}[i] & \text{otherwise,} \end{cases} \tag{10}$$

where $\{l_0, \ldots, l_{s-1}\}$ defines a set of indices related to each PoI, $\oplus_{b=0}^{n} Y[b] = Y[0] \oplus \ldots \oplus Y[n]$, $Y[b] = Sbox[X \oplus k^*][b]$ denotes the b^{th} bit of the output of the Sbox, and $\mathbf{Z}[i]$ is a Gaussian noise following $\mathcal{N}(0, \sigma^2)$ such that $\sigma^2 = 1$.

– **Scenario 2** – We assume that the leakage model induces interactions of degree 2 between bits (*i.e.* $\mathcal{F}_3$) but differs by the location of the PoIs. The i^{th} time sample of the trace $\mathbf{T}$ is defined as follows:

$$\mathbf{T}[i] = \begin{cases} 1 \cdot (X \oplus k^*)[5] + 1 \cdot (X \oplus k^*)[3] & \\ \oplus\, (X \oplus k^*)[7] + \mathbf{Z}[i] & \text{if } i \in \{l_0^{'}, \ldots, l_{s'-1}^{'}\}, \\ 1 \cdot Sbox\left[X \oplus k^*\right][3] + 1 \cdot Sbox\left[X \oplus k^*\right][6] + \mathbf{Z}[i] & \text{if } i \in \{l_0, \ldots, l_{s-1}\}, \\ \mathbf{Z}[i] & \text{otherwise,} \end{cases} \tag{11}$$

where $Sbox\left[X \oplus k^*\right][b]$ denotes the b^{th} bit of the output of the Sbox considering a plaintext X and the secret key k^*, $\mathbf{Z}[i]$ is a Gaussian noise following $\mathcal{N}(0, \sigma^2)$ such that $\sigma^2 = 1$.

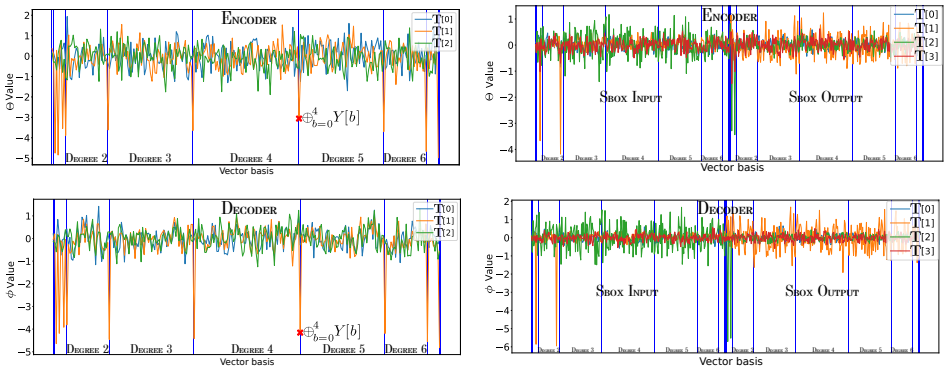
A set of $10,000$ traces ($9,000$ for the profiling phase and $1,000$ for the validation phase) is simulated for each scenario.

Limitations of the monomial basis. In [18], Schindler *et al.* suggest a non-orthonormal monomial basis to approximate the leakage model ψ. This proposition limits the adversary's interpretation when combination of bits are induced in the leakage model. To ease its interpretation, Guilley *et al.* propose a decomposition of the monomial basis to isolate the leakage from the combination of bits [6]. Through the application of the well-known *Gram-Schmidt orthonormalization* on the monomial basis, Guilley *et al.* introduce a new orthonormal monomial basis that uncorrelates each basis vector and preserve the degree of bits' interaction. Hence, constructing the cVAE-SA on this orthonormal monomial basis is beneficial to evaluate the ability of the neural network to retrieve the leakage model and maintain its interpretability. This approach will be considered in the rest of the paper. Using the orthonormal monomial basis has a major benefit. When the basis is able to describe the switching activity of the circuit, the estimated basis coefficients highlight specific exploitable security flaws in the studied implementation. Hence, visualizing the basis coefficients that characterize the cVAE-SA model $F_{\Theta,\phi}$, namely Θ and ϕ, is useful to get deeper information on the exploitable security flaws. The next section proposes

to visualize the trainable parameters Θ and ϕ in order to assess the suitability of the cVAE-SA to extract the expected leakage model ψ.

4.2 Leakage model estimation & multi-task learning

Single-sensitive variable attacks. As mentioned in Sec.3.2, the encoder (resp. decoder) is trained to retrieve the trainable parameters Θ (resp. ϕ) in order to maximize their correlation with the targeted leakage model. Once the cVAE-SA is correctly trained, the adversary can visualize these trainable parameters (*i.e.* Θ and ϕ) in order to find the security flaws induced in the studied implementation. In the considered scenario (see Fig.2a), the weight visualization can be used to assess the ability of the encoder (resp. decoder) to retrieve the leakage function defined in Eq.10. Indeed, these figures illustrate the coefficients associated to each vector of the orthonormal monomial basis. The first coefficients of each figure define the lowest bits' interaction induced in the leakage model. For example, the first element, included in the interaction of degree 5 area, is characterized by $\oplus_{b=0}^{4} Y[b]$. While the related weight is non-negligible, the cVAE-SA identifies that interaction $\oplus_{b=0}^{4} Y[b]$ influences the leakage model. This observation can be confirmed in Eq.10. Proceeding this analysis for the entire set of non-negligible weights can be helpful to evaluate the ability of the cVAE-SA to retrieve the leakage model. Indeed, if we compare the real simulated leakage model defined in Eq.10 with the non-negligible weights depicted in Fig.2a, we can see that all the peaks are associated with the correct basis vector. In addition, each



(a) Single-sensitive variable attacks (see Eq.10).

(b) Multi-task learning attacks (see Eq.11).

Fig. 2: Weight visualization assessing the suitability of our generative model to retrieve the leakage model.

coefficient associated with the sensitive interactions seems to get approximately the same impact which corresponds to the real leakage function defined in Eq.10. Consequently, if the cVAE-SA is correctly trained, it sounds helpful to retrieve complex leakage models and the related security flaws. Moreover, through the

visualization provided in Fig.2a, the adversary can also identify the time samples where the sensitive information leaks. Indeed, this figure highlights that only $\mathbf{T}[1]$ is useful to extract the leakage model. Hence, once the cVAE-SA is correctly trained, the adversary can easily retrieve the PoIs. Then, during the attack phase, the adversary can decide to focus its attack by computing Eq.9 only on $\mathbf{T}[1]$ instead of the entire trace dimension as mentioned in Sec.3.4. Contrary to classical profiled SCA approach, the cVAE-SA provides a more flexible solution during the exploitation phase by reducing the impact of irrelevant samples.

One advantage of the stochastic model is to approximate the data that depends on the secret key. Through this process, the adversary directly obtains a score related to the key manipulated by the crypto-system. Hence, the adversary can adapt the orthonormal monomial basis to target simultaneously multiple cryptographic primitives (e.g. input and output of the Sbox). Therefore, the cVAE-SA can be adapted to perform multi-tasking attacks.

Multi-task learning attacks. To exploit all the bits' interaction for each sensitive variable, we set $d = 8$ such that Fig.2b illustrates the impact of each vector of the basis $\mathcal{F}_9$. When the time sample $\mathbf{T}[1]$ is considered, we can see an interaction of degree 1 and 2 that corresponds to the bit 5 and the interaction between the bits 3 and 7 of the input of the Sbox. Then, through Fig.2b, we can see that $\mathbf{T}[2]$ extracts a leakage model with two interactions of degree 1. When we refer to Eq.11, we observe that the true leakage model depends on 3^{rd} and the 6^{th} bit of the Sbox's output. Hence, through this simulation, we can validate the ability of the cVAE-SA to correctly retrieve the leakage model of multiple sensitive variables simultaneously. This approach is highly beneficial in SCA context as it gathers more information about the targeted secret key and results in a better performance. However, as mentioned in Sec.3.4, the degree d of the orthonormal monomial basis $\mathcal{F}_{d+1}$ directly affects the complexity of the cVAE-SA. Hence, considering the attacks of multi-sensitive variable increases by $2D \cdot (1 + \sum_{i=0}^{d} \binom{n}{i})$ the number of trainable parameters (*i.e.* weights) for each new targeted sensitive variable[c]. Thus, depending on his computational capacity, the adversary has to define the most suitable structure to employ for defeating the targeted crypto-system.

However, in SCA context, the adversary mainly deals with a non-negligible number of uninformative samples. The following section assesses the ability of the cVAE-SA to mitigate this constraint.

4.3 Curse of dimensionality

When the adversary performs a side-channel attack, he wants to precisely find the relevant key-dependent time samples even if a large part of the trace contains uninformative time samples. Usually, the number of PoIs s is far lower than the trace dimension D (*i.e.* $s \ll D$). Thus, to assess the benefits of the cVAE-SA,

[c] This number can be reduced to $2s^{'} \cdot (1 + \sum_{i=0}^{d} \binom{n}{i})$ if the adversary only targets the PoIs, denoted $\{l_0^{'}, \ldots, l_{s^{'}-1}^{'}\}$.

we have to understand the ability of this new model to retrieve the PoIs when a lot of samples are irrelevant. In order to evaluate it under this restriction, we consider **Scenario 1** (see Sec.4.1) such that we construct 6 sub-scenarios where $D \in \{3, 10, 50, 100, 500, 1000\}$ and $s = 1$ such that the related *Signal-to-Noise Ratio* (SNR) equals to 0.549. Hence, for each case study, only a single PoI is configured while the dimension of the simulated traces increases. In Tab.1, we denote $P_{RI} = \frac{s}{D}$, the fraction of relevant information in each sub-scenario and evaluate the impact of this variable on other parameters, namely the batch-size and the learning rate. Finally, N_v denotes the number of samples $\mathbf{V}$ used to perform Eq.9. As suggested in Sec.3.4, the attack process is performed only

Table 1: Impact of the trace dimension on the cVAE-SA performance (with $s = 1$, $N_v = 10$, batch-size $= 10$).

P_{RI}%	D	Learning rate	$\bar{N}t_{\text{rank}}$	Network complexity	Training time
33%	3	10^{-2}	47	$1,566$	$7s$
10%	10	10^{-2}	51	$5,360$	$8s$
2%	50	10^{-2}	49	$30,800$	$35s$
1%	100	10^{-2}	46	$71,600$	$47s$
0.2%	500	10^{-3}	52	$758,000$	$401s$
0.1%	1000	10^{-3}	65	$2,516,000$	$933s$

on the time samples defined as relevant[d] by the cVAE-SA. Hence, the weight visualization applied on ϕ and Θ is very helpful to define which samples can be considered as PoIs. Through Tab.1, we can see that increasing D does not impact the resulted performance of the cVAE-SA (*i.e.* $\bar{N}t_{\text{rank}}$). Indeed, if the adversary adequately finds the correct hyperparameters, namely batch-size and learning rate, he can expect to get similar results for high values of D. However, as detailed in Sec.3.4, increasing the input dimension highly impacts the complexity of the cVAE-SA. Finding a way to focus the interest of the model only on the relevant time samples can drastically reduce the network complexity without altering its resulted performance. Such investigation could be part of a future work to highlight even more the benefits of DL in SCA context.

Once the adversary validates the ability of the generative model to deal with a low percentage of relevant information, he can question the benefits of the cVAE-SA to defeat boolean masking implementations. The next section deeply investigates this protection against this new model.

4.4 Generalization on boolean masking implementation

Typically, the discriminative models are built to automatically extract the relevant information from a trace without providing a clear interpretability of its decision making. In opposition, the cVAE-SA models the leakage model without

[d] Here, the relevance of a time sample is characterized by its coefficients ϕ and Θ such that the most relevant time samples have the highest coefficient values.

altering its representation in order to get a global characterization of $\mathsf{Pr}[\mathbf{T}|Y]$. Consequently, our generative approach does not automatically recombine the PoIs but preserve the network's explainability that is mandatory for an adversary. Thus, if we have to face with a masking implementation, some preprocessing phases are needed. In particular, a masking scheme of order o consists in a $o+1$-sharing of the sensitive target variable. In our setting, this corresponds to a situation where the leakages of the shares are hidden among values that have no relation with the target. Consequently, to perform a successful high-order attack [16], the adversary has to find the $o+1$ shares of the targeted sensitive information. Typically, classical approaches considered in profiled SCA uses some *recombination techniques* as preprocessing. This approach involves the combination of $o+1$ shares in order to "*demask*" the masked values and perform the attacks on the unmasked value. To apply this proposition, various recombination techniques [16] are introduced, namely *product combining*, *absolute difference combining* and *optimal product combining*. If the adversary wants to apply one of these techniques, he has to recombine the samples related to each of the $o+1$ shares and then, target the unmasked sensitive value Y.

To evaluate the suitability of cVAE-SA in such scenarios, we decide to simulate a 5-dimensional traces with different level of masking order $o \in \{0, 1, 2, 3\}$. For each case-study, we apply the absolute difference, the product and the optimal product combining functions and list the best result we obtained in Tab.2. Through this table, we demonstrate the ability of the proposed cVAE-SA to de-

Table 2: Impact of boolean masking implementations on the conditional variational autoencoder performance (with batch-size = 64, $N_v = 10$).

Order	**Learning rate**	$\bar{\boldsymbol{N}}\boldsymbol{t}_{\text{rank}}$	**Combining function**	**Network complexity**
0	10^{-2}	9	Optimal product	$2,630$
1	10^{-2}	32	Optimal product	$14,150$
2	10^{-3}	100	Optimal product	$95,750$
3	10^{-3}	247	Absolute difference	$1,103,750$

feat a high-order boolean masking implementation. Surprisingly, the number of shares does not highly impact the hyperparameters' value[e], namely the learning rate and the batch-size, unlike the network complexity. Indeed, for a given set of D-dimensional traces, the combining methods multiplied by D the number of time samples for each mask reduction. Hence, for performing an o order attack, the adversary has to deal with traces of D^{o+1} samples. As the dimension of the traces impacts the network complexity (see Sec.3.4), the adversary has to exponentially increase his computational ability with the attack order.

Once all these simulations validate the theoretical observations provided in Sec.3, we compare the benefits of considering the new cVAE-SA with the classical profiled side-channel attacks on real unprotected and protected implementations.

[e] The readers must be aware that this observation cannot be generalized on all implementations and would benefit from further investigations.

5 Experimental results

The experiments are implemented in Python using the *Keras* library and are run on a workstation equipped with 128GB RAM and a NVIDIA GTX1080Ti with 11GB memory. In the following section, the discriminative models are based on the CNN architectures provided by [21] and then, a global benchmark is provided with other typology of discriminative models (see Tab.4). For the generative models, the configurable hyperparameters, namely the batch-size and the learning rate, are respectively set to $\{8, 16, 32, 64\}$ and $\{10^{-1}, 10^{-2}, 10^{-3}\}$. We construct each model with the following number of epochs $\{10, 20, 30, 40, 50, 75, 100\}$ and select the value that provides the best rank. As mentioned in Sec.4, we denote $\bar{Nt}_{\text{rank}}$ the average value of Nt_{rank} over 10 shuffled experiments. In the following, we always capture the maximum amount of interactions (*i.e.* $\mathcal{F}_9$). This choice was made because an adversary does not have *a priori* knowledge on the bits' interactions.

5.1 Presentation of the datasets

We used three different datasets for our experiments. All the datasets correspond to implementations of *Advanced Encryption Standard* (AES). The datasets offer a wide range of use cases: high-SNR unprotected implementation on a smart card, low-SNR unprotected implementation on a FPGA, low-SNR protected implementation with first-order masking.

- **DPA contest-v4**[f] is an AES software implementation with a first-order masking. Knowing the mask value, we can consider this implementation as unprotected and recover the secret key directly. In this experiment, we attack the first round Sbox operation. We identify each trace with the sensitive variable $Sbox[X[0] \oplus k^*] \oplus M$ where M denotes the known mask and $X[0]$ the first byte of the plaintext.

- **AES_HD**[g] is an unprotected AES-128 implemented on FPGA. The attack targets the register writing in the last round such that the label of the i^{th} trace is $Sbox^{-1}[C[j] \oplus k^*] \oplus C[j']$ where $C[j]$ and $C[j']$ are two ciphertext bytes such that $j = 12$ and $j' = 8$.

- **ASCAD-v1** is introduced in [1]. The target platform is an 8-bit AVR microcontroller (ATmega8515) where a AES-128 protected with a boolean masking scheme is implemented. The targeted sensitive variable is the first round Sbox operation such that $Y = Sbox[X[3] \oplus k^*]$.

Remark 1. While this work bridges DL with SCA, no investigation has been conducted on the desynchronization effect. Indeed, as this countermeasure needs the use of shift-invariant layers (e.g. convolutional layers), further theoretical results

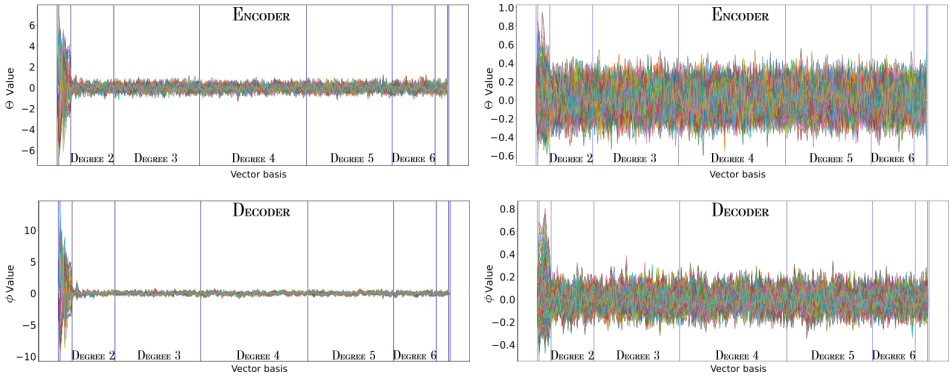
[f] `http://www.dpacontest.org/v4/42_traces.php`
[g] `https://github.com/AESHD/AES_HD_Dataset`

should be provided to clearly explain how those layers should be configured regarding the related work in SCA. This investigation is out of the scope of this paper and should be a part of a future work.

5.2 A comparison with state-of-the-art SCA

In this section, we evaluate the benefits of the cVAE-SA against the classical side-channel attacks (*i.e.* template attacks, stochastic attacks) by respecting the same experimental conditions as the state-of-the-art result. For the DPA contest-v4 and AES_HD datasets, Kim *et al.* [9] propose to select 50 features with the highest SNR in order to reduce the needs of computation when classical side-channel attacks are considered. For ASCAD-v1 dataset, we select the 8 most relevant samples related to the mask and the masked values and then, apply the three combining functions introduced in Sec.4.4.

DPA contest-v4. Once the cVAE-SA is trained, the adversary can observe the coefficients related to each time sample as illustrated in Fig.3a. Then, he can select those with the highest trainable parameters (*i.e.* Θ and ϕ) and perform his attack on this subset. For this dataset, we compute Eq.9 on the 50 time samples previously extracted. When a high-SNR unprotected implementation is considered, we observe that our generative model has the same performance as classical profiled side-channel attacks (see Tab.3). Hence, for this implementation, similar results can be obtained whatever the attack performed. Consequently, in this configuration, considering the cVAE-SA is equivalent to classical profiled side-channel attacks.



(a) DPA contest-v4 dataset. **(b)** AES_HD dataset.

Fig. 3: Weight visualization of 50 time samples assessing the suitability of our generative model to retrieve the leakage model.

AES_HD. Following the state-of-the-art results [9], a template attack needs approximately $25,000$ attack traces to retrieve the secret key. Surprisingly, performing the stochastic attack on the same dataset highly improves the related

performance. Indeed, when this approach is considered, the adversary can recover the secret key with $4,500$ attack traces which is 5.5 times better. Finally, when the cVAE-SA is applied, an even better attack can be performed. As mentioned in Sec.3.4, the attack phase is based on sample similarity measures. Hence, the adversary can only compute Eq.9 on the relevant samples detected during the profiling phase. Thus, we drastically reduce the impact of the uninformative samples during the attack phase. In this configuration, we only consider the samples where the related ϕ coefficients are greater than 0.5 (see Fig.3b). Accordingly, while the training process was performed on traces with 50 samples, the computation of Eq.9 was made on the 14 time samples complying with the configured restriction. This processing tremendously increases the performance of the resulted attack. Indeed, the cVAE-SA model divides by 100 (resp. 18) the number of attack traces that are needed to perform a template attack (resp. stochastic attack).

ASCAD-v1. As mentioned in Sec.4.4, we perform high-order attacks with the help of combining functions as preprocessing (*i.e. product combining*, *optimal product combining*, *absolute difference combining*). Then, we profile the generative models on the unmasked value in order to extract the relevant information. In Tab.3, the optimal product combination provides the best performance on the ASCAD dataset. Through this experiment, we observe that the cVAE-SA performs better than template or stochastic attacks. While 351 (resp. 290) attack traces are needed to reach a constant rank of 1 when the template attack (resp. stochastic attack) is considered, our generative model retrieves the secret key within 190 attack traces. As previously mentioned for the AES_HD dataset, this result can be explained by the ability of the cVAE-SA to target a specific range of relevant combined time samples during the attack phase. On the contrary, the classical profiled SCA have to consider the 64 time samples (*i.e.* 8 time samples related to the masks and the masked value) used to perform the related attacks. Hence, resulted noisy time samples can highly influence the performance of the resulted attacks.

Table 3: Comparison of $\bar{Nt}_{\text{rank}}$ value depending on datasets. The best performance for each dataset is denoted in blue.

	Stochastic Attacks	**Template Attacks**	**cVAE-SA** [This work]
DPA-contest v4	**4**	**4** [9]	**4**
AES_HD	$4,500$	$25,000$ [9]	**250**
ASCAD-v1	290	351	**190**

In conclusion, when a classical profiled SCA is trained on D-dimensional traces, the adversary has to perform the exploitation phase on the same trace dimension. Unfortunately, the adversary does not know *a priori* which time samples are considered as relevant once the profiling phase is applied. Hence, performing the exploitation phase on the D-dimensional traces could be impacted by the uninformative time samples. On the other hand, once the profiling phase is performed on the D-dimensional traces, the cVAE-SA is beneficial to select a

subset of s time samples, such that $s \ll D$, in order to compute Eq.9 only on the informative time samples. Hence, this new proposition is more flexible than classical profiled SCA and results in a better attack perspective as we are less impacted by uninformative time samples.

However, the adversary can question the benefits of the generative approach with respect to discriminative models. The following section highlights the benefits and the limitations of both approaches in DLSCA.

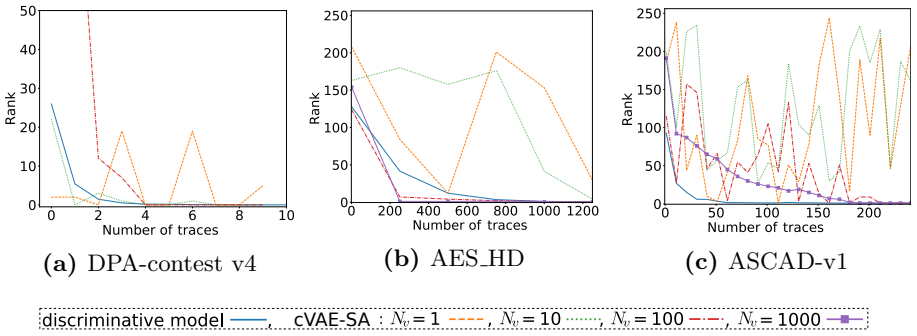
5.3 A comparison with state-of-the-art DLSCA

When the discriminative approaches are considered, a major drawback can be highlighted regarding the architecture configuration. Indeed, the resulted models have a plethora of hyperparameters to tune. The more effort we spend on the hyperparameter tuning of the network architecture, the more efficient the resulted attack is expected. In addition, due to their black-box property, the discriminative models are difficult to interpret. However, the main benefit of this approach is about automatically combining the points of interest to limit the masking effect. While the discriminative model considers all the samples of the trace, we decide to focus the interest of the generative model on the most relevant samples only. As highlighted in Sec.4.3, increasing the number of irrelevant samples highly impact the network complexity and the training time without altering the related performance. While our work does not investigate the dimensionality reduction techniques, we want to evaluate the pledge of this new proposition in the side-channel field.

DPA contest-v4. First, on Fig.4a, we can visualize the rank evolution of the generative and the discriminative models on the DPA contest-v4 dataset. While a discriminative model can retrieve the secret key with 4 attack traces, the cVAE-SA reaches the same performance depending on the number N_v of latent samples. As mentioned in Sec.3.4, the value of N_v depends on the ability of the cVAE-SA to correctly approximate $\mathsf{Pr}[\mathbf{T}|Y,\phi]$. The higher the N_v, the more confident the resulted attack. This observation can be confirmed in Fig.4a. Indeed, if $N_v = 1$, two attack traces are needed to retrieve the secret key. However, a poor rank stabilization is observed. To rectify this point, increasing the N_v value preserve a constant rank convergence towards 1.

AES_HD. The same observation can be made when the AES_HD dataset is considered. Indeed, when the N_v value increases, a rank stabilization is observed when the number of attack traces grows. In addition, Fig.4b highlights a better model when the generative approach is considered. Indeed, for $N_v = \{100, 1,000\}$, the resulted model converges towards a constant rank of 1 with 250 attack traces. Even if the discriminative approach directly estimates $\mathsf{Pr}[Y|\mathbf{T}]$, this figure indicates a lower performance when classical DLSCA model is performed. Indeed, the related number of traces that are needed for retrieving the

secret key is about $1,000$. As illustrated by Ng and Jordan [12], this result suggests that a better discriminative model can be found on this dataset. Indeed, an optimal discriminative model should be, at least, as efficient as a generative approach. However, finding the best discriminative model can be difficult due to the broad hyperparameter selection. This result highlights the benefits of the cVAE-SA in comparison with the classical DLSCA models from a practical perspective.



(a) DPA-contest v4 **(b)** AES_HD **(c)** ASCAD-v1

Fig. 4: Mean rank evolution for cVAE-SA and discriminative models.

ASCAD-v1. One benefit of the discriminative approach is to automatically recombine the points of interest. In opposition, the generative approach does not take advantage of this property (see Sec.4.4). Through Fig.4c, we can visualize the benefits of automatically combining the points of interest. Indeed, the discriminative approach converges faster towards a constant rank of 1. This result could be explained by the ability of the discriminative model to find a custom combining function that maximizes the posterior probabilities $\mathsf{Pr}[Y|\mathbf{T}]$. Hence, this custom unknown function can be more adapted for the targeted dataset. On the other hand, the cVAE-SA model is trained on combined traces that are constructed from classical approaches (*i.e.* optimal product combining). Consequently, when masking implementations are considered, a discriminative approach can provide better result than the cVAE-SA. However, applying the discriminative approach can be limited from an interpretation point of view. In addition, the discriminative approach required plethora of additional settings (*i.e.* architecture, activation function, weight initialization, etc.) that has not to be considered when the cVAE-SA is constructed. Hence, the configuration of discriminative models can be an issue from a practical perspective.

Benchmark. In Tab.4, we refer the main models introduced in the DLSCA literature. In particular, it can be mentioned that the cVAE-SA is the only model which considers the generative approach. Through this benchmark, we can assess that the cVAE-SA performs similarly, or even better, than classical DLSCA models. As suggested by Ng and Jordan [12], the asymptotic error of the discriminative model is lower or equal to the one related to the generative approach.

Therefore, discriminative models should be at least as efficient as a cVAE-SA. However, as their construction phase is not deterministic, an irrelevant model can be designed to solve a given classification task and the resulted performance can be less efficient than the cVAE-SA due to a poor approximation of the true unknown expected leakage model. This observation can be confirmed through the results provided in Tab.4.

Table 4: DLSCA benchmark of $\bar{N}t_{\text{rank}}$ value depending on datasets. The best performance for each dataset is denoted in blue.

	Discriminative model						**Generative model**
	CNN	**FCNN** (or **MLP**)	**ResNet**	**TransNet**	**SVM**	**Random Forest**	**cVAE-SA** [This work]
DPA contest-v4	**3** [21]	4 [13,14]	**3** [8]	-	**3** [14]	5 [14]	**3**
AES_HD	831 [22]	7,400 [14]	2,100 [8]	-	6,653 [14]	2,877 [14]	**250**
ASCAD-v1	191 [21]	**155** [13]	552 [8]	300 [7]	-	-	190

The results provided in this section highlight the benefits and the limitations of cVAE-SA against classical DLSCA models. In SCA, the cVAE-SA can be helpful to evaluate the feasibility of an attack and get a security bound of a device. However, while the configuration of such neural network is simple in comparison with DLSCA models, this new proposition can perform worse under masking conditions. As a perspective, we suggest to consider an hybrid approach combining the discriminative and the generative models in order to keep the explainability and the interpretability while preserving the benefits of the automatic recombination.

6 Conclusion

This paper proposes to reduce the gap between historical SCA (*i.e.* generative models) and classical DLSCA (*i.e.* discriminative models). In that purpose, we introduce the first DLSCA model based on generative approach. From the stochastic attack introduced by Schindler *et al.* [18], we first design an explainable and interpretable architecture that aims at retrieving the real unknown leakage model. Based on stochastic attack modeling, this new model can be easily constructed whatever the implementation an adversary has to deal with. Furthermore, this analogy helps us to define theoretical bounds on the network complexity (e.g. number of trainable parameters) as well as identifying mutual problematics and perspectives (e.g. dimensionality reduction, multi-task learning). Then, we theoretically explain the impact on each individual loss in SCA such that, the reconstruction loss penalizes the model in order to estimate a trace as similar as possible to the real one. On the other hand, we demonstrate that the KL-divergence loss is beneficial to correctly estimate the latent space. Compared with historical profiled SCA, the cVAE-SA is beneficial by providing the ability to carefully select the samples the adversary wants to focus on

during the exploitation phase. Hence, by providing a more flexible generative approach, we drastically reduce the impact of uninformative samples on the attack performance. This observation was confirmed on real case-study.

To bridge the gap between generative and discriminative approaches, we conduct experiments on simulations and public datasets on wide-range of use-cases and observe that the generative approach does not perform worse than a discriminative one. However, using the discriminative approach seems beneficial when masked implementations are targeted because more approriate unknown combining function can be automatically retrieved by the related model. This solution cannot be considered with generative models.

All these results suggest a lot of future works. First, as the generative approach aims at approximating a joint distribution between two random variables, we have to assess the suitability of using the cVAE-SA as model to perform non-profiled SCA, or even more generally, blind SCA. Finally, while the limitations of the discriminative (resp. generative) approach seem solved by the generative (resp. discriminative) approach, one solution could be to consider an hybrid model that preserves the automatic sample recombination property (*i.e.* discriminative approach) while keeping the explainability/interpretability and reducing the hyperparameter selection (*i.e.* generative approach).

References

1. Benadjila, R., Prouff, E., Strullu, R., Cagli, E., Dumas, C.: Deep learning for side-channel analysis and introduction to ASCAD database. Journal of Cryptographic Engineering **10**(2), 163–188 (2020). https://doi.org/10.1007/s13389-019-00220-8, `https://doi.org/10.1007/s13389-019-00220-8`
2. Bruneau, N., Guilley, S., Heuser, A., Marion, D., Rioul, O.: Less is more. In: Güneysu, T., Handschuh, H. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2015. pp. 22–41. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
3. Cagli, E., Dumas, C., Prouff, E.: Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing. In: Fischer, W., Homma, N. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10529, pp. 45–68. Springer (2017). https://doi.org/10.1007/978-3-319-66787-4_3, `https://doi.org/10.1007/978-3-319-66787-4_3`
4. Chari, S., Rao, J., Rohatgi, P.: Template attacks. In: Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems. pp. 13–28. CHES '02, Springer-Verlag, London, UK, UK (2003), `http://dl.acm.org/citation.cfm?id=648255.752740`
5. Choudary, M., Kuhn, M.: Efficient stochastic methods: Profiled attacks beyond 8 bits. In: Joye, M., Moradi, A. (eds.) Smart Card Research and Advanced Applications. pp. 85–103. Springer International Publishing, Cham (2015)
6. Guilley, S., Heuser, A., Ming, T., Rioul, O.: Stochastic side-channel leakage analysis via orthonormal decomposition. In: Farshim, P., Simion, E. (eds.) Innovative Security Solutions for Information Technology and Communications. pp. 12–27. Springer International Publishing, Cham (2017)

7. Hajra, S., Saha, S., Alam, M., Mukhopadhyay, D.: Transnet: Shift invariant transformer network for power attack. Cryptology ePrint Archive, Report 2021/827 (2021), https://eprint.iacr.org/2021/827
8. Jin, M., Zheng, M., Hu, H., Yu, N.: An enhanced convolutional neural network in side-channel attacks and its visualization. CoRR **abs/2009.08898** (2020), https://arxiv.org/abs/2009.08898
9. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(3), 148–179 (May 2019). https://doi.org/10.13154/tches.v2019.i3.148-179, https://tches.iacr.org/index.php/TCHES/article/view/8292
10. Kingma, D., Welling, M.: Auto-encoding variational bayes. In: Bengio, Y., LeCun, Y. (eds.) 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings (2014), http://arxiv.org/abs/1312.6114
11. Maghrebi, H., Portigliatti, T., Prouff, E.: Breaking cryptographic implementations using deep learning techniques. In: Carlet, C., Hasan, A., Saraswat, V. (eds.) Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings. Lecture Notes in Computer Science, vol. 10076, pp. 3–26. Springer (2016). https://doi.org/10.1007/978-3-319-49445-6_1, https://doi.org/10.1007/978-3-319-49445-6_1
12. Ng, A., Jordan, M.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems. vol. 14. MIT Press (2002), https://proceedings.neurips.cc/paper/2001/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf
13. Perin, G., Chmielewski, L., Picek, S.: Strength in numbers: Improving generalization with ensembles in machine learning-based profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems **2020**(4), 337–364 (Aug 2020). https://doi.org/10.13154/tches.v2020.i4.337-364, https://tches.iacr.org/index.php/TCHES/article/view/8686
14. Picek, S., Heuser, A., Jovic, A., Bhasin, S., Regazzoni, F.: The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(1), 209–237 (Nov 2018). https://doi.org/10.13154/tches.v2019.i1.209-237, https://tches.iacr.org/index.php/TCHES/article/view/7339
15. Pinkus, A.: Approximation theory of the mlp model in neural networks. Acta Numerica **8**, 143–195 (1999). https://doi.org/10.1017/S0962492900002919
16. Prouff, E., Rivain, M., Bevan, R.: Statistical analysis of second order differential power analysis. IEEE Transactions on Computers **58**(6), 799–811 (2009). https://doi.org/10.1109/TC.2009.15
17. Rijsdijk, J., Wu, L., Perin, G., Picek, S.: Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems **2021**(3), 677–707 (Jul 2021). https://doi.org/10.46586/tches.v2021.i3.677-707, https://tches.iacr.org/index.php/TCHES/article/view/8989
18. Schindler, W., Lemke, K., Paar, C.: A stochastic model for differential side channel cryptanalysis. In: Rao, J., Sunar, B. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2005. pp. 30–46. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)

19. Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), https://proceedings.neurips.cc/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf
20. Yu, R.: A tutorial on vaes: From bayes' rule to lossless compression (2020)
21. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Methodology for efficient cnn architectures in profiling attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems **2020**(1), 1–36 (Nov 2019). https://doi.org/10.13154/tches.v2020.i1.1-36, https://tches.iacr.org/index.php/TCHES/article/view/8391
22. Zhang, L., Xing, X., Fan, J., Wang, Z., Wang, S.: Multi-label deep learning based side channel attack. In: 2019 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). pp. 1–6 (2019). https://doi.org/10.1109/AsianHOST47458.2019.9006657