A Comprehensive Study of Deep Learning for Side-Channel Analysis

Loïc Masure[1,2], Cécile Dumas[1] and Emmanuel Prouff[2,3]

[1] Univ. Grenoble Alpes, CEA, LETI, DSYS, CESTI, F-38000 Grenoble name.surname@cea.fr
[2] Sorbonne Universités, UPMC Univ Paris 06, POLSYS, UMR 7606, LIP6, F-75005, Paris, France
[3] ANSSI, France emmanuel.prouff@ssi.gouv.fr

Abstract. In Side Channel Analysis, masking is known to be a reliable and robust counter-measure. Recently, several papers have focused on the application of the Deep Learning (DL) theory to improve the efficiency of side channel attacks against implementations protected with this approach. Even if these seminal works have demonstrated the practical interest of DL in the side-channel context, they did not argue on their theoretical soundness nor quantify their efficiency, especially with respect to the optimality bounds published so far in the literature. This paper aims at addressing this question of optimality, in particular when masking is applied. We argue that minimizing the Negative Log Likelihood during the training of Deep Learning models is actually asymptotically equivalent to maximizing a lower bound of the mutual information between the observations and the target secret chunk, or equivalently to minimizing an upper bound on underlying side-channel efficiency. Also, we argue that training a Deep Neural Networks consists in finding the parameters that maximize the Perceived Information introduced by Renauld *et al.* at EUROCRYPT 2011. These theoretical results allowed us to formally study the impact of masking counter-measures against Deep Learning based Side Channel attacks. In particular, and as expected, we verified, both on simulations and on experimental traces, that Boolean masking is sound against such a class of Side Channel attacks.

Keywords: Side Channel Analysis · Profiling Attacks · Machine Learning · Deep Learning · Masking · Mutual Information · Perceived Information · Negative Log Likelihood · Cross Entropy

1 Introduction

1.1 Context

Side-channel analysis is a class of cryptanalytic attacks that exploits weaknesses of a physical implementation of a cryptographic primitive. During its execution, the primitive processes values, called *sensitive variables*, that depend on both a piece of public data (*e.g.* a plaintext) and on some chunk of a secret value (*e.g.* a key), and of a plaintext. Knowing the plaintext and a sensitive value (or at least having some information about it) and the plaintext enables an attacker to reduce the search space of the key chunk. Implementations of secure cryptographic algorithms such as the *Advanced Encryption Standard* (AES) can then be attacked by recovering each byte of the secret key separately thanks to a *divide-and-conquer* strategy, thereby breaking the high complexity usually required to defeat such an algorithm. This information is usually gathered thanks to physical leakages such as the power consumption or the electromagnetic emanations measured on the target device. Actually, conducting an SCA is equivalent as studying the conditional probability distribution of the sensitive variables given the physical measure. It can be done for example through the computation of statistics such as a difference of means [KJJ99] or a correlation coefficient [BCO04].

For the specific type of SCA called *profiling attacks*, an attacker will try to estimate the whole conditional distribution thanks to a profiling phase during which he has access to an *open sample* for which he knows the value of the target variable. Such an access allows him to estimate the conditional distribution. Next, he will try to recover the secret key thanks to a Maximum Likelihood principle with the help of the estimated distribution. This Maximum likelihood is done on *attack traces*, that is to say traces that have been acquired on the final target device. Historically, *Gaussian Template Attacks* (GTA) have first been proposed in the early 2000's [CRR02].

Recently, the SCA community has benefited the resurgence of *Deep Neural Networks* (DNNs) [KSH12] to apply them to profiling attacks, *e.g.* in [MZ13, LBM14, GHO15, LBM15, MDM16]. Their main advantage is that they do not always require pre-processing, and are at least as efficient as the other state-of-the-art profiling attacks. The training of a DNN model is done through the minimization of a loss function computed on empirical data (*i.e.* the profiling traces in our context). The application of DNN models has been afterwards experimentally shown to be robust to some counter-measures, namely de-synchronization [CDP17] and masking [MPP16]. The first type of counter-measure consists in inducing a deformation to the leakage signal by adding for example random dummy operations or jitter. The second one consists in sharing a sensitive variable in several parts (each share being prone to noise, recovering the original data will be harder). While desynchronization transforms the signal without removing information, masking has been proved to reduce the information leakage by a factor which is exponential in the number of shares [CJRR99, PR13]. Nevertheless both counter-measures have been shown to be empirically sound against practical attacks [SVO$^+$10, VMKS12], and their use in industrial implementations is today common. In view of this popularity, it is today mandatory to measure, for each new implementations' context, their effectiveness against the most powerful class of adversaries, namely adversaries performing profiling attacks (and especially Deep Learning attacks).

When it comes to assess the security of an implementation against profiling attacks, an *evaluator* follows a Maximum Likelihood approach and tries to find the best model, *i.e.* a function that outputs scores that can be taken as probabilities, such that the secret key recovery in the attack phase is successful with a minimal number of queries $N_a^\star$ to the target device [BGH$^+$17]. This problem is called the *Evaluation Problem* (denoted in Figure 1 by the blue rectangle on the right).

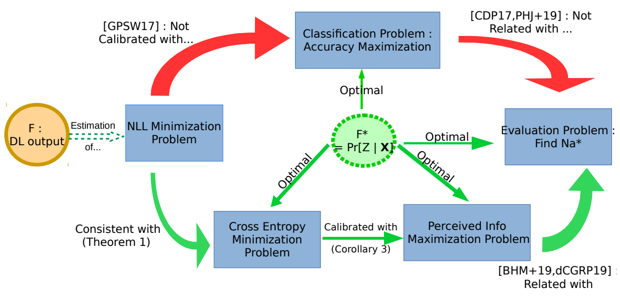
[GPSW17] : Not Calibrated with...
Classification Problem : Accuracy Maximization
[CDP17,PHJ+19] : Not Related with ...
Optimal
F : DL output
Estimation of...
NLL Minimization Problem
F* = Pr[Z | X]
Optimal
Evaluation Problem : Find Na*
Optimal
Optimal
Consistent with (Theorem 1)
Cross Entropy Minimization Problem
Calibrated with (Corollary 3)
Perceived Info Maximization Problem
[BHM+19,dCGRP19] Related with

Figure 1: Principle of Mutual Information Estimation with Deep Learning.

Denoting by Z the attack target and by $\mathbf{X}$ the noisy observation, an optimal analytical solution is to consider the true conditional *probability mass function (pmf)* $F^\star = \Pr[Z|\mathbf{X}]$ illustrated by the green disk on Figure 1. Unfortunately, such a pmf remains unknown to the evaluator. To deal with this unknown, the proposed approach in the literature of profiling attacks with Deep Learning consists in solving a *Supervised Classification* task (illustrated by the top blue rectangle on Figure 1). This task aims at finding a model that maximizes the so-called *accuracy*, namely the probability that the best score assigned to a hypothetical value of the sensitive variable corresponds to the true value of this variable. The reason for solving this problem is that Supervised Classification shares the same analytical optimal solution as in the Evaluation Problem, namely the true conditional pmf $F^\star$. However, such an analytical solution is not perfectly reachable, depending on the bias induced by the choice of a class of Machine Learning models for approximating the solution.

1.2 Problematic

Though Supervised Classification with Deep Learning has led to good practical profiling attacks, this approach has been first questioned by Cagli *et al.* who mentioned in [CDP17] that the accuracy metric may not be equivalent to the most commonly used Side Channel metrics, namely the *Guessing Entropy* (GE) and the *Success Rate* (SR). This fact has been empirically verified by Picek *et al.* in [PHJ+19]. These observations imply that the principle consisting in approaching an optimal solution to the Supervised Classification Problem may not be theoretically equivalent as approaching the optimal solution to the Evaluation Problem. This is illustrated by the top right red arrow on Figure 1.

Yet, the question of the relevance of the accuracy is ill-posed since its maximization is not directly feasible in practice. Instead, Deep Learning models are typically trained by using the *Maximum Likelihood Estimation* principle, which consists in minimizing a surrogate loss function called *Negative Log Likelihood* (NLL) as illustrated on Figure 1 on the left. This trick has been applied in many contexts such as Computer Vision [GBC16], without being proved to be theoretically sound for maximizing the Accuracy.

We think that the study, in the Side Channel Analysis context, of supervised classification approached with a Maximum Likelihood Estimation principle is an important step towards a good understanding of their efficiency and soundness. Indeed, since though an attacker only needs to know an efficient practical approach to train a DNN for attacking a device, a developer needs a theoretically grounded approach to be able to give the best security bounds on the complexity of mounting a profiling attack, especially when the implementation is protected by classical counter-measures, such as masking.

1.3 Our contribution

In this paper, we will address the question of the optimality of a Deep Learning based attack in presence of masking from the Evaluation Problem point of view.

As our first contribution, we claim that in order to assess the Evaluation Problem, one must not consider the Supervised Classification approach as theoretically grounded. This is illustrated by the red arrows in Figure 1, and will be explained in details in Subsection 2.3. In brief, not only the link between the accuracy and the optimal solution of the Evaluation Problem is questionable (as previously mentioned), but the link between the actual loss function minimized when training Deep Learning models and accuracy is also not theoretically grounded. This is illustrated in Figure 1 by the top left red arrow. As a consequence, it is not proved yet that the current training approach for Deep Learning is sound with regards to the Evaluation Problem.

On the positive side, and it will be our second contribution, we emphasize that Maximum Likelihood Estimation and the Evaluation Problem may be indeed connected to each other through another link illustrated in Figure 1 by the two bottom green arrows. Indeed it turns out that a series of papers [PR13, DFS15, BHM+19, dCGRP19] has highlighted the link between the minimal number $N_a^\star$ of queries to target device needed to succeed the attack is connected to the *Mutual Information* (MI) through the following inequality:

$$\frac{f(\beta)}{\mathsf{MI}(Z; \mathbf{X})} \leq N_a^\star, \tag{1}$$

where β is the minimal success rate required by the evaluator, f is a strictly increasing function, $N_a^\star$ is the minimal number of attack traces needed to build the optimal model, and $\mathsf{MI}(Z; \mathbf{X})$ is the mutual information between the sensitive target variable Z and the corresponding leakage $\mathbf{X}$.

Inequality 1 implies that estimating the mutual information enables to lower-bound the solution of the Evaluation Problem. We will show that actually the Maximum Likelihood Estimation principle is a sound approach to achieve such an estimation.

Indeed, we will explain that minimizing the NLL loss of a DNN model is *consistent* with minimizing the *Cross Entropy* between the true conditional pmf $F^\star = \Pr[Z|\mathbf{X}]$ and the pmf induced by the model, denoted by $F(\mathbf{X})[Z]$. This means that when the number of profiling traces converges towards infinity, both problems of minimizing the NLL loss and of minimizing the Cross Entropy are equivalent. This consistency is useful, since it turns out that the Cross Entropy is always an upper bound of the conditional entropy of the sensitive variable when knowing the leakage model, as we explain in Subsection 3.2. In the specific Side Channel Analysis context, knowing such a conditional entropy directly gives the MI. Hence, minimizing the NLL loss is asymptotically equivalent to maximizing a lower bound of the MI, or equivalently minimizing an upper bound on the left-hand side of Inequality 1. This is actually the approach an evaluator is looking for in order to assess the worst-case scenario in Side Channel Analysis. That is why Deep Learning based Side Channel Analysis makes sense in an evaluation context.

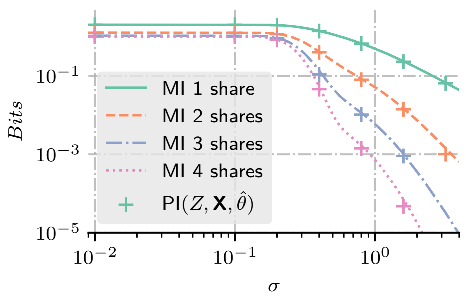

Figure 2: DNNs can optimally extract information of masked data (see fourth contribution).

Our third contribution is to identify connections between the preceding result and an information theoretic notion called *Perceived Information* (PI), introduced in [RSV+11] and recently studied by Bronchain *et al.* in [BHM+19]. In the latter paper, the authors show that the PI is a lower bound of the MI. Rephrased to our context, namely if θ denotes the vector containing all the parameters of a DNN model, and $\mathsf{PI}(Z; \mathbf{X}; \theta)$ denotes the information between a sensitive target variable Z and its leakage $\mathbf{X}$ that has been perceived by a DNN model with parameters θ, their result tells that

$$\forall \theta, \mathsf{PI}(Z; \mathbf{X}; \theta) \leq \mathsf{MI}(Z; \mathbf{X}) . \qquad \text{(Bronchain } et\ al.\text{)}$$

We will show that the information perceived by a DNN and its Cross Entropy are actually two sides of a same concept. Provided with this link, we can show that training a DNN actually consists in finding the best parameters such that the PI is maximal. More precisely, if $\hat{\theta}$ is the parameter vector obtained by minimizing the NLL loss over a profiling set of N_p traces, then:

$$\mathsf{PI}\left(Z; \mathbf{X}; \hat{\theta}\right) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_\theta \mathsf{PI}(Z; \mathbf{X}; \theta) \leq \mathsf{MI}(Z; \mathbf{X}) , \qquad \text{(This paper)}$$

where $\xrightarrow[N_p \to \infty]{\mathcal{P}}$ denotes the convergence in probabilities.

Since it is easy to compute the NLL loss on a DNN, the PI is also very straightforward to compute and can advantageously replace the accuracy as a metric to measure the quality of a training with regards to the underlying side-channel attack. As a consequence, not only the training of a DNN easily gives a lower bound on the MI, but the Maximum Likelihood Estimation will be proved to be sound to maximize the PI, in order to have the tightest possible bound for a DNN. Note that there is no reason why the latter inequality would actually be an equality. However, we will give theoretical reasons why the gap between the supremum and the MI is negligible for Multi-Layer Perceptrons (MLP), which constitutes the most simple class of DNNs.

Provided with all these theoretical results, it was now possible to formally study the impact of counter-measures against practical Deep Learning based Side Channel attacks, which is our fourth contribution. We verified, both on simulations and on experimental traces, that the Boolean masking counter-measure is sound against such a class of Side Channel attacks. We have been able to verify on simulations that the information optimally perceived by a MLP (denoted by the crosses in Figure 2) fits well the theoretical MI (denoted by the lines in Figure 2), thereby confirming the soundness of DNN attacks up to a Boolean masking order of three. The same experiment has been also conducted on experimental traces acquired on a XMEGA128D thanks to the Chip Whisperer Lite board [O'F16]. They allowed us to experimentally verify that the Perceived Information approximately follows the same trend, namely an exponential decrease, as theoretically predicted for the MI in [PR13, DFS15].

1.4 Related Work

Masking has first been shown to be sound byChari *et al.* in [CJRR99]. This result has been extended by the authors of [PR13, DFS15]. It claims that the required number of attack traces in a worst-case scenario grows exponentially with the number of shares. This has been empirically verified in [GBPV10, SVO$^+$10, GSM17, LPR$^+$14].

The first Deep Learning based attacks on masked implementations have been conducted by [GHO15, LBM15, MDM16]. However, they relied either on low-entropy masking schemes or on the assumption that one has access to the values of each share of the masking. The very first paper to present a DL based attack defeating masking while relaxing the previous assumptions has been proposed by [MPP16]. In order to answer the problem of breaking protected implementations with DL, the authors of [PSB$^+$18] have released the public database named ASCAD. They also present a CNN model that has been able to defeat the first order Boolean masking scheme contained in the database. This has also been noticed in [KPH$^+$18, Tim19]. The author of [Tim19] has even succeeded in targeting a sensitive variable protected by two masks, in a non-profiled scenario. However, he only needs to target one bit of the sensitive variable, namely the *Most Significant Bit*, to distinguish the right key hypothesis from the wrong ones. Our paper extends this study to masking orders greater than 2 and builds theoretical foundations to explain the experimental results.

In parallel, the previously mentioned papers have evaluated the efficiency of their attacks with classical Side Channel metrics that have been introduced in [SMY09]: the *Guessing Entropy* (GE) and the *Success Rate* (SR). Such metrics have sometimes been compared to the Accuracy classically involved to quantify the quality of a training. Surprisingly, none of the papers using the NLL loss as the function to minimize reported the obtained values with this minimization, though the vast majority of the works on Deep Learning for SCA used this loss to train their models.

Picek *et al.* studied in [PHJ$^+$19] the relevance of several Machine Learning metrics, without finding any good candidate strongly related to the Side Channel metrics. Nevertheless, they have not investigated the NLL loss, as the classifiers they considered were not set to output scores that could have been interpreted as probabilities. Our work will show that directly monitoring the NLL loss is relevant in the context of Side Channel Analysis, since it can be related to the MI (as claimed in Subsection 1.3).

So far, the estimation of MI has been conducted with parametric and non-parametric methods [PR10, BGP$^+$11]. To the best of our knowledge, none of these techniques have been shown to be efficient: parametric estimations suffered from the bias induced by the choice of parametric models, while non-parametric techniques suffer from the fact that the leakage has a continuous nature, though being discretized. We will explain why our approach for estimating the MI can give fairly good results.

1.5 Organization of the paper

The paper is organized as follows. In Section 2, we introduce the profiling attack scenario. We discuss the relevance of the Supervised Classification Problem in a Side Channel context, and propose another way to tackle the evaluation. Section 3 discusses the soundness of minimizing the NLL loss since is consists in maximizing the Perceived Information which is always a lower bound of the MI. A discussion about the tightness of such a lower bound can be found in Appendix C. This will then be verified by simulations in Section 4, and also illustrated on an experimental example in Section 5. .

2 Preliminaries

2.1 Notations

Throughout the paper we use calligraphic letters as $\mathcal{X}$ to denote sets, the corresponding upper-case letter X to denote random variables (resp. random vectors $\mathbf{X}$) over $\mathcal{X}$, and the corresponding lower-case letter x (resp. $\mathbf{x}$) to denote realizations of X (resp. $\mathbf{X}$). The i-th entry of a vector $\mathbf{x}$ is denoted by $\mathbf{x}[i]$. We denote the probability space of a set $\mathcal{X}$ by $\mathcal{P}(\mathcal{X})$. If $\mathcal{X}$ is discrete, it corresponds to the set of vectors $[0,1]^{|\mathcal{X}|}$ such that the coordinates sum to 1. If a random variable X is drawn from a distribution $\mathcal{D}$, then $\mathcal{D}^N$ denotes the joint distribution over the sequence of N i.i.d. random variables of same probability distribution than X. The symbol $\mathbb{E}$ denotes the expected value, and might be subscripted by a random variable $\mathbb{E}_X$, or by a probability distribution $\underset{X\sim\mathcal{D}}{\mathbb{E}}$ to specify under which probability distribution it is computed. Likewise, $\mathbb{V}$ denotes the variance of a random variable. The output of a cryptographic primitive $\mathbf{C}$ is considered as the target sensitive variable $Z = \mathbf{C}(P, K)$, where P denotes some public variable, *e.g.* a plaintext chunk, where K denotes the part of secret key the attacker aims to retrieve, and where Z takes values in $\mathcal{Z} = \{s_1, \dots, s_{|\mathcal{Z}|}\}$. Among all the possible values K may take, $k^\star$ will denote the right key hypothesis. Side-channel traces will be viewed as discrete realizations of a random column vector $\mathbf{X}$ with values in $\mathcal{X} = [0, 2^\omega - 1]^D$ where ω depends on the vertical resolution of the oscilloscope used for the acquisitions (usually, we have $\omega \in \{8, 10, 12\}$.

Let $(A_n)_n$ be a sequence of random variables and let A be another random variable. We say that A_n converges in probabilities towards A, denoted as $A_n \underset{n\to\infty}{\overset{\mathcal{P}}{\longrightarrow}} A$ when the following property holds:

$$\forall\epsilon > 0, \Pr\left[|A_n - A| \geq \epsilon\right] \underset{n\to\infty}{\longrightarrow} 0.$$

We now define some Information Theoretic quantities that have been taken from [CT06]. Let $Z \in \mathcal{Z}$ be a discrete random variable. The *entropy* of Z, denoted by $\mathsf{H}(Z)$, describes the uncertainty to guess the value of a realization of a discrete random variable Z. It is formally defined by:

$$\mathsf{H}(Z) \triangleq -\sum_{s\in\mathcal{Z}} \Pr[Z = s] \log_2 \Pr[Z = s].$$

Likewise, the *conditional entropy* of a discrete random variable Z given another random variable $\mathbf{X}$ quantifies the remaining uncertainty on the guess of Z once $\mathbf{X}$ is known. It is formally defined as:

$$\mathsf{H}(Z|\mathbf{X}) \triangleq \mathop{\mathbb{E}}_{\mathbf{X}}\left[-\sum_{s\in\mathcal{Z}} \Pr[Z=s|\mathbf{X}]\log_2 \Pr[Z=s|\mathbf{X}]\right].$$

If $\mathcal{D}$ and $\mathcal{D}'$ are two probability distributions on $\mathcal{Z}$, we define the *Kullback - Leibler divergence* (or KL divergence) as:

$$\mathsf{D}(\mathcal{D} \parallel \mathcal{D}') \triangleq \sum_{s\in\mathcal{Z}} \mathcal{D}(s)\log_2 \frac{\mathcal{D}(s)}{\mathcal{D}'(s)}.$$

This quantity is typically used to measure the difference between two discrete probability distributions, since it is always non-negative and equals zero if and only if $\mathcal{D} = \mathcal{D}'$. Thanks to the previous definitions, we can introduce the ***Mutual Information*** (MI) between two variables Z and $\mathbf{X}$ as:

$$\mathsf{MI}\left(Z;\mathbf{X}\right) \triangleq \mathsf{H}(Z) - \mathsf{H}(Z|\mathbf{X}) = \mathsf{D}(\Pr[\mathbf{X},Z] \parallel \Pr[\mathbf{X}]\Pr[Z]).$$

This characterizes how much information can be obtained about Z by observing $\mathbf{X}$.

2.2 Profiling Attacks and their Evaluation

This section presents the framework we will consider when attacking a device through a profiling attack. Once presented, we will set the goal of an evaluator.

The considered scenario is made of the following steps:

- *Profiling acquisition*: a dataset of N_p *profiling traces* is acquired on the prototype device. It will be seen as a realization of the random variable $S_p \triangleq \{(\mathbf{x}_1, z_1), \dots, (\mathbf{x}_{N_p}, z_{N_p})\} \sim \Pr[\mathbf{X}, Z]^{N_p}$, where all the $\mathbf{x}_i$ (resp. all the z_i) are i.i.d. realizations of $\mathbf{X}$ (resp. Z).

- *Profiling phase*: based on S_p, a model is built that returns a set of scores for each hypothetical value of Z, that can be assimilated to a pdf (possibly after normalization). $F : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Z})$.

- *Attack acquisition*: a dataset of N_a *attack traces* is acquired on the target device. It will be seen as a realization of $S_a \triangleq (k^\star, \{(\mathbf{x}_1, p_1), \dots, (\mathbf{x}_{N_a}, p_{N_a})\})$ such that $k^\star \in \mathcal{K}$, and for all $i \in [\![1, N_a]\!]$, $p_i \sim \Pr[P]$ and $\mathbf{x}_i \sim \Pr[\mathbf{X}|Z = \mathbf{C}(k^\star, p_i)]$.

- *Predictions*: a prediction vector is computed on each attack trace, based on the previously built model: $\mathbf{y}_i = F(\mathbf{x}_i), i \in [\![1, N_a]\!]$. For each trace, it assigns a score to each key hypothesis, namely, for every $j \in [\![1, |\mathcal{Z}|]\!]$, the value of the j-th coordinate of $\mathbf{y}_i$ corresponds to the score assigned by the model to the hypothesis $Z = s_j$ when observing $\mathbf{x}_i$.

- *Guessing*: the scores are combined over all the attack traces to output a *likelihood* for each key hypothesis; the candidate with the highest likelihood is predicted to be the right key. A Maximum Likelihood score can be used for the guessing. For every key hypothesis $k \in \mathcal{K}$, this score is defined as:

$$\mathbf{d}_{S_a}[k] \triangleq \sum_{i=1}^{N_a} \log\left(\mathbf{y}_i[z_i]\right) \text{ where } z_i = \mathbf{C}(p_i, k). \qquad (2)$$

Based on the scores in Equation 2, the key hypotheses are ranked in a decreasing order. Finally, the attacker chooses the key that is ranked first. More generally, the *rank* $g_{S_a}(k^\star)$ of the correct key hypothesis $k^\star$ is defined as:

$$g_{S_a}(k^\star) \triangleq \sum_{k \in \mathcal{K}} 1_{\mathbf{d}_{S_a}[k] > \mathbf{d}_{S_a}[k^\star]} . \tag{3}$$

If $g_{S_a}(k^\star) = 1$, then the attack is considered as *successful.*

To assess the difficulty of attacking a target device with profiling attacks (which is assumed to be the worst-case scenario for the attacked device), it has initially been suggested to measure or estimate the minimum number of traces required to get a successful attack [Man04]. Observing that many random factors may be involved during the attack, the latter measure has been refined to study the average ranking of the correct guess as a function of the number of attacked traces [SMY09]. This corresponds to the so-called ***G**uessing **E**ntropy* (GE) [SMY09] defined as:

$$\mathrm{GE}(N_a) \triangleq \mathop{\mathbb{E}}_{S_a}\left[g_{S_a}(k^\star)\right] \quad . \tag{4}$$

One can equivalently study the quantiles of the rank of the right key. This metric is called the ***S**uccess **R**ate* [SMY09]:

$$\mathrm{SR}(N_a) \triangleq \Pr\left[g_{S_a}(k^\star) = 1\right] . \tag{5}$$

Within this framework, it is common to rely the evaluator's goal in the worst-case scenario to the following problem [LPB+15, HGM+11, PHJ+19]:

Problem 1 (Evaluation Problem)**.** *Given a profiling set S_p, find a model F that minimizes N_a such that $\mathrm{SR}(N_a) \geq \beta$, where β is a threshold defined by the evaluator.*

The advantage of this Problem 1 is that a trivial optimal solution is given by the conditional pdf of the leakage, as stated in the following proposition.

Proposition 1 ([HRG14])**.** *The model $F^\star$ defined by $\forall \boldsymbol{x} \in \mathcal{X}, \forall s \in \mathcal{Z}, F^\star(\boldsymbol{x}) = \Pr[Z = s|\boldsymbol{X} = \boldsymbol{x}]$ (or $F^\star = \Pr[Z|\boldsymbol{X}]$) is the optimal solution of Problem 1. We denote $N_a^\star$ the corresponding minimum number of attack traces.*

Proposition 1 tells us that the conditional pdf is the best model we can build so far for a profiling attack. Yet, such a solution is still analytical and remains unknown to the evaluator, so we need to approximate it. A natural approach consists in finding such an approximation on a restricted set of models. This restriction will enable us to consider the found approximation as a solution to a surrogate problem that will be easier to solve. This is the current approach that is usually done when applying Machine Learning for profiling attacks. The next section will explain how sound or not this approach can be for the Evaluation Problem.

2.3 Problems with Supervised Classification

Machine Learning aims at finding, in a set of functions, the element that maximizes a performance metric for a given task given a set of empirical data. In an evaluation context, this implies first to precisely define the task we want to learn and the associate performance metric. We will present in this section the approach commonly used for profiling attacks, namely supervised classification, and will discuss its soundness.

Before continuing in the reasoning, we need to introduce the notion of *parametrized hypothesis class* that is defined as follows.

Definition 1 (Parametrized hypothesis class)**.** A parametrized hypothesis class is a set $\mathcal{H}$ of parametric pdfs of the form: $\mathbf{x} \mapsto F(\mathbf{x}, \theta) \in \mathcal{P}(\mathcal{Z})$ where $\theta \in \Theta \subseteq \mathbb{R}^q$ denotes the vector gathering all the parameters. Thus, Θ is the parameters space and equivalently defines $\mathcal{H}$.[1]

The main interest of this definition is that the optimization problem over a functional space is now reduced over a subset of $\mathbb{R}^q$.

Based on this reduction, several Machine Learning based models have been proposed to solve Problem 1, with fairly good practical results as recalled in the introduction. They are inspired by the most commonly known Machine Learning task: supervised classification. In this task, the goal is to maximize the *accuracy* of the model, namely the rate of good predictions over the distribution of the data. For any function $F : \mathcal{X} \longrightarrow \mathcal{P}(\mathcal{Z})$, this accuracy is denoted by $Acc(F)$ and is defined as:

$$Acc(F) \triangleq \Pr\left[\operatorname*{argmax}_{s \in \mathcal{Z}} F(\mathbf{X})[s] = Z\right]$$

It turns out that the classical supervised classification and the Evaluation Problem are linked, thanks to the following proposition:

Proposition 2 (Bayes Error for Supervised Classification [SSBD14])**.** *Let $F^\star$ be the conditional pdf, i.e. $F^\star = \Pr[Z|\boldsymbol{X}]$. Then for any function $F : \mathcal{X} \to \mathcal{P}(\mathcal{Z})$ we have*

$$Acc(F) \leq Acc(F^\star). \tag{6}$$

In other words, no model can have a better accuracy than the one defined with the true conditional pdf. As a consequence the approach of finding the best model according to the accuracy seems sound, as the optimal solution is the same that for Problem 1. Yet, in practice, such an optimal solution is never reachable, and sub-optimal solutions are searched with supervised classification. This therefore raises the question of the soundness of such a sub optimal solution for Problem 1. This question has first been pointed out by Cagli *et al.* who mentioned in [CDP17] that the accuracy only corresponds to finding the model that maximizes SR(1), which is different from the criterion we consider in Problem 1. Likewise, Picek *et al.* empirically verified in [PHJ+19] that the accuracy of some Machine Learning models such as SVM or Random Forest they trained was not always related to the Guessing Entropy.

Even more problematic, for the most part of Machine Learning models, the accuracy is not even the metric that is directly optimized. The reason is that this is a non-continuous function, whose derivatives are zero almost everywhere, which makes the search of an optimal model NP-hard [SSBD14]. Instead, a *surrogate loss* is used with the hope that minimizing this new loss will also maximize the accuracy as a side effect. In most use cases of Deep Learning applied to SCA, the used surrogate loss is the *Negative Log Likelihood*, which is defined as follows:

Definition 2 (Negative Log Likelihood)**.** Given $S_p = \{(\mathbf{X}_1, Z_1), \ldots, (\mathbf{X}_{N_p}, Z_{N_p}) \sim \Pr[\mathbf{X}, Z]^{N_p}$, and a DNN model defined by θ from a hypothesis class $\mathcal{H}$, the Negative Log Likelihood is defined as:

$$\mathcal{L}_{S_p}(\theta) \triangleq \frac{1}{N_p} \sum_{i=1}^{N_p} -\log_2 F(\mathbf{X}_i, \theta)[Z_i]. \tag{7}$$

Furthermore, we define the *Maximum Likelihood Estimator* as the parameter vector from Θ that minimizes the NLL loss computed over the profiling set S_p: $\hat{\theta} \triangleq \operatorname{argmin}_{\theta \in \Theta} \mathcal{L}_{S_p}(\theta)$.

[1]In the following, both notations $\mathcal{H}$ and Θ will be used interchangeably according to the context.

The soundness of the use of a surrogate loss is formulated by a property called *calibration*.

Definition 3 (Calibration [BJM06])**.** Let $\mathcal{H}$ be an hypothesis class (not necessarily parametrized), and let $\mathcal{L} : \mathcal{P}(\mathcal{Z}) \to \mathbb{R}_+$ be a *loss function*. We say that the surrogate loss function $\tilde{\mathcal{L}} : \mathcal{P}(\mathcal{Z}) \to \mathbb{R}_+$ is *calibrated with* $\mathcal{L}$ *for* $\mathcal{H}$ if for any sequence $(h_n)_{n\in\mathbb{N}}, h_n$ defined in $\mathcal{H}$, we have:

$$\tilde{\mathcal{L}}(h_n) \underset{n\to\infty}{\longrightarrow} \min_{h\in\mathcal{H}} \tilde{\mathcal{L}}(h) \implies \mathcal{L}(h_n) \underset{n\to\infty}{\longrightarrow} \min_{h\in\mathcal{H}} \mathcal{L}(h).$$

In other words, minimizing the surrogate loss leads to the minimum of the target loss.

For example, a surrogate loss called *Hinge Loss* is shown to be calibrated with the accuracy for the set of linear binary classifiers such as SVM [BJM06]. Unfortunately, finding calibrated surrogate loss functions is hard in general, especially since it depends on the nature of $\mathcal{H}$. In particular, Guo *et al.* have empirically verified the following negative result[2]:

Proposition 3 ([GPSW17])**.** *The NLL loss is not calibrated with the Accuracy for modern Deep Neural Networks.*

To sum up, in all DL-based Side Channel Attacks reported so far the adversary is minimizing a surrogate loss function that does not guarantee to minimize the accuracy, which is itself poorly related to the actual metrics an evaluator wants to assess. This is illustrated in Figure 1 by the red curves. In other words, the classical Deep Learning approach used for Supervised Classification cannot explain why practical profiling attacks made with Deep Learning has been so successfull. We propose such an argumentation in the next sections.

2.4 Training for Leakage Assessment

Starting from the observation discussed in previous sections, this paper proposes to substitute Problem 1 with another problem that is presented hereafter. This substitution comes as a direct consequence of the recent literature on SCA that has stated that $N_a^\star$, namely the number of required attack traces in order to succeed an attack with the optimal model of the Evaluation Problem, is linked to the MI between the sensitive attack target and the leakage by the following inequality [dCGRP19]:

$$\frac{f(\beta)}{\mathsf{MI}(Z;\mathbf{X})} \leq N_a^\star, \tag{8}$$

where f is an invertible, strictly increasing function defined in [dCGRP19], and β is the threshold defined in Problem 1. Cherisey *et al.* claim that the lower the MI, the tighter the bound in Equation 8. Therefore, evaluating $N_a^\star$ by estimating $\mathsf{MI}(Z;\mathbf{X})$ is hence relevant for cases where the MI can be assumed to be low, in particular when counter-measures are involved. Therefore, from the point of view of conservative security evaluation, it is interesting to estimate the left-hand side in Equation 8, or equivalently to deal with the following problem.

Problem 2 (Leakage Assessment)**.** *Given a profiling set* $S_p \sim \Pr[\boldsymbol{X}, Z]^{N_p}$*, find the tightest (i.e. the highest) lower bound on* $\mathsf{MI}(Z;\boldsymbol{X})$.

The following section aims at addressing the Leakage Assessment problem. We will show that training Deep Learning models is theoretically sound with respect to the latter which implies that conducting profiled SCA with Deep Learning can be argued to be relevant within this framework.

[2] We present in Appendix A a concrete example of miscalibration we have encountered during our experiments

3 Deep Learning for MI Estimation

This section is devoted to show that a Deep Learning model trained by minimizing the NLL loss fits with Problem 2. Subsection 3.1 studies the link between the NLL loss and an information theoretical quantity called *Cross Entropy*, that we will define hereafter. Then, Subsection 3.2 recalls the link between the Cross Entropy and the Mutual Information (MI) between the sensitive variable and the leakage, which implies to emphasize that minimizing the NLL loss actually may be viewed as maximizing a lower bound of the MI. Subsection 3.3 will make a link between the Cross Entropy, and another information theoretical quantity called *Perceived Information* (PI) introduced in [RSV+11] and recently studied by Bronchain *et al.* in [BHM+19]. Finally, Subsection 3.4 discusses the gap between the MI and a PI estimated by training Deep Learning based models. Eventually, it will be concluded that the MI can be accurately estimated thanks to this approach.

3.1 The Consistency of the NLL Loss with Cross Entropy

We present in this subsection the property of consistency of the NLL loss minimization with the minimization of an information theoretic quantity called *Cross-Entropy*, which is first defined hereafter. This way, both minimization problems will be asymptotically equivalent.

Definition 4 (Cross Entropy)**.** Given a joint probability distribution of a target sensitive variable Z and its leakage $\mathbf{X}$ denoted as $\Pr[\mathbf{X}, Z]$, we define the *Cross Entropy* as the expected value of each term in Equation 7:

$$\mathcal{L}_{\Pr(\mathbf{X},Z)}(\theta) \triangleq \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[-\log_2 F(\mathbf{X},\theta)[Z]\right], \tag{9}$$

The idea behind Maximum Likelihood Estimation is to find the parameter $\hat{\theta}$ that minimizes the NLL loss defined in Equation 7 with the hope that for N_p high enough, $\hat{\theta}$ will be a good candidate for the solution of the minimization of Equation 9. Such a trick is needed, as the true joint distribution of $\mathbf{X}$ and Z is actually unknown, and so is the Cross Entropy.

According to the law of large numbers, the empirical loss converges in probabilities towards the Cross Entropy [SSBD14] for any fixed θ. It is not trivial though that $\mathcal{L}_{S_p}(\hat{\theta})$ converges in probabilities towards $\min_{\theta\in\Theta}\mathcal{L}_{\Pr(\mathbf{X},Z)}(\theta)$, as $\hat{\theta}$ is varying for each value of N_p. Actually, the Cramer-Rao bound (a well known result in Statistics) [Cra99] guarantees the latter convergence, but relies on assumptions that cannot be taken for granted, in particular the assumption that there exists $\theta \in \Theta$ such that $F(.,\theta) = \Pr[Z|\mathbf{X}]$. Hopefully, we can circumvent this problem by using a more general result:

Theorem 1 (Consistency of Maximum Likelihood Estimation)**.** *Let $N_p \in \mathbb{N}$ and let S_p be a profiling set of size N_p. Assume that $\mathcal{H}$ is a set of Neural Networks (or equivalently Θ is its parameter space). Then:*

$$\mathcal{L}_{S_p}(\hat{\theta}) \xrightarrow[N_p\to\infty]{\mathcal{P}} \min_{\theta\in\Theta}\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta), \tag{10}$$

$$\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\hat{\theta}) \xrightarrow[N_p\to\infty]{\mathcal{P}} \min_{\theta\in\Theta}\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta). \tag{11}$$

In other words, the solution given by Maximum Likelihood Estimation converges towards the best possible solution for the Cross Entropy, and the NLL loss of $\hat{\theta}$ is a good approximation of the generalization loss of $\hat{\theta}$. We say that *the NLL loss is consistent with the Cross Entropy for the set of Neural Networks $\mathcal{H}$.*

Thanks to Theorem 1 we are guaranteed that the Maximum Likelihood Estimation is consistent with the minimization of the Cross Entropy. As a consequence, any property verified by the Cross Entropy is also asymptotically verified by the NLL loss (*i.e.* when the number of profiling traces N_p converges towards infinity). Therefore we can substitute the analysis of the NLL loss with that of the Cross Entropy. It remains now to emphasize interesting properties of the Cross Entropy in the context of the Leakage Assessment Problem. A discussion about the consistency in the non-asymptotic case is conducted in Subsection C.2

3.2 The Link between Cross Entropy and Mutual Information

In this subsection, we state the link between the Cross Entropy loss and the Mutual Information. Such a link and the reduction argued in the previous section will allow us to guarantee that minimizing the NLL loss is a consistent approach for solving the Leakage Assessment Problem.

Let us now consider the Cross Entropy. The latter one can be rewritten in terms of information theoretic quantities, as stated in the following lemma.

Lemma 1. *The generalization loss can be rewritten as the sum of the conditional entropy of $Z|\boldsymbol{X}$ and the expected value of the KL divergence:*

$$\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) = \mathsf{H}(Z|\boldsymbol{X}) + \underset{\boldsymbol{X}}{\mathbb{E}}\left[\mathsf{D}(\Pr[Z|\boldsymbol{X}] \parallel F(\boldsymbol{X},\theta))\right]. \tag{12}$$

Lemma 1 gives an interesting decomposition of the Cross Entropy into two terms: the conditional entropy and one term of KL divergence. The properties of the latter divergence enable to state the following corollary.

Corollary 1. *For every $\theta \in \Theta$, we have $\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) \geq \mathsf{H}(Z|\boldsymbol{X})$. Moreover, there is equality if and only if there exists θ such that the model $F(.,\theta)$ identically equals the true conditional pmf* $\Pr[Z|\boldsymbol{X}]$.

Corollary 1 explains that no model can have a better Cross Entropy than the conditional entropy computed with the true pmf. The corresponding result is usually called the *Bayes error* and is the error (in the sense of the Cross Entropy) made by the true conditional pmf. The expected divergence term is the *intrinsic error* due to the choice of our model.

It turns out that if the target variable Z follows a known distribution over $\mathcal{Z}$, *e.g.* Z follows a uniform law, then we are able to make a link between the Cross Entropy and the MI between the target sensitive variable and the leakage, as proposed in the following corollary.

Corollary 2. *We set the same context as in Corollary 1. If in addition we assume that Z follows the uniform law over $\mathcal{Z} = \mathbb{F}_2^n$ for some $n \in \mathbb{N}$, then:*

$$n - \mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) \leq \mathsf{MI}\left(Z; \boldsymbol{X}\right), \tag{13}$$

with equality if and only if there exists θ such that $F(.,\theta)$ identically equals $\Pr[Z|\boldsymbol{X}]$.

In other words, the Cross Entropy of any Deep Learning model enables to get a lower bound of the MI. This tells nothing about the tightness of such a bound. Hopefully, based on the previous results stated in this section, we can refine the preceding corollary, which is our main result: minimizing the NLL loss is consistent with reaching the highest value that can be obtained from the left-hand side of Equation 13.

Theorem 2 (Main result)**.** *Let $N_p \in \mathbb{N}$ and let S_p be a profiling set of size N_p. Assume that $\mathcal{H}$ is a set of Neural Networks (or equivalently Θ is its parameter space). Let $\hat{\theta}$ be the*

Maximum Likelihood Estimator, i.e. the parameter from Θ that minimizes the NLL loss computed over S_p. Then:

$$n - \mathcal{L}_{S_p}(\hat{\theta}) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_{\theta \in \Theta} \left\{ n - \mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) \right\} \leq \mathsf{MI}\left(Z; \boldsymbol{X}\right), \tag{14}$$

$$n - \mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\hat{\theta}) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_{\theta \in \Theta} \left\{ n - \mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) \right\} \leq \mathsf{MI}\left(Z; \boldsymbol{X}\right). \tag{15}$$

Equation 15 tells us that the left-hand side, converges towards the tightest lower bound that can be obtained by computing the Cross Entropy of a model from $\mathcal{H}$, thereby proving the soundness of the Maximum Likelihood Estimation approach for addressing the Leakage Assessment Problem. Even better, Equation 14 tells us that the NLL loss computed over the profiling set converges towards the same supremum, so the value of the loss function might give an accurate value of the MI if N_p is *high enough*, and provided that the gap between the supremum and the MI is negligible. Therefore, it might provide a It is then of natural interest to discuss the conditions needed to get an accurate estimation of the MI based on the left-hand side of Equation 14. This will be addressed in Subsection 3.4. But before that, we would like to make a link between the Cross Entropy and another information theoretic quantity introduced in Side Channel Analysis. This will be the topic of the next section.

3.3 The Link between Cross Entropy and Perceived Information

In [BHM+19], the authors emphasize two quantities that bound the MI: the ***H**ypothetical **I**nformation*(HI), and the ***P**erceived **I**nformation* (PI). The second one is *the amount of information that can be extracted from some data thanks to an estimated model, possibly affected by estimation or assumption error.* [3] We will show in this section that the PI actually equals the Cross Entropy up to constant factors. Before proving this result, let us recall the definition of the PI.

Definition 5 (Perceived Information [BHM+19])**.** Let Θ be the parameter space of a parametrized hypothesis class $\mathcal{H}$. Let $\theta \in \Theta$. The ***P**erceived **I**nformation* (PI) by the model $F(.,\theta) \in \mathcal{H}$ between the target sensitive variable Z and the leakage $\mathbf{X}$, denoted as $\mathsf{PI}\left(Z; \mathbf{X}; \theta\right)$ is defined as:

$$\mathsf{PI}\left(Z; \mathbf{X}; \theta\right) \triangleq \mathsf{H}(Z) + \sum_{s \in \mathcal{Z}} \Pr[Z = s] \mathop{\mathbb{E}}_{\mathbf{X}|Z=s} \left[\log_2 F(\mathbf{X}, \theta)[s]\right]. \tag{16}$$

Likewise, an *Empirical Perceived Information*, denoted as $\widehat{\mathsf{PI}_{N_p}}\left(Z; \mathbf{X}; \theta\right)$ is defined from a profiling set S_p:

$$\widehat{\mathsf{PI}_{N_p}}\left(Z; \mathbf{X}; \theta\right) \triangleq \mathsf{H}(Z) + \sum_{s \in \mathcal{Z}} \Pr[Z = s] \frac{1}{N_p} \sum_{\substack{(\mathbf{x},z) \in S_p \\ z = s}} \log_2 F(\mathbf{x}, \theta)[z] \tag{17}$$

Informally, the PI is defined the same way as the MI, but by substituting the *uncertainty* of the true pmf, namely $\log_2 \Pr[Z|\mathbf{X} = \mathbf{x}]$ with the uncertainty of the approximating pmf, namely $\log_2 F(\mathbf{X}, \theta)$. This phenomenon is exactly what defines the Cross Entropy. Therefore, the PI can be recognized in the left-hand side of Equation 13 in Corollary 2.

[3] The HI is *the amount of information that would be revealed by (hypothetical) data following the model distribution.* In our context, we only aim at estimating the conditional pmf $\Pr[Z|\mathbf{X}]$, or in other words, at estimating a discriminative model. The computation of the HI requires to have access to a generative model that approximates $\Pr[\mathbf{X}|Z]$, which we do not assume in our context.

Lemma 2. *The Cross Entropy and the NLL loss are respectively linked to the Perceived Information and its empirical estimation as follows:* [4]

$$\begin{aligned} n - \mathcal{L}_{\Pr[\boldsymbol{X},Z]}(\theta) &= \mathsf{PI}\left(Z; \boldsymbol{X}; \theta\right), & (18) \\ n - \mathcal{L}_{S_p}(\theta) &= \widehat{\mathsf{PI}_{N_p}}\left(Z; \boldsymbol{X}; \theta\right). & (19) \end{aligned}$$

Therefore Lemma 2 tells us that the Cross Entropy and the Perceived Information are exactly the same concept. As a direct consequence of Corollary 2, and as already pointed out in [BHM+19, Theorem 6], we have for all $\theta \in \Theta$: [5]

$$\mathsf{PI}\left(Z; \mathbf{X}; \theta\right) \leq \mathsf{MI}\left(Z; \mathbf{X}\right). \tag{20}$$

More interestingly, Theorem 2 can also be rewritten in terms of Perceived Information, giving the following corollary.

Corollary 3 (Main result, with PI)**.** *Let $\hat{\theta}$ denote the Maximum Likelihood Estimator, namely such that $\hat{\theta} \triangleq \mathrm{argmin}_{\theta \in \Theta} \mathcal{L}_{S_p}(\theta)$. Then: (1) $\hat{\theta}$ maximizes the empirical PI and (2) the information perceived by $\hat{\theta}$ converges in probabilities towards the supremum of Perceived Information over $\mathcal{H}$.*

$$\widehat{\mathsf{PI}_{N_p}}\left(Z; \boldsymbol{X}; \hat{\theta}\right) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_{\theta \in \Theta} \mathsf{PI}\left(Z; \boldsymbol{X}; \theta\right) \leq \mathsf{MI}\left(Z; \boldsymbol{X}\right) \tag{21}$$

$$\mathsf{PI}\left(Z; \boldsymbol{X}; \hat{\theta}\right) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_{\theta \in \Theta} \mathsf{PI}\left(Z; \boldsymbol{X}; \theta\right) \leq \mathsf{MI}\left(Z; \boldsymbol{X}\right) \tag{22}$$

Hence, Corollary 3 tells us that the sequence $\mathsf{PI}\left(Z; \mathbf{X}; \hat{\theta}\right)$ maximizes the Perceived Information that can be extracted from the leakage thanks to our hypothesis class $\mathcal{H}$. This result is a bit stronger than the one given in [BHM+19, Theorem 5], though ours relies on the assumption that the model is the Maximum Likelihood Estimator, whereas Bronchain *et al.* did not assume anything about the nature of the model.

3.4 To what Extent the Obtained Bound is Tight?

So far we have argued that minimizing the NLL loss is a sound approach to tackle Problem 2: it is indeed consistent with minimizing the Cross Entropy (*cf* Equation 12), thereby consistent with maximizing the PI (*cf* Equation 16). In the particular case where the hypothesis class $\mathcal{H}$ is a set of Neural Nets, it becomes now of natural interest to study the gap between the MI and the NLL loss we are minimizing to assess the quality of the built solution.

Such a minimization is typically done with a *Stochastic Gradient Descent* (SGD) algorithm. For convex functions, SGD is shown to converges towards the minimum [SSBD14]. Unfortunately this assumption does not hold for the NLL loss applied to DNNs [GBC16]. Hopefully, the nature of the loss landscape implies that SGD still remains a good heuristic to approach the minimum [LBH15]. We denote by θ_{SGD} the solution found by applying SGD, and it becomes of natural interest to measure the suboptimality of the latter solution

[4] Actually, Equation 19 only holds if S_p is *balanced*, *i.e.* if the number of traces is the same for each class in S_p. This can be assumed without loss of generality, since Z is drawn uniformly.

[5] Comparing with the proof of Equation 20 given in Bronchain *et al.*, the underlying idea is the same: substituting $F(.,\theta)$ with the true pmf by introducing a KL divergence term that is non-negative, which implies the inequality. The only difference is that Bronchain *et al.* compute the expectation over $\mathbf{X}|Z$ first, then over Z, whereas we compute first over $Z|\mathbf{X}$ and then over $\mathbf{X}$. This prevents from introducing the generative counterpart of our model, which we do not assume to have.

compared to the solution of Maximum Likelihood Estimation. To this end, one can decompose the gap between the solution found with SGD and the MI into three parts:

$$
\begin{aligned}
\widehat{\mathsf{PI}_{N_p}}\left(Z;\mathbf{X};\theta_{SGD}\right) - \mathsf{MI}\left(Z;\mathbf{X}\right) &= \left(\widehat{\mathsf{PI}_{N_p}}\left(Z;\mathbf{X};\theta_{SGD}\right) - \widehat{\mathsf{PI}_{N_p}}\left(Z;\mathbf{X};\hat{\theta}\right)\right) && (23)\\
&+ \left(\widehat{\mathsf{PI}_{N_p}}\left(Z;\mathbf{X};\hat{\theta}\right) - \sup_{\theta\in\Theta}\mathsf{PI}\left(Z;\mathbf{X};\theta\right)\right) && (24)\\
&+ \left(\sup_{\theta\in\Theta}\mathsf{PI}\left(Z;\mathbf{X};\theta\right) - \mathsf{MI}\left(Z;\mathbf{X}\right)\right). && (25)
\end{aligned}
$$

Equation 25 corresponds to the *approximation error*: this error is due to the choice of a restricted hypotheses class $\mathcal{H}$ from which we select our model. This error is of particular interest as it gives a *computational security bound*. Corollary 3 shows that no model from $\mathcal{H}$ can give a tighter lower bound on the MI. Equation 24 corresponds to the *estimation error*. It is the error due to the fact that we do not maximize the PI (as the true pmf is unknown) but rather its empirical estimation, since we only have a finite set of profiling traces. Equation 23 corresponds to the *optimization error*. As mentioned in the beginning of this section, this is the error made by the SGD algorithm, since it is not proved to converge towards the solution of the Maximum Likelihood Estimation.

We remark that each error term refers to a restriction in the capacity of an evaluator (finite hypothesis class, finite profiling set, heuristic for MLE instead of an exact solution). That is why it is interesting to study in details those error terms separately. Some theoretical discussions are proposed in Appendix C for each error term, and the experiments conducted in Sections 4 and 5 assess each error term.

3.5 Partial Conclusions

The results we have stated so far are threefold:

First, the approach inspired by supervised classification with Deep Learning, namely the search for the model with the best accuracy by minimizing the NLL loss, cannot explain the good performances of the practical Deep Learning based Side Channel Attacks.

Second, the loss function we are usually minimizing can be interpreted as a Perceived Information that aims at being maximized through the Maximum Likelihood Estimation. This approach is sound with the Leakage Assessment Problem which is related to the Evaluation Problem thanks to Equation 8. That is why in Section 4 and Section 5, we will plot the PI, as computed with Equation 19, since it will enable to replace the accuracy in order to compare and evaluate the efficiency of a trained model.

Third, to discuss the tightness of the Inequality 22, we can decompose the gap into three terms, namely the approximation error, the estimation error and the optimization error. A discussion about each error term can be found in Appendix C, but the experiments conducted in Sections 4 and 5 study the practical impact of each term.

Eventually, the whole discussion conducted in this section, aiming at emphasizing the links between Machine Learning concepts and metrics and the ones used in SCA, can be synthesized in Table 1.

Table 1: Machine Learning metrics and their meaning in Side Channel Analysis

ML description	ML metric		SCA metric	SCA description
Bayes error	$\mathsf{H}(Z\|\mathbf{X})$	$=$	$n - \mathsf{MI}\left(Z;\mathbf{X}\right)$	Informational security bound on $Z\|\mathbf{X}$
+ Approximation error	$\inf_{\theta\in\Theta}\mathcal{L}_{\Pr(\mathbf{X},Z)}(\theta)$	$\Longleftrightarrow$	$\sup_{\theta\in\Theta}\mathsf{PI}(\mathbf{X};Z;\theta)$	Computational bound
Cross Entropy	$\mathcal{L}_{\Pr(\mathbf{X},Z)}(\theta)$	$=$	$n - \mathsf{PI}\left(Z;\mathbf{X};\theta\right)$	Perceived Information
NLL loss	$\mathcal{L}_{S_p}(\theta)$	$=$	$n - \widehat{\mathsf{PI}_{N_p}}\left(Z;\mathbf{X};\theta\right)$	Estimated PI

4 Study on simulated data

This section confronts the different propositions made so far with simulated experiments. The aim of these experiments are: (1) to show experimentally that the PI, as defined by Equation 19, is indeed a lower bound of the MI; (2) to show, in a case where we can compute the exact MI between a sensitive target variable and a leakage, that the latter lower bound is tight, so that the PI gives an accurate estimation of the MI; (3) to see to what extent the masking order has a practical impact on the training of DNNs.[6] To this end, we first present the settings of our simulation in Subsection 4.1, and we afterwards analyze them in Subsection 4.2.

4.1 Settings of the experiments

To verify the tightness of the bounds, we simulate simple D-dimensional leakages from an n-bit sensitive variable Z. The traces are defined such that for every $t \in [\![1, D]\!]$:

$$\mathbf{x}_i[t] = \begin{cases} U_i + B_i, \text{ if } t \notin \{t_1, \ldots, t_{d+1}\} \\ hw(z_{t,i}) + B_i \text{ otherwise} \end{cases} , \tag{26}$$

where $(U_i)_i, (B_i)_i$ and all $(z_{t,i})_i$ are independent, $U_i \sim \mathcal{B}(n, 0.5)$ (*i.e.* U_i is drawn from a binomial law of parameters n and 0.5), $B_i \sim \mathcal{N}(0, \sigma^2)$, where hw denotes the Hamming weight function and where $(z_{1,i}, \ldots, z_{d+1,i})$ is a $(d+1)$-sharing of z_i for the bit-wise addition law.[7] This example corresponds to a situation where the leakages of the shares are hidden among values that have no relation with the target, but have the same marginal pmf. Since the $z_{t,i}$ are drawn uniformly, $hw(z_{t,i})$ follows a binomial marginal pmf so they are indistinguishable without prior knowledge. Hence the choice of U_i. Every possible combination of the $(d+1)$-sharing has been generated and replicated a given number of times (denoted by q) before adding the noise, in order to have an *exhaustive dataset*. Therefore, it contains $q \times 2^{(d+1)n}$ simulated traces. Once the data were generated, we trained a MLP with one hidden layer made of $r = 1000$ neurons. The training loss is naturally the NLL loss.[8] The training lasts 200 epochs, with a Stochastic Gradient Descent and a learning rate of 10^{-3}.

Our simulations comprise two main campaigns:

- **Experiment 1** (without fool components): in this experiment, we set $D = d + 1$ in order to avoid to consider irrelevant input features. The simulations are done over $n = 4$ bits, $m \in \{1, 2, 3, 4\}$ and $\sigma \in \{0.01, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2\}$. We also generate enough data so that the training set is "exhaustive", *i.e.* the number of replicas is $q = 2000$. Therefore, we expect to make the estimation error negligible in Equation 24.[9] The gap between the MI and the PI should therefore only be composed of the optimization error (23) and the approximation error (25).

- **Experiment 2** (with fool components): in a second experiment, we have $D = 40$; the fool components are taken into account and their position is randomly shuffled among the relevant ones. Since all components share the same margin law, we recall that they cannot be distinguished without knowing Z. Compared to Experiment 1, we might expect the optimization error to be more important because of the potential difficulty induced by the fool components. The estimation error might also increase, but since we still have an exhaustive dataset, we expect it to remain negligible. The

[6] The two first aim at confirming previous analyses (*e.g.* done in [BHM+19]).

[7] A masking scheme of order d consists in a $d+1$-sharing of the sensitive target variable.

[8] Beware that in Pytorch and Tensorflow, the NLL loss is computed with natural logarithms, whereas one must consider the logarithm in base 2.

[9] One can assess it by computing the ratio $\frac{h}{N_p}$ defined in Equation 29.

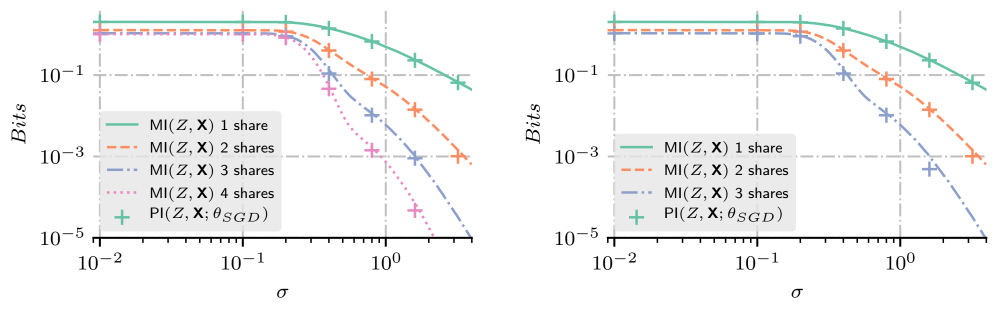

Figure 3: Information perceived by the MLP in Experiments 1 (left) and 2 (right).

goal is also to validate the results of Proposition 4, namely that non informative components have no impact on the approximation error.

From those experiments, the Perceived Information $\mathsf{PI}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ is estimated thanks to a hold-out dataset of 1/5-th of the size of the training set size. For the sake of comparison, we estimate the MI between the target sensitive 4-bit variable and its simulated leakage model with a *Monte Carlo* sampling.

4.2 Analysis of the Results

In this section we analyze the results obtained by running the Experiments 1 and 2.

Experiment 1. The results of Experiment 1 are given in Figure 3 (left). The plain lines correspond to the estimated MI and the crosses correspond to the information perceived by the trained MLP, as computed from the Cross Entropy with Equation 19. We can see that the crosses are always below the lines while staying very close. This experimentally validates (1) that we actually have a lower bound of the MI as a performance metric, and (2) that the gap between PI and MI is very small. Since we argued that the gap could only be made of the approximation and optimization error, it implies that both are negligible, relatively to the MI, in the particular case of a Hamming weight model with Gaussian noise, no matter the order of masking (at least for $d \leq 4$). This also empirically confirms that the Universal Approximation Theorem might also hold when considering the KL divergence instead of the Euclidean norm as an error metric. In other words, this shows on a simple case that the loss optimized by a Neural Net can give a very accurate estimation of the Mutual Information, thereby efficiently solving the Leakage Assessment Problem (*i.e.* Problem 2). Note that for this experiment, we have never assumed to have any prior knowledge on the leakage model during the profiling phase.[10]

Experiment 2. Likewise, the results of Experiment 2 are given in Figure 3 (right). The lines correspond to the estimated MI and the crosses correspond to the information perceived by the trained MLP. The case for $m = 4$ shares is missing because the training would have taken too much time and memory: the exhaustive dataset would have weighed 20 GB, and one epoch, *i.e.* the time made by the SGD algorithm to process all the data at least once, would have taken approximately one day. Figure 3 (right) looks almost exactly

[10] A careful reader may remark that some points are missing for $d \in \{3, 4\}$ and $\sigma = 3.2$: the trainings were not successful. Either the validation loss was still equal to 4 bits, or marginally below 4 bits. In the latter case, a confidence interval computed with Equation 31 in Appendix D was too wide to conclude that the PI was strictly more than zero. This must tell that below a given MI threshold, the learning might not be possible, or at least much harder even with an exhaustive dataset.

the same as Figure 3 (left), except for some points, such as for 3 shares and $\sigma = 1.6$. The latter point denotes a PI that is lower than the one obtained in Experiment 1. This is a clue indicating that the optimization might have been harder than without fool components. Besides this slight difference, the curves look the same. This provides clues to the veracity of Proposition 4 given in Appendix, namely that the Approximation error is exponentially increasing when the input dimensionality of the trace increases.

4.3 Discussions

Through the simulations, we have empirically verified that for a theoretical multivariate Hamming weight leakage with a Gaussian noise, a simple MLP can not only approximate well the true conditional pmf $\Pr[Z|\mathbf{X}]$. Indeed, we recall that the gap between the Cross Entropy and the conditional Entropy, or equivalently the gap between the information perceived by the model and the MI, can be written as a KL divergence between the true pmf and the underlying pmf of the model. Since we have argued that this gap is negligible, it follows that so is the KL divergence. This proves that our model is a good estimation of the true pmf in the sense of the KL divergence.

Minimizing the NLL loss also gives accurate bounds on the mutual information between the sensitive target variable and the leakage. This result is even stronger since it has been obtained without giving any prior knowledge on the leakage model to the MLP. More precisely, the simulations show that in practice, the approximation error might not be limiting in a context of Side Channel traces. The capacity of a DNN to correctly estimate the MI still needs to be verified on experimental conditions, and especially in cases where one cannot assume to have an exhaustive profiling dataset. Therefore more information will be learned about the estimation error.

5 Application on Experimental Data

So far, we have seen that Deep Neural Nets could reach the informational security bounds of a leakage in simulated experiments, thereby giving useful estimations for the developer. This success did not rely on any prior knowledge on the leakage, but was achieved thanks to a simple MLP with one hidden layer. To confirm these observations, we propose to complete the investigations by considering experimental leakage traces. Subsection 5.1 presents the acquisition of the dataset used for the experiments, Subsection 5.2 presents the methodology of our experiments, Subsection F.1 details the MLP architecture that has been used for the experiments, and Subsection 5.3 discusses their results.

5.1 Presentation of the Dataset

The leakage traces represent the power consumption of a XMEGA128D4 chip supported on a Chip Whisperer Lite board [O'F16]. The program ran on the Chip aims at simulating the leakage of several shares that may be processed by a protected implementation of a cryptographic primitive. The firmware is directly written in assembly code and consists in loading each byte of an input plaintext array to a register, setting it to zero and then storing it back to the input array. Some details of the code are given in Appendix E. $500,000$ traces of 2500 time samples each have been acquired, along with the corresponding bytes array denoted by `plain[i]`, $i \in [\![0, 15]\!]$. The complete acquisition has been done within 15 hours.

To reproduce conditions similar to the simulations, we only target the $n = 4$ most significant bits of the target variable. In other words, $|\mathcal{Z}| = 2^n = 16$. The generated target values are $Z = \bigoplus_{i \in [\![0,d]\!]} \texttt{plain}[i]$ for $d \in \{0, 1, 2\}$, where $\oplus$ denotes the `xor` operation between two bytes. This way, it can simulate leakages of order d.

Provided with these target values, we selected Points of Interest (PoIs) based on the magnitude of the *Signal-to-Noise Ratio* [MOP07]: between 4 and 6 PoIs are selected in decreasing order of magnitude of SNR from each of the three first bytes of the plaintext array. The time coordinates 13 to 16, 25 to 30 and 37 to 41 respectively correspond to the PoIs of the latter bytes manipulation. This gives an input dimension of $D = 15$. This way, we hoped to reduce the quantity of irrelevant components, which would have made the optimization with SGD harder, and therefore hoped to get a good estimate that corresponds the best to the approximation error (25).

Eventually, we verified that the traces did not contain unexpected leakages that might help targeting masked variables (see Figure 6 and the corresponding discussion in Appendix F).

5.2 Methodology

Details of the trained MLP can be found in Subsection F.1 The training has been done with a variant of the SGD algorithm called *Adam* [KB15] through a number of epochs denoted by T, *i.e.* each trace has been processed T times by the Adam algorithm. Over the $500,000$ profiling traces, a portion k is used for the training, and the remaining is used as a hold-out set for computing an unbiased estimate of the Perceived Information. In other words, the profiling set is made of $N_p = k \times 500,000$ traces while the hold-out set is made of $N_v = (1-k) \times 500,000$ traces. We fix the limit $k \leq 4/5$ so that the quality of the estimation over the hold-out set remains satisfying: the error margin will be at most 10^{-2} with a confidence at least 90% in the worst case, according to Equation 31.

Two experiments have been done:

- Experiment 3: we set $k = 1/2$, $T = 100$. The aim of this experiment is to illustrate the result of a simple training on real traces, and discuss how to interpret them in a SCA context, in order to validate whether the MI may exponentially decrease with the masking order.
- Experiment 4: we set $T = 200$ and let k vary so that $N_p \in [\![1,000; 400,000]\!]$. This way, we will be able to plot the so-called *learning curve*, namely plotting the values of $\mathsf{PI}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ and $\widehat{\mathsf{PI}_{N_p}}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ depending on N_p. This is a classical representation in Machine Learning to verify the consistency property claimed in Corollary 3, evaluate the underlying convergence rate [SSBD14] and thereby the non-asymptotic estimation error (24). It is expected that the empirical estimation of the PI decreases with N_p while the true PI increases, and both converge towards the supremum of the PI.

5.3 Results and Discussions

Figure 4 presents the trainings done when targeting every combination of $1, 2$ or 3 shares among the considered ones. The plot illustrates the information perceived by the MLP through the training epochs. It is reminded that since the PI is a lower bound of the MI, one must read the curves as the information *at least* contained in the leakage.

It may be observed that the amount of information leaking on the sensitive un-split variable seems to decrease at an exponential rate in the number of shares, as expected from both theory [PR13, DFS15] and our simulations (see Section 4).

Figure 4 (right) presents the learning curves. As expected, the PI converges towards its highest possible value obtained from the hypothesis class $\mathcal{H}$, while the estimated PI from the profiling set S_p decreases and also converges towards the same value. Note that there is no theoretical reason that the true value of the MI is bounded by both curves. However, since we have shown that the approximation error may be negligible in our context, there are still strong chances that the MI can be bounded by both curves.

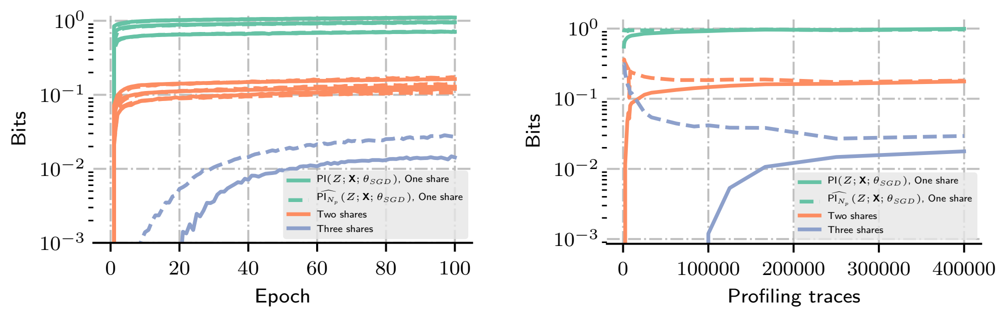

Figure 4: Left: result of Experiment 3 for each considered target variable (7 in total). Right: learning curve of Experiment 4 for three different target variables, where $d = 0, 1, 2$.

Another interesting observation is that the blue plain curve corresponding to the PI obtained when targeting a second order Boolean masking does not directly increases before $N_p = 250,000$. This tells that the profiling phase completely failed, since no information at all could be extracted. Moreover, when the profiling phase did not fail, it took much more traces for the PI of the trained model to converge towards its optimal value. So not only masking highly drops the amount of information hidden in the Side Channel Traces, but it also harden the extraction of information for Deep Learning models.

6 Conclusion

In this paper, we have given some theoretical and experimental reasons why the Deep Learning paradigm is suitable for evaluating implementations protected by a masking scheme through a Profiling Attacks scenario.

Contrary to what is commonly believed, the supervised classification approach is not theoretically grounded. Yet, Deep Learning based attacks still worked. The reason is that the underlying surrogate task DNNs are trained to is much more interesting, since it is consistent with minimizing an information theoretical quantity named Cross Entropy which is closely linked to the Mutual Information (MI) between the sensitive target variable and the leakage.

We have made some parallels with another information theoretical quantity introduced for Side Channel Analysis and recently studied: the Perceived Information (PI). It turns out that the PI studied in SCA, and the Cross Entropy studied in Deep Learning are two sides of the same coin. Therefore, we could state our main result, namely that the Deep Learning approach actually consists in maximizing the information perceived by a Deep Neural Network. Since the PI has the interesting property to be a lower bound of the MI, the latter approach is nothing but refining the most possible such a lower bound.

The tightness of this lower bound has been studied, and both theoretical results and empirical verifications have been provided through this paper. This study showed that the PI estimated from a one hidden layer Neural Net can approach with a negligible error, thereby providing a new way to accurately evaluate the MI.

This leads to the takeaway message of this paper: the Deep Learning paradigm is a sound approach for defeating a masking scheme of higher order, even maybe the optimal one. But it cannot outperform what is theoretically guaranteed by the security bounds given for masking given by the Mutual Information.

This work being a theoretical study, it can be extended in many ways: a first one could be the study of other masking schemes. Though it is not related in this paper, similar results of Experiment 1 in Section 4 have been obtained for an arithmetical masking scheme. A second way to extend this work would be to not restrict the input traces to a few PoIs, but also to study the case of high dimensionality. This would raise a challenge for controlling the optimization error since the PoIs would be flooded into lots of irrelevant features.

References

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings, volume 3156 of Lecture Notes in Computer Science, pages 16–29. Springer, 2004.

[BGH+17] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, Damien Marion, and Olivier Rioul. Optimal side-channel attacks for multivariate leakages and multiple models. J. Cryptographic Engineering, 7(4):331–341, 2017.

[BGP+11] Lejla Batina, Benedikt Gierlichs, Emmanuel Prouff, Matthieu Rivain, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Mutual information analysis: a comprehensive study. J. Cryptology, 24(2):269–291, 2011.

[BHM+19] Olivier Bronchain, Julien M. Hendrickx, Clément Massart, Alex Olshevsky, and François-Xavier Standaert. Leakage certification revisited: Bounding model errors in side-channel security evaluations. Cryptology ePrint Archive, Report 2019/132, 2019. https://eprint.iacr.org/2019/132.

[BJM06] Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. Journal of the American Statistical Association, 101(473):138–156, 2006. (Was Department of Statistics, U.C. Berkeley Technical Report number 638, 2003).

[BR14] Lejla Batina and Matthew Robshaw, editors. Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings, volume 8731 of Lecture Notes in Computer Science. Springer, 2014.

[CDP16] Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. Enhancing dimensionality reduction methods for side-channel attacks. In Naofumi Homma and Marcel Medwed, editors, Smart Card Research and Advanced Applications, Lecture Notes in Computer Science, pages 15–33. Springer International Publishing, 2016.

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

[CHM+15] Anna Choromanska, Mikael Henaff, Michaël Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In Guy Lebanon and S. V. N. Vishwanathan, editors, Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San

Diego, California, USA, May 9-12, 2015, volume 38 of JMLR Workshop and Conference Proceedings. JMLR.org, 2015.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Wiener [Wie99], pages 398–412.

[Cra99] Harald Cramér. Mathematical methods of statistics. Princeton University Press, 1999. OCLC: 185436716.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers, volume 2523 of Lecture Notes in Computer Science, pages 13–28. Springer, 2002.

[CT06] Thomas M. Cover and Joy A. Thomas. Elements of information theory (2. ed.). Wiley, 2006.

[dCGRP19] Eloi de Chérisey, Sylvain Guilley, Olivier Rioul, and Pablo Piantanida. Best information is most successful mutual information and success rate in side-channel analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst., 2019(2):49–79, 2019.

[DFS15] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Elisabeth Oswald and Marc Fischlin, editors, Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I, volume 9056 of Lecture Notes in Computer Science, pages 401–429. Springer, 2015.

[GBC16] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. Deep Learning. Adaptive computation and machine learning. MIT Press, 2016.

[GBPV10] Benedikt Gierlichs, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. Revisiting higher-order DPA attacks:. In Josef Pieprzyk, editor, Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings, volume 5985 of Lecture Notes in Computer Science, pages 221–234. Springer, 2010.

[GHO15] Richard Gilmore, Neil Hanley, and Máire O'Neill. Neural network based attack on a masked implementation of AES. In IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2015, Washington, DC, USA, 5-7 May, 2015, pages 106–111. IEEE Computer Society, 2015.

[GPSW17] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, volume 70 of Proceedings of Machine Learning Research, pages 1321–1330. PMLR, 2017.

[GSM17] Hannes Groß, David Schaffenrath, and Stefan Mangard. Higher-order side-channel protected implementations of KECCAK. In Hana Kubátová, Martin Novotný, and Amund Skavhaug, editors, Euromicro Conference on Digital System Design, DSD 2017, Vienna, Austria, August 30 - Sept. 1, 2017, pages 205–212. IEEE Computer Society, 2017.

[Har] Moritz Hardt. Generalization Theory and Deep Nets, An introduction.

[HGM+11] Gabriel Hospodar, Benedikt Gierlichs, Elke De Mulder, Ingrid Verbauwhede, and Joos Vandewalle. Machine learning in side-channel analysis: a first study. J. Cryptographic Engineering, 1(4):293–302, 2011.

[HRG14] Annelie Heuser, Olivier Rioul, and Sylvain Guilley. Good is not good enough - deriving optimal distinguishers from communication theory. In Batina and Robshaw [BR14], pages 55–74.

[HSW89] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. Neural Networks, 2(5):359–366, 1989.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, volume 37 of JMLR Workshop and Conference Proceedings, pages 448–456. JMLR.org, 2015.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Wiener [Wie99], pages 388–397.

[KPH+18] Jaehun Kim, Stjepan Picek, Annelie Heuser, Shivam Bhasin, and Alan Hanjalic. Make some noise: Unleashing the power of convolutional neural networks for profiled side-channel analysis. Cryptology ePrint Archive, Report 2018/1023, 2018. https://eprint.iacr.org/2018/1023.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States., pages 1106–1114, 2012.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[LBM14] Liran Lerman, Gianluca Bontempi, and Olivier Markowitch. Power analysis attack: an approach based on machine learning. IJACT, 3(2):97–115, 2014.

[LBM15] Liran Lerman, Gianluca Bontempi, and Olivier Markowitch. A machine learning approach against a masked AES - reaching the limit of side-channel attacks with a learning model. J. Cryptographic Engineering, 5(2):123–139, 2015.

[LPB+15] Liran Lerman, Romain Poussier, Gianluca Bontempi, Olivier Markowitch, and François-Xavier Standaert. Template attacks vs. machine learning revisited (and the curse of dimensionality in side-channel analysis). In Stefan Mangard and Axel Y. Poschmann, editors, Constructive Side-Channel

Analysis and Secure Design - 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers, volume 9064 of Lecture Notes in Computer Science, pages 20–33. Springer, 2015.

[LPR+14] Victor Lomné, Emmanuel Prouff, Matthieu Rivain, Thomas Roche, and Adrian Thillard. How to estimate the success rate of higher-order side-channel attacks. In Batina and Robshaw [BR14], pages 35–54.

[Man04] Stefan Mangard. Hardware countermeasures against DPA ? A statistical analysis of their effectiveness. In Tatsuaki Okamoto, editor, Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings, volume 2964 of Lecture Notes in Computer Science, pages 222–235. Springer, 2004.

[MBvLM12] Dimitrios Mavroeidis, Lejla Batina, Twan van Laarhoven, and Elena Marchiori. Pca, eigenvector localization and clustering for side-channel attacks on cryptographic hardware devices. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part I, volume 7523 of Lecture Notes in Computer Science, pages 253–268. Springer, 2012.

[MDM16] Zdenek Martinasek, Petr Dzurenda, and Lukas Malina. Profiling power analysis attack based on MLP in DPA contest V4.2. In 39th International Conference on Telecommunications and Signal Processing, TSP 2016, Vienna, Austria, June 27-29, 2016, pages 223–226. IEEE, 2016.

[MDP19] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. Gradient visualization for general characterization in profiling attacks. In Ilia Polian and Marc Stöttinger, editors, Constructive Side-Channel Analysis and Secure Design - 10th International Workshop, COSADE 2019, Darmstadt, Germany, April 3-5, 2019, Proceedings, volume 11421 of Lecture Notes in Computer Science, pages 145–167. Springer, 2019.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. Power analysis attacks - revealing the secrets of smart cards. Springer, 2007.

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

[MZ13] Zdenek Martinasek and Vaclav Zeman. Innovative method of the power analysis. Radioengineering, 22:586–594, 06 2013.

[O'F16] Colin O'Flynn. ChipWhisperer®, February 2016.

[Pet98] P. Petrushev. Approximation by ridge functions and neural networks. SIAM Journal on Mathematical Analysis, 30(1):155–189, 1998.

[PHJ+19] Stjepan Picek, Annelie Heuser, Alan Jovic, Shivam Bhasin, and Francesco Regazzoni. The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. IACR Trans. Cryptogr. Hardw. Embed. Syst., 2019(1):209–237, 2019.

[Pin99] Allan Pinkus. Approximation theory of the MLP model in neural networks. Acta Numerica, 8:143–195, 1999.

[PR10] Emmanuel Prouff and Matthieu Rivain. Theoretical and practical aspects of mutual information-based side channel analysis. IJACT, 2(2):121–138, 2010.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, volume 7881 of Lecture Notes in Computer Science, pages 142–159. Springer, 2013.

[PSB+18] Emmanuel Prouff, Remi Strullu, Ryad Benadjila, Eleonora Cagli, and Cecile Dumas. Study of deep learning techniques for side-channel analysis and introduction to ascad database. Cryptology ePrint Archive, Report 2018/053, 2018. https://eprint.iacr.org/2018/053.

[RSV+11] Mathieu Renauld, François-Xavier Standaert, Nicolas Veyrat-Charvillon, Dina Kamel, and Denis Flandre. A formal study of power variability issues and side-channel attacks for nanoscale devices. In Kenneth G. Paterson, editor, Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, volume 6632 of Lecture Notes in Computer Science, pages 109–128. Springer, 2011.

[SHK+14] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1):1929–1958, 2014.

[SMY09] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings, volume 5479 of Lecture Notes in Computer Science, pages 443–461. Springer, 2009.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, 2014.

[STIM18] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada., pages 2488–2498, 2018.

[SVO+10] François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In Masayuki Abe, editor, Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings, volume 6477 of Lecture Notes in Computer Science, pages 112–129. Springer, 2010.

[Tim19] Benjamin Timon. Non-profiled deep learning-based side-channel attacks with sensitivity analysis. IACR Trans. Cryptogr. Hardw. Embed. Syst., 2019(2):107–131, 2019.

[Vap99] Vladimir Vapnik. An overview of statistical learning theory. IEEE Trans. Neural Networks, 10(5):988–999, 1999.

[VMKS12] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In Xiaoyun Wang and Kazue Sako, editors, Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings, volume 7658 of Lecture Notes in Computer Science, pages 740–757. Springer, 2012.

[Wie99] Michael J. Wiener, editor. Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings, volume 1666 of Lecture Notes in Computer Science. Springer, 1999.

A Example of non Calibrated Training

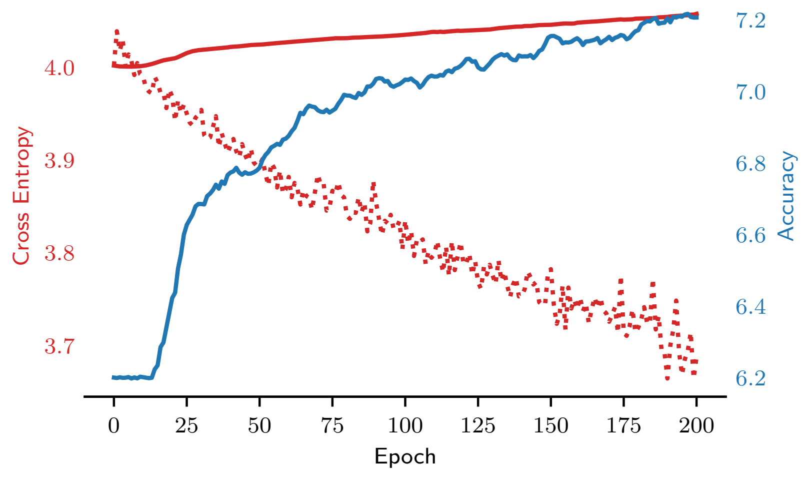

Figure 5

We present an example of training showing how the NLL loss can be miscalibrated with the Accuracy. A DNN has been trained to minimize the NLL loss of a 4 bits target sensitive variable protected by two masks. The NLL loss is in red: the dotted line denotes the loss computing over the profiling phase, *i.e.* what is minimized during the training, whereas the plain line denotes the Cross Entropy over a *hold-out* set, *i.e.* some traces that have not been used for minimizing the loss. Here, learning the sensitive target variable has been so hard that while the training loss is indeed minimized, it poorly generalizes on the hold-out set, since the NLL loss over this set slightly increases and is even greater than the number of bits. The problem is that by looking at the Accuracy on the hold-out set, we would conclude that the model performances generalize well: the final Accuracy is greater than what could have been obtained with a random prediction. Therefore depending on the choice of the performance metric, the conclusion radically change.

B Proofs of Lemmas and Theorems

Proof of Theorem 1. This is a direct consequence of the Fundamental Theorem of Statistical Learning which states that the result holds if and only if the *VC-dimension* of $\mathcal{H}$ is finite [Vap99], which is a property verified by the set of Neural Networks [SSBD14], as it can be bounded by a function of the number of neurons. □

Proof of Lemma 1.

$$\begin{aligned}
\mathcal{L}_{\Pr(\mathbf{X},Z)}(\theta) &\triangleq \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[-\log_2 F(\mathbf{X},\theta)[Z]\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[-\log_2 \Pr[Z|\mathbf{X}]\right] + \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[-\log_2 F(\mathbf{X},\theta)[Z]\right] - \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[-\log_2 \Pr[Z|\mathbf{X}]\right] \\
&= \mathsf{H}(Z|\mathbf{X}) + \mathop{\mathbb{E}}_{\mathbf{X},Z}\left[\log_2\left(\frac{\Pr[Z|\mathbf{X}]}{F(\mathbf{X},\theta)[Z]}\right)\right] \\
&= \mathsf{H}(Z|\mathbf{X}) + \mathop{\mathbb{E}}_{\mathbf{X}}\left[\mathop{\mathbb{E}}_{Z|\mathbf{X}}\left[\log_2\left(\frac{\Pr[Z|\mathbf{X}]}{F(\mathbf{X},\theta)[Z]}\right)\right]\right] \\
&= \mathsf{H}(Z|\mathbf{X}) + \mathop{\mathbb{E}}_{\mathbf{X}}\left[\mathsf{D}(\Pr[Z|\mathbf{X}] \parallel F(\mathbf{X},\theta))\right]
\end{aligned}$$

□

Proof of Corollary 1. Let $\mathbf{x} \in \mathcal{X}$ be fixed. The KL divergence is always non-negative and equals zero if and only if both discrete distributions $\Pr[Z|\mathbf{X} = \mathbf{x}]$ and $F(\mathbf{x}, \theta)$ are identically equal [CT06]. This holds for all $\mathbf{x} \in \mathcal{X}$ so the expected value of the KL divergence is also non-negative and equals zero if and only if the latter equals zero almost everywhere on $\mathcal{X}$, which is true since $\mathbf{X}$ is assumed to be a discrete random variable. Hence the equality between $F(., \theta)$ and $\Pr[Z|\mathbf{X}]$. □

Proof of Corollary 2. Straightforward from Corollary 1 by substituting $\mathsf{H}(Z|\mathbf{X})$ with $\mathsf{MI}(Z; \mathbf{X}) - \mathsf{H}(Z) = \mathsf{MI}(Z; \mathbf{X}) - n$. □

Proof of Theorem 2. Straightforward, by combining Theorem 1 and Corollary 2. □

Proof of Lemma 2. Starting from the definition of the PI, and by using the formula of total probabilities for the expected value we have:

$$\begin{aligned}
\mathsf{PI}(Z; \mathbf{X}; \theta) &\triangleq \mathsf{H}(Z) + \sum_{s \in \mathcal{Z}} \Pr[Z = s] \mathop{\mathbb{E}}_{\mathbf{X}|Z=s} [\log_2 F(\mathbf{X})[s]] , \\
&= n + \mathop{\mathbb{E}}_{Z} \left[\mathop{\mathbb{E}}_{\mathbf{X}|Z} [\log_2 F(\mathbf{X}, \theta)[Z]] \right] , \\
&= n + \mathop{\mathbb{E}}_{\mathbf{X},Z} [\log_2 F(\mathbf{X}, \theta)[Z]] , \\
&= n - \mathop{\mathbb{E}}_{\mathbf{X},Z} [-\log_2 F(\mathbf{X}, \theta)[Z]] , \\
&= n - \mathcal{L}_{\Pr[\mathbf{X},Z]}(\theta).
\end{aligned}$$

The proof for the empirical PI follows exactly the same reasoning with averages instead of sums. □

Proof of Corollary 3. Straightforward, by combining Equation 19, Equation 14 and Equation 15. □

C Discussions about the Error Terms

C.1 The Approximation Error

It turns out that the approximation error for Deep Neural Networks is well documented. The Universal Approximation theorem [Pin99] ensures that the set of *Multi-Layer Perceptrons* (MLP) with one hidden layer is dense in the set of continuous conditional pmf over $\mathcal{Z}$, provided that the activation function is not a polynomial. Actually, there is not a single formulation of the Theorem of Universal Approximation, but rather many similar ones for different topologies or different sets of approximated functions. To the best of our knowledge, there is no version of the theorem in the topology of the KL divergence but the version proposed here-after may give satisfaction in our context. Before introducing it, let us first start by setting some necessary definitions in order to introduce the theorem.

Definition 6. Let B^D denote the unit ball in $\mathbb{R}^D$ for the Euclidean norm. Let $\mathcal{W}_2^m$ be the Sobolev space over B^D, namely the *completion* of $\mathcal{C}^m(B^D)$ the space of functions m times differentiable over B^D and whose derivatives at order $|k| \leq m$ are continuous. The Sobolev space is equipped with the following norm:

$$\|f\|_m = \left(\sum_{0 \leq |k| \leq m} \|D^k f\|_2^2 \right)^{1/2} .$$

Finally, we set $\mathcal{B}_2^m = \{f \in \mathcal{W}_2^m : \|f\|_m \leq 1\}$.

We can now introduce the theorem:

Theorem 3 (Universal Approximation Theorem [Pet98])**.** *Let $D \in \mathbb{N}^*$ denote the dimension of the input traces, and let $\mathcal{B}_2^m$ be defined as in Definition 6. Let $\mathcal{M}_r$ be the set of one hidden layer MLP with r hidden neurons activated by the ReLU activation function. Then for all $m \in \left[1, 2 + \frac{D-1}{2}\right]$ we have:*

$$\sup_{f \in \mathcal{B}_2^m} \inf_{g \in \mathcal{M}_r} \|f - g\|_2 \leq \alpha r^{-m/D}, \tag{27}$$

where α is a constant independent of r.

Remark 1. All the Universal Approximation theorems only study the case where the output dimensionality is one. The result can be extended without loss of generality to multi-dimensional ($|\mathcal{Z}|$ in our case) outputs [HSW89, Corollary 2.6].

Theorem 3 says that when the input dimension increases, the number of neurons in the hidden layer must exponentially increase at the same time to keep the approximation error stable. On the opposite, the smoother the approximated function, the larger the value of m such that the approximated function belongs to $\mathcal{B}_2^m$; and the smaller such an exponential rate. This raises the problem of the curse of dimensionality when using Neural Networks. But this worst-case scenario implies that each component of the input is independent from the others, which is rarely the case when considering time series. In our particular case, we may even assume the following statement made in several previous studies [MBvLM12, CDP16, MDP19].

Assumption 1 (Sparsity)**.** *There only exists a small set of coordinates $\mathcal{I}_Z \triangleq \{t_1, \ldots, t_C | C \ll D\}$ such that $F^\star \triangleq \Pr[Z|\boldsymbol{X}] = \Pr\left[Z|\boldsymbol{X}[t_1], \ldots, \boldsymbol{X}[t_C]\right]$. Moreover, $|\mathcal{I}_Z| = \mathcal{O}(d+1)$ where d is the order of the masking scheme used in the implementation.*

In other words, the true pmf is effectively defined on a restricted set of time coordinates which is much smaller than the input trace length. Therefore, the previous theorem can be rewritten as follows:

Proposition 4. *Given Assumption 1, the approximation error depends on the number C of input coordinates that co-jointly depend on the attack target variable:*

$$\sup_{f \in \mathcal{B}_2^m} \inf_{g \in \mathcal{M}_r} \|f - g\|_2 \leq \alpha r^{-m/C}, \tag{28}$$

Proposition 4 tells us that the input dimension of the leakage no longer contributes to the approximation error, thereby breaking the curse of dimensionality. In particular, the size C of $\mathcal{I}_Z$ typically grows with the number of shares, provided the considered implementation is protected by a masking scheme. However, in real cases, the order of masking remains low (*e.g.* below 10), which prevents the approximation error to explode.

As a conclusion of this study of the approximation error we have shown that in the context of Multi-Layer Perceptron, which is a simple type of Neural Nets, this error can be easily controlled with the number of hidden neurons. Moreover, this error does not depend on the input size of the traces, which prevents a curse of dimensionality regularly met in SCA [CDP16, LPB+15]. [11]

[11]It remains an open question whether this discussion actually holds when considering the topology induced by the KL divergence and instead of the norm used in Theorem 3. Empirical verifications of this claim have been provided in Section 4 and Section 5.

C.2 The Estimation Error

The estimation error, as previously introduced in Subsection 3.4, is the error made by optimizing the empirical PI instead of the actual PI. This gap often implies a common flaw in Machine Learning called *overfitting*, namely when the model starts by learning *by heart* the mapping of the finite profiling set S_p. The study of this error is also well documented (see *e.g.* [Vap99, SSBD14]). The literature in Statistical Learning has proposed some bounds on the quality of the result that do not depend on the nature of the true pmf:

Theorem 4 ([Vap99]). *With probability at least $1 - \delta$ the following holds true for all $\theta \in \Theta$:*

$$\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\theta) \leq \frac{\mathcal{L}_{S_p}(\theta)}{(1-\gamma\sqrt{\epsilon})_+}, \textit{ where} \tag{29}$$

$$\epsilon = 4\frac{h\left(\log\frac{2N_p}{h}+1\right)-\log\delta}{N_p} \tag{30}$$

where h denotes the VC-dimension of $\mathcal{H}$, where N_p denotes the number of traces used to compute the NLL loss, and where γ is a constant.

Informally, the VC-dimension $h(\Theta)$ of an hypotheses class quantifies its capacity to contain a model $F(.,\theta)$ that perfectly fits a mapping of at most $h(\Theta)$ points in $\mathcal{X}$ to the set $\{0,1\}$. It often (but not always) approximately equals the size $\#\theta$ of the parameter vector. In particular, we have a specific result for a class of Neural Nets, as proposed hereafter.

Proposition 5 ([SSBD14]). *The VC-dimension of a MLP with one hidden layer made of r neurons and one output neuron (i.e. $|\mathcal{Z}| = 2$) is $\mathcal{O}(\#\theta\log\#\theta)$, where $\#\theta$ is the size of the parameter vector, namely $\#\theta = r(D + |\mathcal{Z}| + 1) + |\mathcal{Z}|$, with D being the size of the input trace and $|\mathcal{Z}|$ being the number of hypothetical values.*

Equation 4 is very strong, since it does not depend on the true conditional pmf $\Pr[Z|\mathbf{X}]$ nor on the value of θ. Indeed, we are only interested in this result for the specific case where θ is the solution of Maximum Likelihood Estimation $\hat{\theta}$.

Unfortunately, the strength of Equation 4 comes with a major drawback: the bound given in Equation 4 is very conservative. Indeed, the denominator in Equation 29 converges towards one with a very slow rate, which cannot give strong guarantees in practise against overfitting. Actually, some regularization techniques make one able to better control such a convergence rate [Vap99], but this is not enough. The so-called *mystery of Deep Learning* states that based on the practical successes of such models, there should be other bounds less conservative [Har].

Hopefully, the estimation error can still be experimentally evaluated thanks to a *hold-out* set of profiling traces. Concretely, we also compute the empirical PI on a hold-out set (or validation set) S_v made of N_v profiling traces that have not been used to minimize the NLL loss. Therefore, the estimation of PI from S_v will be unbiased. This is a classical technique used in Machine Learning. But in our context, we might also need some confidence interval for this estimation. To this end, we can use the Chebycehv's inequality (see Appendix D) to have an interval with a conservative confidence. This will be useful especially when the MI (which we recall to be an upper bound of the PI) will be very low, in order to get an hypothesis test for the question "Is $\mathsf{PI}\left(Z;\mathbf{X};\hat{\theta}\right) > 0$?", or equivalently "Did my model extract some information from my traces?".

Provided with this accurate tool for estimating the true PI, the estimation error will naturally appear as the gap between $\widehat{\mathsf{PI}_{S_p}}\left(Z;\mathbf{X};\hat{\theta}\right)$ and $\widehat{\mathsf{PI}_{S_v}}\left(Z;\mathbf{X};\hat{\theta}\right)$.

The takeaway message of this discussion is that when applying the Maximum Likelihood Estimation with MLPs, one cannot get useful theoretical guarantees on the convergence rate of the consistency property given by Theorem 1. Hopefully, this is not a fatality, since in practise, the convergence rate is much faster, and can be accurately estimated thanks to the hold-out set.

C.3 The Optimization Error

It is recalled that the optimization error quantifies the fact that the solution to the Maximum Likelihood Estimation cannot be directly found. Instead, the use of Stochastic Gradient Descent (SGD) is empirically found to give a sound approximation of such a solution. Actually, SGD is shown to converge to the exact solution when the loss function is convex [SSBD14]. This cannot be assumed for Neural Nets [CHM+15]. Therefore, this could lead to sub-optimal models. We will show in Section 5 a case where we can emphasize problems due to optimization.

D Confidence Interval with a Hold-Out Set

As described in Subsection C.2, the bounds on the estimation error might be too high in practical uses with Deep Neural Nets. This is why usually, the estimation of the Cross Entropy is done otherwise. In cases where the evaluator has lots of data, he can take a so-called *hold-out* that will be distinct from the profiling set S_p for the minimization of the NLL loss. Therefore, these *fresh* data will give a more correct estimation of the Cross Entropy.

Lemma 3 (Chebychev's inequality [SSBD14])**.** *Let $\mathcal{H}$ be a parametrized hypothesis class and Θ its corresponding parameter space. Let $S_v \triangleq \{(\boldsymbol{x}_1, z_1), \ldots, (\boldsymbol{x}_{N_v}, z_{N_v})$ be a hold-out set of N_v i.i.d. leakages and the corresponding values of the sensitive target variable. Assume that $\mathbb{V}(-\log_2 F(\boldsymbol{X}, \theta)[Z]) \leq 1$. Then for all $\theta \in \Theta$, it holds with probability at least $1 - \delta$ that:*

$$|\mathcal{L}_{\Pr(\boldsymbol{X},Z)}(\hat{\theta}) - \mathcal{L}_{S_v}(\hat{\theta})| \leq \sqrt{\frac{1}{\delta N_v}} \tag{31}$$

Equation 31 will be used in Section 4 and Section 5 to estimate provide a conservative confidence interval of the Cross Entropy and thereby the information between a sensitive target variable and a leakage perceived by a Neural Net (PI).

E Source Code for the Acquisitions

Algorithm 1 loadData

```
1: LD r0, X        ▷ Loads the first byte in r0
2: CLR r0          ▷ Clears the register
3: ST X, r0        ▷ Stores 0 in the plaintext array
4: LD r0, X        ▷ Do it again to clear the bus
5: CLR r0
6: ST X, r0
7: LD r0, X        ▷ One more time to be sure
8: CLR r0
9: ST X+, r0
```

F The Experimental Traces

To verify that there is no leakage implying a combination of different bytes, we have also computed a SNR of *order 2*. That is to say that for each combination of 2 among the 16 bytes, the `xor` has been computed and used as a target variable in order to compute the SNR. The absence of peaks confirms that there is no undesirable leakage. An example of a trace and the SNRs of order 1 and 2 can be found in Figure 6.

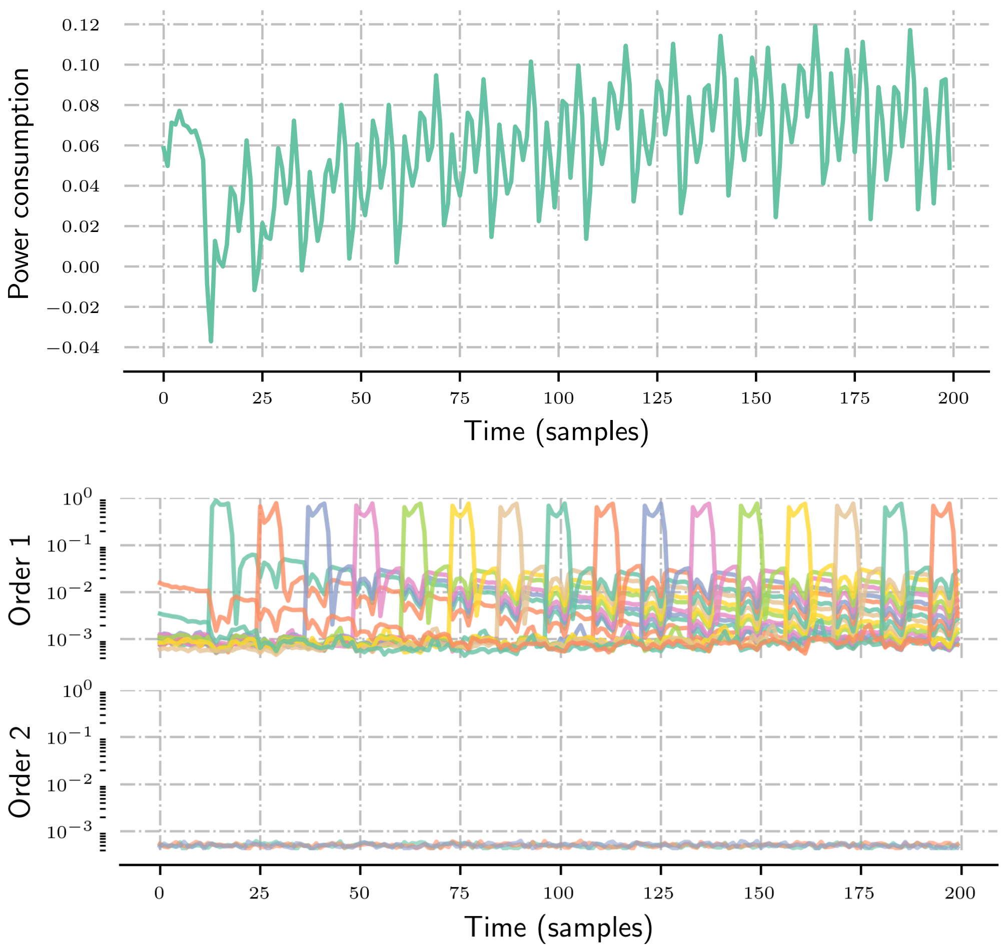

Figure 6: Top: an example of trace. Bottom: the SNRs of order 1 and 2 (in logarithm scale).

F.1 The hypothesis class $\mathcal{H}$

The hypothesis class $\mathcal{H}$ that will be used for the experiment has been defined as the set of MLP with one hidden layer and $r = 500$ hidden neurons. In other words, there are two *linear* layers: the hidden one, denoted by λ_{θ_1}, and the output one denoted by λ_{θ_2}. Symbols θ_1 and θ_2 denote the associated real parameters of the hidden layer and the output layer respectively. Between these linear layers, an *activation* layer called *ReLU* and denoted by σ is added. This is a non-linear real valued function that is responsible of the high capacity of MLP to approximate any pmf [GBC16].

In addition, two *batch normalization* layers [IS15] have been applied at the input of each linear layer. Batch Normalization (BN) layers simply normalize the input features to a mean and a deviation that are automatically set by the SGD algorithm. BN has been shown to make the loss function smoother, making the optimization easier and faster [STIM18]. Finally, a *dropout* layer [SHK$^+$14, GBC16] has been added on the input of the softmax classifier. Dropout is known to prevent DNNs from *overfitting* (*i.e.* to prevent the estimation error to explode), which is useful when one lacks data. The dropout parameter has been set to $p = 0.1$ *i.e.* each neuron of the hidden layer is randomly set to 0 with probability p each time an output $F(\mathbf{x}, \theta)$ is computed during the optimization.

All together, we can sum up the architecture of our MLP as follows:

$$F(\mathbf{x}, \theta) = s \circ \lambda_{\theta_2} \circ \mu \circ \delta_p \circ \sigma \circ \lambda_{\theta_1} \circ \mu(x) \tag{32}$$