# Unsupervised Side-Channel Analysis Based on Mutual Information and its Neural Estimation

Valence Cristiani

**THESIS TO OBTAIN THE DEGREE OF DOCTOR**
**OF THE UNIVERSITY OF MONTPELLIER**

**In Information Security**

**Doctoral School: Information, Structures and Systems sciences (I2S)**

**Research Unit: LIRMM**

# Unsupervised Side-Channel Analysis Based on Mutual Information and its Neural Estimation


**Presented by Valence CRISTIANI**


**December 07, 2022**

**Under the supervision of:**
**Philippe MAURINE and Maxime LECOMTE**

**In front of the jury composed of:**

| | |
|---|---|
| Louis GOUBIN, Professor, Université de Versailles Saint-Quentin | Reporter |
| Matthieu RIVAIN, Security Expert, CryptoExperts, Paris | Reporter |
| Emmanuel PROUFF, Security Expert, Paris | Examiner |
| François-Xavier STANDAERT, Professor, Université Catholique de Louvain | Examiner |
| Christophe CLAVIER, Professor, Université de Limoges | Examiner |
| Bruno ROUZEYRE, Professor, Université de Montpellier, LIRMM | President of Jury |
| Philippe MAURINE, Associate Professor, Université de Montpellier, LIRMM | Supervisor |
| Maxime LECOMTE, Research Scientist, CEA-Leti, Grenoble | Co-supervisor |
| Thomas ROCHE, Security Expert, NinjaLab, Montpellier | Invited member |
| Eleonora CAGLI, Research Scientist, CEA-Leti, Grenoble | Invited member |

UNIVERSITÉ DE MONTPELLIER

*"Yes, I am a criminal. My crime is that of curiosity. My crime is that of outsmarting you, something that you will never forgive me for. I am a hacker, and this is my manifesto. You may stop this individual, but you can't stop us all... after all, we're all alike."*

The Mentor (Loyd Blankenship)

UNIVERSITY OF MONTPELLIER

# *Abstract*

Doctor of Philosophy

**Unsupervised Side-Channel Analysis Based on Mutual Information and its Neural Estimation**

by Valence CRISTIANI

Side-Channel Analysis (SCA) is defined as the process of gaining information on a device holding a secret through its physical leakage such as power consumption or Electromagnetic (EM) emanations. Whatever the utilized strategy, the amount of information one could gain from a side-channel data, called a trace, is always bounded by the Mutual Information (MI) between the secret and the trace. This makes it, all punning aside, a key quantity for leakage evaluation. Unfortunately, traces are usually of too high dimension for classical statistical estimators to stay sound when computing the MI over full traces. However, recent works from the machine learning community have shown that it is possible to evaluate the MI in high dimensional space thanks to newest deep learning techniques. This thesis explores how this new estimator impacts the side-channel domain.

The first part is dedicated to an analysis of the Mutual Information Neural Estimation (MINE) technique in a side-channel context which aim is to derive the best way of using such tool in practice. It shows that the intrinsic multi-dimensional aspect of the technique is highly valuable for SCA since there are often multiple leakage sources in side-channel traces. The method is derived as a generic leakage assessment tool that can be used whatever the type of data, device or implementation.

Knowing how much information is contained in the traces is different from knowing how to exploit it optimally to recover a secret such as a cryptographic key, especially in an unsupervised context when no profiling of the target is allowed. Therefore, the second part of this thesis presents a new mathematical framework, designed to bridge classical Mutual Information Attacks (MIA) and the multidimensional aspect of neural-based estimators. This allows to derive, to the best of our knowledge, the first unsupervised attack able to benefit from both the power of deep learning techniques and the valuable theoretical properties of MI. In practice this attack suffers from two drawbacks : the time complexity, since it requires as many network trainings as there are key hypotheses (often 256), and a strong a priori on the leakage model of the target device.

The third part of the thesis makes use of the previously introduced mathematical framework to build a deep learning architecture able to recover by itself such a leakage model. It allows to derive a new unsupervised attack, the EVIL Machine Attack, with only one network training solving the two precedent issues at the same time.

The analysis of the EVIL machine in the context of masked implementations gave rise to questions about stochastic attacks and their generalization to higher-order versions. The last part of this thesis is dedicated to an analysis followed by a new unsupervised attack proposition, the Joint Moment Regression, which is agnostic to the underlying masking scheme as opposed to state-of-the-art techniques.

# *Acknowledgements*

J'ai vécu cette thèse comme une aventure. Une aventure scientifique, intellectuelle mais surtout une aventure humaine. C'est pourquoi j'aimerais profiter de ces quelques lignes pour prendre le temps d'affirmer toute ma reconnaissance envers les personnes qui m'ont accompagné, aidé et avec qui j'ai partagé des moments inoubliables durant ces trois belles années.

Tout d'abord, si j'ai choisi de me lancer dans cette thèse, c'est grace à Maxime, mon encadrant, que j'ai rencontré lors de mon stage de fin de master au CEA. J'avais l'intuition que nos personnalités pourraient correspondre et je peux dire *a posteriori* que, au moins selon mes critères d'encadrement, Maxime était proche de l'optimalité (et ce dès la première *epoch* ce qui est plutôt impressionnant dans notre domaine). Il a su me laisser une grande liberté en me faisant confiance même lorsque je partais dans des directions qui semblaient un peu folles, toujours en m'accompagnant avec un mélange de bienveillance et de pertinence (deux qualités que j'ai rarement trouvées si bien réunies chez une même personne). Merci Maxime pour tout cela et j'éspère que tu me pardonnera toutes les fois où je t'ai retenu jusqu'à la fermeture du CEA. Nos discussions étaient tout simplement trop passionantes...

J'aimerais également témoigner toute ma gratitude envers Philippe, mon directeur de thèse. Il a su malgré la distance, m'accompagner dans ma thèse, en montrant toujours beaucoup d'intérêt pour mes idées, tout en les challengeant de manière constructive. Les questions que nos discussions soulevaient m'amenaient très souvent à formaliser les concepts et participaient largement à l'émergence de nouvelles idées. Un grand merci Philippe pour ta bienveillance et ta dévotion à la transmission, j'ai toujours eu le sentiment que tu ne laisserais jamais tomber un doctorant, même quand il s'agit de relire un papier imbittable un dimanche soir, avant une deadline !

Je voudrais aussi remercier particulièrement Thomas Hiscock, mon encadrant de stage. Il m'a permis de rejoindre le CEA, m'a donné le goût pour le domaine des attaques par canaux cachés et à continuer de suivre mes travaux tout au long de ma thèse. Merci aussi pour les innombrables coups de pouce techniques, sans lui je serais toujours en train d'installer tensorflow-gpu...

Je souhaite remercier sincèrement Louis Goubin et Matthieu Rivain, rapporteurs de mon manuscrit, d'avoir produit des rapports particulièrement pertinents malgré la contrainte de temps demandée. Je remercie également Emmanuel Prouff, François-Xavier Standaert, Christophe Clavier, Bruno Rouzeyre, Thomas Roche et Eleonora Cagli d'avoir tous très gentiment accepté de prendre part à mon jury de thèse.

Si l'encadrement joue un role central, il n'est certainement pas le seul ingrédient nécessaire à une thèse épanouïe. J'ai eu la chance d'évoluer dans un laboratoire avec une ambiance fantastique. J'aimerais remercier tous mes collègues et amis doctorants avec qui j'ai partagé le plus clair de mes journées depuis trois ans. Merci Raphaël pour ton authenticité, merci à Thomas et Romain pour la dynamique positive que vous insufflez dans le labo (ps: j'emm**** Caro !). Merci à Julien pour ta gentillesse et ta curiosité, merci Gaëtan pour toutes ces soirées karaokés absolument nécessaire à la réussite d'une thèse, merci également à Ninon, Camille et Nina d'avoir égayé ces soirées (et pleins d'autres moments comme ces bivouacs incongrus), merci à Amine, mon compagnon au CEA from day one, merci à Nikos, Dorian, Jonathan et en particulier à Guillaume pour nous avoir si souvent tous réunis.

Un grand merci à mes anciens collègues du LSOSP, en particulier à Jacques, Alexis, Antoine, Meriem, Soraya, Sylvain pour m'avoir accueilli et guidé lors de mon arrivée. Je remercie Loïc Masure (et de nouveau à FX) pour les discussions passionnantes au sujet de l'apprentissage des schémas de masquage par les réseaux neuronaux et pour la collaboration qui en a découlé. Merci également à Victor Lomne pour l'ensemble de nos discussions qui m'ont amené à devenir un Ninja :).

J'aimerais remercier de tout cœur mes proches pour leur soutien infaillible. Merci Alice de m'avoir ensoleillé lors de la rédaction de ce manuscrit et merci à Sandrine et Valéry de m'avoir hebergé pendant cette même periode. Merci également à mes grand-parents pour leur indéfectible soutien ainsi que pour l'hospitalité dont ils font preuve et qui m'est si chère chaque fois que je viens à Paris. Enfin et surtout un immense merci à ma mère pour la quantité, si grande que je ne pourrais tous les citer, de services rendus pendant cette thèse et surtout pour son amour inconditionel qui me guide au quotidien.

Pour conclure, je tiens aussi à remercier mon réveil, qui, ayant eu la bonne idée de ne jamais sonner durant ces trois années, m'a permis de rester toujours en forme, prêt à festoyer avec les équations.

$$(x^2 + y^2 - 1)^3 - x^2 y^3 = 0$$

# Contents

# List of Figures

# List of Publications

## Main Works

1. Leakage Assessment Through Neural Estimation of the Mutual Information - Valence Cristiani, Maxime Lecomte & Philippe Maurine - Published in International Conference on Applied Cryptography and Network Security (ACNS) 2020.

2. Revisiting Mutual Information Analysis: Multidimensionality, Neural Estimation and Optimality Proofs - Valence Cristiani, Maxime Lecomte & Philippe Maurine - Submitted to Journal of Cryptology (JoC).

3. The EVIL Machine: Encode, Visualize and Interpret the Leakage - Valence Cristiani, Maxime Lecomte & Philippe Maurine - Submitted to Symposium On Applied Computing (SAC) 2023.

4. Fit the Joint Moments: How to Attack any Masking Schemes - Valence Cristiani, Maxime Lecomte, Thomas Hiscock & Philippe Maurine submitted - Submitted to IEEE Open Access.

## Side Works

5. A Bit-Level Approach to Side Channel Based Disassembling - Valence Cristiani, Maxime Lecomte & Thomas Hiscock - Published to CARDIS 2019.

6. Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking - Loïc Masure, Valence Cristiani, Maxime Lecomte & Fançois-Xavier Standaert - Published to Cryptographic Hardware and Embedded Systems (CHES) 2023.

*À mon père, qui m'a énormément inspiré, et avec qui j'aurais adoré partager les découvertes et les questionnements qui m'ont été offerts durant ces trois années.*

# Chapter 1

# Context and motivations

*"If you reveal your secrets to the wind, you should not blame the wind for revealing them to the trees."*

Kahlil Gibran, *The Wanderer*

## 1.1 Introduction to Cryptography

Everyone remembers that day in primary school when this message written on a small piece of paper, sent by a little boy discovering the wriggling of love, was intercepted by the teacher while passing from hand to hand with the ultimate goal of getting to Ema. A simple shift of each letter three letters further down the alphabet would have saved him from the intense feeling of shame of having the whole class know that he had just declared his love to Ema. Unfortunately, he only discovered cryptography a few years later...

The term *cryptography*, from the greek *kryptós* (secret) and *graphein* (writing), refers to the art of encrypting messages. It encompasses all the techniques useful for a secure communication in the presence of an adversary. The basic idea is to transform a private message, the *plaintext*, into an unintelligible message, the *ciphertext*, which should appear like nonsense to anyone who intercepts it. The only way to understand this message would be to reverse to process of encryption, which often involves owning a secret. As stated by A. Kerckhoff in his famous principle,

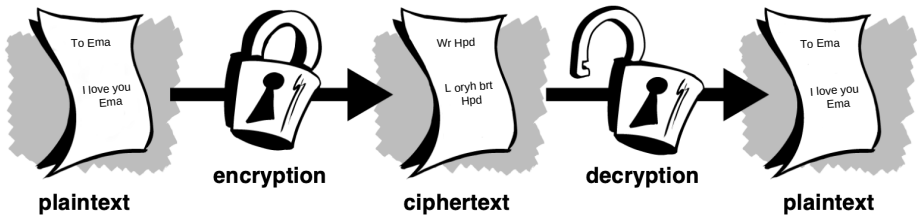



FIGURE 1.1: Generic cryptographic scheme

the general specifications of the encryption scheme should not be part of this secret. The encryption procedure should rely on a secret key, often shared by the two parties trying to communicate, but everything else about the system should be public knowledge. For example, in Caesar's code described above, the fact that letters may be shifted by a certain amount would be public knowledge and the specific shift of three would be the shared secret. Obviously, under these conditions, the latter scheme does not feel very secure. One could easily try the 26 possible shifts and recover the private message, thereby conducting the most basic cryptanalysis technique...

*Cryptanalysis* is the branch of *cryptology* (the science of the secret) studying the resistance of cryptographic schemes. The main goal is to develop attacks in order to assess to what extent a malicious adversary could break a crypto-system's security claims. Cryptography and cryptanalysis are complementary domains that evolve in symbiosis with the aim of converging toward secure algorithms.

The art of cryptography is very ancient but has experienced tremendous growth in the last century, with the advent of computers and the general rise of information technology. It has become increasingly complex and is now considered a proper science, often called *Modern Cryptography*, taking place at the intersection of many disciplines such as applied mathematics, computer science, electrical engineering, and information security. Modern cryptographic protocols are designed to ensure at least one of the four following information security properties:

1. *Confidentiality*: The private message should only be retrievable by a set of authorized parties.
2. *Integrity*: the receiver should be convinced that the message has not been modified during transmission.
3. *Authenticity*: the receiver should be able to identify the sender of the message.
4. *Non-repudiation*: The sender should not be able to deny sending the message afterward.

Low-level cryptographic routines, often called cryptographic primitives, are the basic blocks used to build cryptographic protocols. A primitive is considered mathematically secured if an adversary, granted with the ability to query as many couples (plaintext, ciphertext) as he wants, can not recover the secret key in a reasonable amount of time.

Two branches of cryptography may be distinguished: the *symmetric* and *asymmetric* cryptography. The first one, known as secret key cryptography, is the most ancient one and is predicated on the idea that the two entities involved have shared a common secret prior to the communication. The second, also known as public key

cryptography, was first introduced in the 1970s and differs from symmetric cryptography in the fact that the key used for the encryption is not the same as the one used for the decryption. Such schemes are computationally heavier and are often only used to securely share a symmetric key between two parties. Even though some ideas may be extended to asymmetric cryptography, a significant part of this thesis focuses on analyzing the security of symmetric crypto-systems. Most of the examples and experiments are conducted on the Advanced Encryption Standard (AES) which is the most used symmetric cryptographic primitive nowadays.

## 1.2 Side-Channel Analysis

Modern cryptography proposes solutions to secure communications. Until the middle of the nineties, cryptographic algorithms were considered as a black-box in the sense that an adversary can only observe the inputs and outputs of a computation. Therefore, the actual security of the system was equivalent to the mathematical security of its primitives. However, in his seminal paper about Side-Channel Analysis (SCA) [Koc96], Paul Kocher showed that this black-box model may not always be the appropriate adversary's model. Indeed, cryptographic algorithms are implemented on electronic devices which inevitably manipulate the secret keys. Such manipulations involve physical processes, mostly related to the flow of electrons in semiconductors, and may thus, induces observable variations in the surrounding of the device.

These variations may take different forms: the so called side-channels. Any measurable physical quantity carrying information about the secret can be considered a side channel. The most famous examples are the instantaneous power consumption of a device [KJJ99] and its ElectroMagnetic (EM) emanations [QS01]. Figure 1.2 shows an example of the EM emanation captured by an EM probe positioned close to a device executing an AES. Such acquired data are often called side-channel traces. Many other side-channels have been pointed out in the literature such as timing attacks [BT11], cache monitoring [Per05] or even network packets length analysis [Sch+14].

Secret keys take the form of bit string long enough so that the number of possible combinations would far exceed the brute force capability of any adversary. For example, the AES in its lowest security setting involves 128 bits secret keys leading to $2^{128}$ key possibilities. Enumerating all these possible values would be longer than the age of the universe. The main idea of a side-channel attack is to gather enough information, via the leakage channel, so that the amount of remaining possibilities becomes enumerable in a reasonable time. It often involves a divide and conquer strategy. Indeed, algorithms use key chunks to perform independent computations which may induce leakage on intermediate variables only depending on a sub-part of the key. For example, in the AES the internal variables are naturally processed at

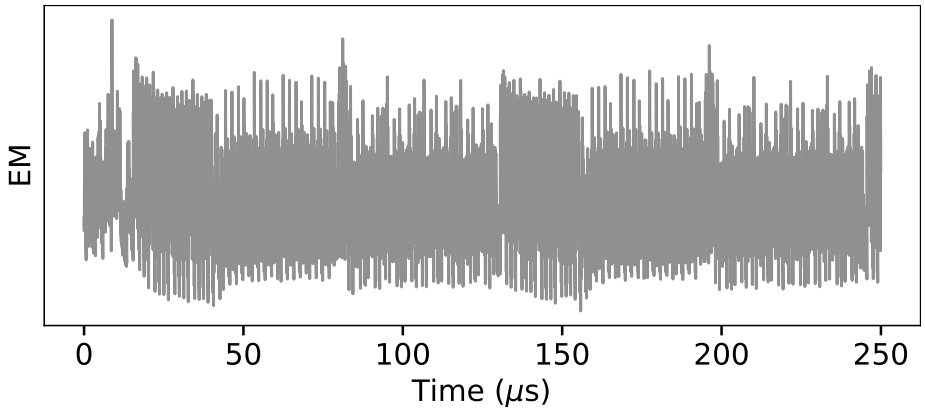



FIGURE 1.2: Electromagnetic trace of an AES

the byte level and a typical SCA strategy would be to focus only on one specific key byte and repeat the process 16 times to recover the whole key.

### 1.2.1 Notations

**Probabilistic notations**

Random variables are represented as upper-case letters such as $X$. They take their values in the corresponding set $\mathcal{X}$ depicted with a calligraphic letter. The carnality of $\mathcal{X}$ is denoted $|\mathcal{X}|$. Lower-case letters such as $x$ stand for elements of $\mathcal{X}$. The probability density function associated to the variable $X$ is denoted $P_X$ (replaced by $P$ when there is no ambiguity). The notations $P_X(X = x)$ and $P_X(x)$ are equivalent. If $f$ is a function defined over $\mathcal{X}$, $\mathbb{E}_X[f(X)]$ denotes the expected value of the random variable $f(X)$, under the distribution $P_X$ so that:

$$\mathbb{E}_X[f(X)] = \sum_{x \in \mathcal{X}} P_X(x) \cdot f(x) \tag{1.1}$$

Such notation is replaced by $\mathbb{E}[f(X)]$ when there is no ambiguity. Similarly, the variance of $f(X)$ is denoted by $\mathrm{Var}_X[f(X)]$ (or $\mathrm{Var}[f(X)]$) , :

$$\mathrm{Var}_X[f(X)] = \sum_{x \in \mathcal{X}} P_X(x) \cdot (f(x) - \mathbb{E}_X[f(X)])^2 \tag{1.2}$$

When two random variables $X$ and $Y$ are considered, their joint probability density function is denoted by $P_{X,Y}$. The conditional probability of $X$ given $Y$ is denoted $P(X \mid Y)$. The covariance of the two variables is denoted by $\mathrm{cov}(X, Y)$:

$$\mathrm{cov}(X, Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P_{X,Y}(x, y) \cdot (x - \mathbb{E}_X[X]) \cdot (y - \mathbb{E}_Y[Y]) \tag{1.3}$$

Eventually, when a set of realization $(x_i)_{1\leq i\leq N}$ of $X$ is available, the hat notation represents a statistical estimation of the underlying quantity. For example:

$$\widehat{\mathbb{E}}_X[f(X)] = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{1.4}$$

**SCA framework**

In this thesis, the evaluation/attack framework is described considering that an adversary targets the manipulation of a sensitive variable $Z \in \mathcal{Z} = \mathbb{F}_2^n$, for a given $n \in \mathbb{N}$. This variable is supposed to functionally depends on a public variable $X \in \mathcal{X} = \mathbb{F}_2^m$, for a given $m \in \mathbb{N}$, and a secret key chunk $k^* \in \mathcal{K} = \mathbb{F}_2^m$ through the relation: $Z = g(X, k^*)$ where $g : \mathcal{X} \times \mathcal{K} \to \mathcal{Z}$ is a known function depending on the underlying cryptographic algorithm. For simplicity, $g(X, k)$ is denoted $g_k(X)$ in the rest of the thesis.

The evaluator/adversary is supposed to be able to acquire a set $\{(\ell_i, x_i), 1 \leq i \leq N\}$ of $N$ side-channel traces labeled with the corresponding public value of $X$. Depending on the situation, the latter may have control over the $X$ variable (be able to choose it) or not. Traces correspond to realizations of a leakage variable $L \in \mathcal{L}$ coming from a stochastic process $\mathcal{S}$, $Z \xrightarrow{\mathcal{S}} L$. When the sensitive variable is directly processed in the internal of the device (*i.e.* no masking countermeasure, see section 1.2.4) the leakage $L$ is supposed to be separable into a deterministic and a noise part:

$$L = \varphi(Z) + N \tag{1.5}$$

where $\varphi$ is a function from $\mathcal{Z}$ to $\mathbb{R}^p$, for some $p \in \mathbb{N}$, denoted the leakage model in this thesis and where $N$ represents an independent noise distribution (which is often supposed to be normal). Note that in this thesis, the expression *Leakage model* stands for the deterministic part of the true leakage which may differ from other SCA materials. In addition, we consider the leakage model to be potentially multi-dimensional: it encompasses the whole side-channel trace which can be composed of thousands of time samples (for example, Figure 1.2 contains $p = 50k$ samples).

### 1.2.2 Leakage Assessment

A first step that a designer/adversary may conduct to identify a side channel, is to perform a leakage detection. Many techniques have been developed for this purpose and are known as leakage assessment techniques. The main goal is to identify if L contains information about $Z$. A minimum requirement for a leakage to occurs is that there exist $z_1$ and $z_2$ such that $\varphi(z_1) \neq \varphi(z_2)$. The more values of $Z$ with a different model, the better the leakage. However, these differences should also be compared to the amount of noise present in the traces. Indeed a low difference between $\varphi(z_1)$ and $\varphi(z_2)$ drowned in the noise would not be very informative.

The main problem to characterize differences between values of $\varphi(Z)$ is that $\varphi$ is often a highly multidimensional function since it represents the whole trace. In addition, the leakage may only occur in small specific areas of the traces. In other words, even in the case of a leakage, $\varphi(z_1)$ and $\varphi(z_2)$ represent long vectors that are very likely equal on most of their components. That is why a classical solution to leakage assessment is to compute sample-wise statistics $S_L(s)$, for $1 \leq s \leq p$, on the variables $L[s]$ representing the projection of the leakage on one of its components. The aim of such statistics is to quantify a sort of instantaneous signal strength, transforming a non-trivial multi-dimensional problem into a simple monodimensional one. There exists multiple of these statistics such as the Sum of Differences (SoD) [RO05], the Sum of Squared T-difference (SoST) [GLRP06], the Normalized Inter-Class Variance (NICV) [Bha+14]. We give here as an example, one of the most famous of them which is the classical Signal-to-Noise Ratio (SNR). It is defined as:

$$\mathrm{SNR}_L(s) = \frac{\mathrm{Var}_Z\big[\mathbb{E}[L[s] \mid Z]\big]}{\mathbb{E}_Z\big[\mathrm{Var}[L[s] \mid Z]\big]} \tag{1.6}$$

The intuition behind this formula is that it compares the variance of the signal inter-class (representing the useful part of the signal) to the intra-class variance (representing the noise). A peak of SNR corresponds to a leakage area in the traces.

Since the essence of such statistic is to assess if there exists some kind of information between $L$ and $Z$, it is also possible to use the Mutual Information (MI) which is exactly designed for such purpose as explained in section 1.4:

$$\mathrm{MI}_L(s) = \mathcal{I}(Z,\ L[s]) \tag{1.7}$$

The way to conduct the MI estimation is not discussed here as it is the object of Chapter 2. However, as shown in [PR09] it can be easily estimated when dealing with univariate variables.

**Remark 1.** *One may argue that such computation cannot be done by an adversary in a black-box setting because it requires the knowledge of $Z$ and therefore of the correct key chunk $k^*$. However, in many cases, the $g$ function linking $Z$ and $X$ for a fixed $k^*$ is a bijective function. For example, in the AES case a classical choice of sensitive variable is $Z = Sbox[X \oplus k^*]$. This implies that partitioning traces according to $Z$ is the same as partitioning them according to $X$. In such a cases, Equation 1.6 and Equation 1.7 can be replaced respectively by:*

$$\mathrm{SNR}_L(s) = \frac{\mathrm{Var}_X\big[\mathbb{E}[L[s] \mid X]\big]}{\mathbb{E}_X\big[\mathrm{Var}[L[s] \mid X]\big]} \tag{1.8}$$

$$\mathrm{MI}_L(s) = \mathcal{I}(X,\ L[s]) \tag{1.9}$$

The two main drawbacks of such univariate statistics are the following:

1. They can not detect higher-order leakages, in other words, they would fail to detect a leakage that comes from the dependency of multiple samples. Such leakage models are widely met when dealing with masked implementations (see section 1.2.4).

2. They can not accumulate several small leakages coming from multiple different sources. They would fail to detect a leakage if all the instantaneous leakages stay under the detection threshold while the aggregation of all these *small* leakages would still make the trace informative.

#### Higher-Order Detection

The previously discussed univariate statistics could be extended in order to detect higher-order leakages. If one wants to detect up to a $d$-order leakage the main idea is to combine all the possible $d$-tuple of time samples from the trace through a *Combining function* $C : \mathbb{R}^d \to \mathbb{R}$ and to apply the univariate statistic on the output of $C$. Formally, for all $(s_1, \ldots, s_d) \in [\![1,\ p]\!]^d$ one can compute:

$$S_L^{(C)}(s_1, \ldots, s_d) = S_L(C(L[s_1], \ldots, L[s_d])) \tag{1.10}$$

The choice of the combining function is not trivial. Some proposals have been made in [PRB09; OM06]. A common choice is the centered product combining function[1]. Even without considering the choice of the combining function, this method suffers from a major drawback. Indeed, the number of possible $d$-tuple increases exponentially with $d$ and becomes quickly critical even for low values of $d$.

In addition, even though such a strategy solves to some extent the first drawback of the univariate method, it does not bring anything regarding the second drawback related to the problem of aggregating enough small leakages to pass above a detection threshold.

### 1.2.3 Side-Channel Attacks

Any side-channel attack aiming at recovering a key chunk $k^* \in \mathcal{K}$ can be summarized in the following two-step procedure:

1. Acquire a set of traces $\mathfrak{L} = \{(\ell_i, x_i), 1 \leq i \leq N\}$ labeled with their corresponding public variable.

[1] The centered product is related to the $d$-order centered joint moment of a distribution and thus, may keep $d$-order statistical information.

2. Apply a distinguisher function mapping any set of traces $\mathfrak{L}$ to a scoring vector in $\mathbb{R}^{|\mathcal{K}|}$:

$$\mathcal{D} : \mathfrak{L} \rightarrow \begin{pmatrix} \vdots \\ \mathcal{D}_{\mathfrak{L}}[k] \\ \vdots \end{pmatrix} \tag{1.11}$$

Such procedure will then be repeated for all key chunks (obviously, the same set of traces can be re-used if they contain leakage related to multiple key chunks). Then, the adversary will feed all the scoring vectors into a key enumeration algorithm [VC+13; PSG16] which should test the keys one by one from the most probable to the least probable according to the scoring vector of each key chunk (most of the times, the scores are normalized to be treated as probabilities). The test usually considers a given couple (plaintext, ciphertext) and consists in encrypting the plaintext and comparing it to the ciphertext. The exact design of such enumeration algorithms is out of the scope of this thesis but as a rule of thumb the better the score of each correct key chunk, the faster the algorithm will find the full key. Even if a correct key chunk is ranked first through the distinguisher one should also consider its *distinguishability* with respect to the other hypothesis because it affects the final enumeration.

Therefore, an exciting part of side-channel attacks lies in the design and choice of such a distinguisher. A first observation from [WOS14] is that there does not exist a generic distinguisher that would asymptotically (with respect to the number of acquired traces) work whatever the leakage model $\varphi$. This means that the adversary has to use some kind of *a priori* knowledge on the leakage model of the target to choose a distinguisher. Side-channel attacks are mostly divided into two categories: the supervised attacks where this *a priori* comes from a preliminary profiling phase of the target and the unsupervised attacks where this *a priori* comes from pure physical knowledge about how electronics works and from assumptions on the device being targeted.

#### Supervised Attacks

Supervised attacks correspond to a strong adversary's model where the latter is supposed to be able to conduct a profiling phase of the target with full control on the device, prior to the attack. Such a profiling phase is a *machine learning* task whose goal is to learn either a *generative* or a *discriminative* model of the leakage from labeled examples. A generative model is an estimation of the conditional probability $P(L \mid Z)$. A discriminative model is an estimation of the conditional probability $P(Z \mid L)$. Both are linked through the well-known Bayes' theorem:

$$P(Z \mid L) = \frac{P(L \mid Z) \cdot P(Z)}{P(L)} \tag{1.12}$$

Given a set of attack traces $\mathfrak{L} = \{(\ell_i, x_i), 1 \leq i \leq N\}$, a common way to derive a distinguisher from a discriminative model is to define:

$$\mathcal{D}_{\mathfrak{L}}[k] = \prod_{i=1}^{N} P(Z = g(x_i, k) \mid L = \ell_i) \tag{1.13}$$

If the model is generative, one can replace $P(Z = g(x_i, k) \mid L = \ell_i)$ by $P(L = \ell_i \mid Z = g(x_i, k)) \cdot P(Z = g(x_i, k))$, since the denominator $P(L)$ is independent from the key and would not change the ranking. A classical example of generative models used in supervised attacks is the Gaussian template attacks [CRR02] which estimates each class $P(L|Z = z)$ assuming that it is a multivariate Gaussian distribution. Deep neural network based attacks, mentioned in subsection 1.3.2, constitutes a common example of discriminative model.

If the model, either generative or discriminative, is perfectly learned during the profiling phase, these attacks are known to be optimal from an information theory point of view. Therefore, research in this field mostly resides in practical improvements of machine learning tools (essentially deep learning tools in the last 5 years) in order to improve the estimation of probability distributions in an SCA context. The main drawback of these attacks is their assumption on the adversary's capability. It is often impossible to run the profiling phase on the target device itself. A more realistic assumption is that the adversary may acquire a clone of the device over which she would have full control. However, it is not always easy to find the exact same model and setup than the targeted device and it introduces the template portability problems [EG12b] due to variations in the manufacturing process.

#### Unsupervised Attacks

As opposed to supervised attacks, unsupervised attacks do not require preliminary profiling of the target, constituting a broader threat since they imply weaker assumptions on the adversary model. Since they do not suppose prior specific information on the device, one has to use physical *a priori* to design a sound distinguisher. Such *a priori* is usually related to the binary representation of the sensitive variable since processors manipulate bits. There exist multiple distinguishers translating different types of *a priori* into an actual strategy. They consist in computing, for all key candidate $k$, a certain statistic from the traces under the assumption that the sensitive variable is $Z_k = g(X, k)$. Since statistics are usually well suited for univariate data, they suffer from the same problem as leakage assessment techniques. The classical method is to compute a sample-wise statistic on all the samples of the traces and to retain the maximum (or minimum) as a score.

**Differential Power Analysis.** Historically, one of the first proposed *a priori* is to suppose that a particular bit of the sensitive variable induces differences in the leakage when set to 0 or 1. This gives rise to the so called Differential Power Analysis (DPA) [KJJ99]. In such case one can define the following distinguishers:

$$\mathcal{D}_{\mathfrak{L}}^{(\mathrm{DPA})}[k] = \max_{s} \left[ \frac{1}{|\mathcal{A}_k|} \sum_{\ell \in \mathcal{A}_k} \ell[s] - \frac{1}{|\mathcal{B}_k|} \sum_{\ell \in \mathcal{B}_k} \ell[s] \right] \tag{1.14}$$

where $\ell[s]$ stands for sample $s$ of trace $\ell$ and $\mathcal{A}_k$ and $\mathcal{B}_k$ form a partition of the set of traces $\mathfrak{L}$ according to the value of the leaking bit of $Z_k$. Such distinguisher exploits the leakage of only one bit of the sensitive variable.

A more generic framework, brought by the stochastic attacks, allows to take into account more generic *a priori*. In such attacks, the adversary expresses the instantaneous leakage model $\varphi_{k^*}[s]$ as a parametric model whose parameters are regressed from the traces through a regression technique, still under a key assumption. A measure of fitness is then used as distinguisher.

**Correlation Power Analysis.** The most famous example is probably the Correlation Power Analysis (CPA) [BCO04] which assumes that there exist some parameters $\alpha, \beta \in \mathbb{R}$ such that:

$$\varphi_{k^*}[s](Z_{k^*}) = \alpha \cdot \mathrm{HW}(Z_{k^*}) + \beta \tag{1.15}$$

where $HW$ stands for the Hamming weight function counting the number of bits set to 1 in the variable. In such a case, the measure of fitness can directly be computed with the Pearson correlation coefficient:

$$\mathcal{D}_{\mathfrak{L}}^{(\mathrm{CPA})}[k] = \max_{s} \left[ \frac{\widehat{\mathrm{cov}}(L[s], \mathrm{HW}(Z_k))}{\sqrt{\widehat{\mathrm{Var}}(L[s]) \cdot \widehat{\mathrm{Var}}(\mathrm{HW}(Z_k))}} \right] \tag{1.16}$$

**Linear Regression Analysis.** Then, the Linear Regression Analysis [Dog+12], came to relax the assumption that all the bits of the sensitive variable should have the same contribution to the leakage. Instead, they may be weighted by real coefficients $(\alpha_0, \ldots, \alpha_n) \in \mathbb{R}$:

$$\varphi_{k^*}[s](Z_{k^*}) = \alpha_0 + \sum_{i}^{n} \alpha_i \cdot Z_{k^*}[i] \tag{1.17}$$

where $Z_{k^*}[i]$ represents the $i^{th}$ bit of $Z_{k^*}$. Such assumption traduces the fact that the instantaneous leakage model can be expressed as a polynomial of degree one. Thus, for all key hypothesis $k$, a linear regression can be performed to find such polynomial $p_k[s]$ of degree 1 minimizing the quadratic error $\mathbb{E}[(L - p_k(Z_k))^2]$. Then, the coefficient of determination ($R^2$) can be used as a distinguisher:

$$\mathcal{D}_{\mathfrak{L}}^{(\mathrm{LRA})}[k] = \max_{s} \left[ 1 - \frac{\widehat{\mathbb{E}}[(L[s] - p_k[s](Z_k))^2]}{\widehat{\mathrm{Var}}(L[s])} \right] \tag{1.18}$$

**Model-Based Attacks.** Another type of strategy, denoted the Model-Based Attack (MBA) in this thesis, partitions the set of traces into classes, collapsing traces that should have a similar leakage according to an *a priori* model $M$. Such partition is done under a key hypothesis $k$ and it corresponds to re-label the traces with $M(Z_k)$ instead of $Z_k$. Then a leakage assessment technique $S_L^{M(Z_k)}$ can be used as a distinguisher considering the new labels. If the model is sound the maximum leakage should be reached when the partition has been done with the correct key hypothesis.

$$\mathcal{D}_{\mathfrak{L}}^{(\mathrm{MBA})}[k] = \max_s \left[ S_L^{M(Z_k)}(s) \right] \tag{1.19}$$

The most classic example of such model-based attacks is probably the Mutual Information Analysis (MIA) [Gie+08] which uses the MI as leakage assessment technique:

$$\mathcal{D}_{\mathfrak{L}}^{(\mathrm{MIA})}[k] = \max_s \left[ \mathcal{I}(L[s],\ M(Z_k)) \right] \tag{1.20}$$

An in-depth analysis about this strategy and the optimal choice of the model is conducted in Chapter 3.

All the distinguishers discussed in this section have their strengths and weaknesses. It is not possible to rank them according to a single objective criteria, for example the number of required traces to rank the correct key at the first position. As shown in [Dog+12] a CPA performs better than a LRA if the true leakage is effectively a Hamming weigh model because it is simpler (with a lower capacity) which induces a higher distinguishability. For the same reason, if the leakage effectively comes from only one bit, the DPA may outperform the other attacks. However, in many practical cases the LRA would perform better because the bit are weighted with very different coefficients, (even sign inversions [CLH20]). The main advantage of the MBA is that they offer a way to work with any leakage assessment techniques. This means that any improvement of such techniques would automatically be translatable into an improvement of the corresponding attack (which is the outline of the two next chapters of this thesis).

Distinguishers discussed in this section all have different strengths, yet they all share the same drawback: the fact of being intrinsically unidimensional. The two inconvenients discussed in subsection 1.2.2 for the leakage assessment technique also apply to these distinguishers. The maximum along time samples could be replaced by some kind of combination of several samples (for example summing the scores over a window) but such a strategy would be impacted by non-informative samples and brings questions about the length of the window. In any case, the incentive of designing an intrinsically multidimensional distinguisher is high and the intuition behind this thesis is that research in this direction might be fruitful, especially with the global rise of deep learning techniques introduced in section 1.3.

### 1.2.4 Side-Channel Countermeasures

In order to counteract side-channel attacks, designers may implement strategies that aim at reducing the dependency between the leakage and the sensitive variables. These countermeasures are mainly divided into two categories: the hiding and the masking countermeasures which can be combined to increase security.

#### Hiding

The main principle of hiding countermeasures is to add perceived noise in the adversary's measurements without modifying the internal variables being processed. Many techniques have been proposed for that purpose. Most of them aim at randomizing the power consumption of the device by arbitrarily altering the processing time of the sensitive variables, making the traces desynchronized with respect to their interesting part. Such misalignment techniques reduce the dependency between the leakage and the secret and complexify the attacks, especially the ones based on univariate statistics. Software-level methods for randomizing the power consumption include the shuffling of independent operations [VC+12], the insertion of dummy instructions [CK09; CK10] or the code polymorphism [Bel+18b] which combines many interesting techniques. Hardware-level methods include instruction randomization through the use of non-deterministic processors [IPS02; MMS01] or the enhancement of a jittering effect *via* a clock with unstable frequency, or *via* an asynchronous logic style [Moo+02; Moo+04].

A common approach to defeat such countermeasures is to apply some kind of preprocessing to the traces before the attack. These include realignment through pattern matching [Nag+07], integration techniques [Man04; MOP10] or transformation of small parts of the traces into probability distribution as in the SCATTER strategy [Thi+18]. More recently, deep learning techniques came proposing a sound alternative to such preprocessing techniques (mainly for the supervised attacks) with promising results as discussed in subsection 1.3.2.

#### Masking

As opposed to hiding, masking countermeasures modify the internal variables of the algorithms using secret sharing techniques [Cha+99]. The idea is to split each sensitive intermediate variable $Z$, into $d$ shares: $(Z_i)_{1 \leq i \leq d}$. The $d-1$ shares $Z_2, ..., Z_d$ are randomly chosen and the last one, $Z_1$ is processed such that:

$$Z_1 = Z * Z_2 * \cdots * Z_d \tag{1.21}$$

for a group operation $*$ of $\mathcal{Z}$. The processing of $Z$ is replaced by the processing of all the shares which are only recombined at the end of the computation to give the correct ciphertext. This has the effect of complexifying the stochastic process

$\mathcal{S}$ generating $L$ from $Z$, rendering it no longer separable into a deterministic and a noise part. Assuming the masks are uniformly distributed, the knowledge of $d-1$ shares does not tell anything about $Z$ (this is why such masking is said to be of order $d-1$).

However, partial information on the $d$ shares can be exploited to retrieve information about $Z$. That is why, to defeat masking, one should use a distinguisher able to combine the leakage of at least $d$ samples of the traces (assuming shares do not leak at the same time). For supervised attacks, this corresponds to modeling multivariate probability densities which does not drastically change the shape of the attack even though it significantly increases the number of required traces. It is a bit trickier for unsupervised attacks. A common strategy is to combine a model-based attack with higher-order leakage assessment techniques discussed in section 1.2.2. However, such strategies bring questions about the optimal choice of the model (as discussed in Chapter 3) and about the combination function. Stochastic attacks may also be extended to deal with masked implementations [DDP13], although they suffer from limitations pointed out in Chapter 5.

## 1.3 Emergence of Deep Learning in SCA

### 1.3.1 General Concept of Deep Learning

Machine learning is a branch of computer science that groups all the techniques that aim at granting a computer the ability to learn from data. Deep Learning (DL) is a sub-part of machine learning based on artificial neural networks. Such networks are inspired by the processing of information in biological systems and aim at approximating a function solving an optimization problem. They consist of a succession of processing units called *neurons* organized in layers as represented in Figure 1.3. Input and output layers are specified by the function that should be approximated. Hidden layers are not specified which gives rise to many different architectures adapted by designers to the nature of the underlying problem. By analogy with the synapses in the brain, neurons share connections with the adjacent layers' neurons. The strength of such connections is materialized by weighting coefficients which are updated during the *training phase* of the network. The objective of the network is encoded in the so called *loss function* which should be minimized. Such a loss function takes as input the output of the network and may also take some additional data often called the *labels*. The training consists in applying a form of the gradient descent algorithm in order to minimize the loss function, computed with the available data. Each parameter is iteratively updated in the opposite direction of the loss function's gradient, which is estimated through the *backpropagation* algorithm [Kel60].

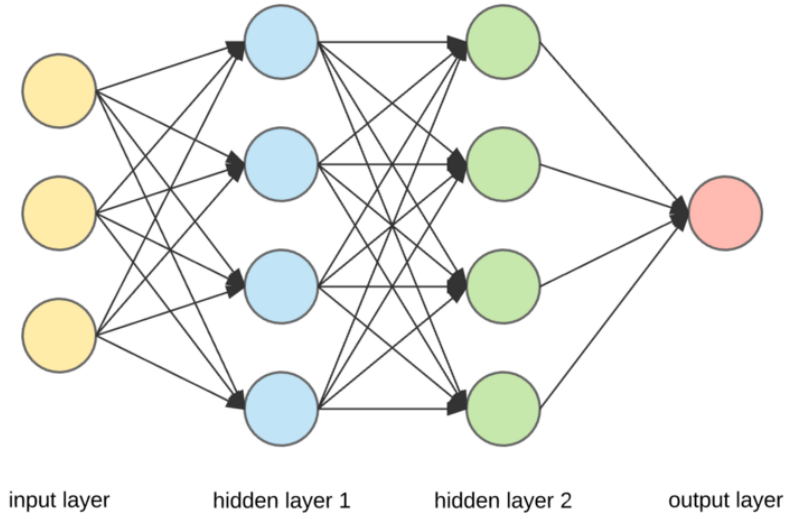



FIGURE 1.3: A simple neural network architecture

### 1.3.2 Deep Learning-based SCA

Deep learning has been introduced in SCA mainly in the context of supervised attacks. Indeed the estimation of a discriminative model is a classification task that fits perfectly with the DL paradigm. Given a set of training traces $(\ell_i)_{1\leq i\leq N} \in \mathbb{R}^p$ associated with their correct labels: $(z_i)_{1\leq i\leq N} \in \mathcal{Z}$, the task of the network is to estimate the probability density: $P(Z \mid L)$, or in other words, a function $f : \mathbb{R}^p \to [0,\ 1]^{|\mathcal{Z}|}$ which, for a given trace, outputs a probability distribution for the values of $Z$. The network is a function $F_\theta : \mathbb{R}^p \to [0,\ 1]^{|\mathcal{Z}|}$ parameterized by some trainable parameters $\theta \in \Theta$. The training process corresponds to minimizing over $\Theta$ a well-chosen loss function $\mathcal{L}(\theta)$. There exist several such loss functions, for example, [MDP19] showed that the negative log-likelihood is sound in an SCA context:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} -log_2(F_\theta(\ell_i)[z_i]) \tag{1.22}$$

The main advantage of using DL over more classical hand-crafted methods is the flexibility of neural networks. Indeed, it has been shown in the universal approximation theorem [HSW89] that even a one (hidden) layer Multi Layer Perceptron (MLP) can approximate any given function with an arbitrary precision (provided that the layer size is not bounded). This means that neural networks can deal, in theory, with big parts of the traces, automatically learning how to combine multiple samples if necessary, while reducing the need for preprocessing techniques. The initial work from Maghrebi *et al.* [MPP16] introduced them to SCA in order to deal with potential non-Gaussian distributions, especially in the context of masking and showed they were a sound alternative to Gaussian template attacks in practice.

The advantage of DL techniques is not only theoretical. Indeed, DL has been a very active field for this last few years and many works focus on turning the

good theoretical properties of neural networks into practical results, for example, by providing new architectures or optimizations in the learning process. Such improvements in the DL techniques directly impact the DL-based SCA. For example, [CDP17] showed that the Convolutional Neural Networks (CNN), originally developed for computer vision due to their shift-invariant properties, can be applied to counter misalignment in side-channel traces.

Many works (not always reproducible...) have followed essentially in the field of supervised attacks. However, such a classification method could very well be turned into a leakage assessment technique. Indeed, a network predicting the $Z$ variable from $L$ better than random indicates that a leakage occurs in $L$ with respect to $Z$. This is essentially what is done in [MWM21] except that they fix only 2 classes for $Z$ and try to predict them. Since any leakage assessment technique can be turned into an unsupervised model-based attack, such DL-based leakage assessment have their corresponding unsupervised attack introduced in [Tim19] (one may notice the inconsistency in the date of publication of these two papers). The main problem of [Tim19] is that it does not discuss the choice of the model. Indeed, it is hard to analytically reason from classification neural network outputs and their analysis is mostly empirical. This is why one of the core ideas of this thesis is to bridge DL techniques to some more tractable concepts from the information theory in order to ease the theoretical reasoning.

## 1.4 Information Theory

The entropy $\mathcal{H}(X)$ [Sha48] of a random variable is a fundamental quantity in information theory that indicates how much information one would gain, in average, by learning a particular realization $x$ of $X$. It is defined as the expectation of the self-information $log_2(1/p_X)$. In a discrete context:

$$\mathcal{H}(X) = \sum_{x \in \mathcal{X}} P_X(x) \cdot log_2(\frac{1}{P_X(x)}) \tag{1.23}$$

In a side-channel context, $X$ could be replaced by $L$, the leakage variable. However, one is not interested in the absolute information provided by $L$ but rather in the amount of information revealed about a second variable such as a sensitive variable $Z$. This is exactly what is measured by the mutual information $\mathcal{I}(Z, L)$. It is defined as:

$$\mathcal{I}(Z, L) = \mathcal{H}(Z) - \mathcal{H}(Z \mid L) = \mathcal{H}(L) - \mathcal{H}(L \mid Z) \tag{1.24}$$

where $\mathcal{H}(A \mid B)$ stands for the conditional entropy of $A$ knowing $B$:

$$\mathcal{H}(A \mid B) = \sum_{b \in \mathcal{B}} P_B(b) \cdot \mathcal{H}(A \mid B = b) \tag{1.25}$$

Another useful way to characterize $\mathcal{I}(Z, L)$ is to express it as the Kullback-Leibler (KL) divergence between the joint distribution and the product of the marginals:

$$\begin{aligned} \mathcal{I}(Z, L) &= D_{KL}(P_{Z,L} \,||\, P_Z \otimes P_L) \\ &= \sum_{z \in \mathcal{Z}} \sum_{l \in \mathcal{L}} P(z, l) \cdot log\left(\frac{P(z, l)}{P(z) \cdot P(l)}\right) \end{aligned} \tag{1.26}$$

## 1.5 Thesis Motivations and Outline

As shown throughout this introductory chapter, many techniques have been developed to extract and exploit the dependency between a sensitive variable and a leakage variable. This diversity makes it hard for designers and evaluators to draw an objective metric in order to assess leakage. From an information theory point of view, the maximum amount of information one could extract from a side-channel trace is bounded by the mutual information, $\mathcal{I}(Z, L)$ between the sensitive variable $Z$ and the trace $L$. This quantity is, indeed, central in the side-channel domain. The goals of the different actors could be summarized as follows:

- **Designers** aim at implementing countermeasures to decrease as far as possible $\mathcal{I}(Z, L)$, with computational, spatial and efficiency constraints.
- **Evaluators** aim at estimating $\mathcal{I}(Z, L)$ as closely as possible to assess leakages in a worst-case scenario.
- **Attackers** aim at developing strategies to partially or fully exploit $\mathcal{I}(Z, L)$ in order to recover a secret.

The main problem of such a unified paradigm is that $\mathcal{I}(Z, L)$ is known to be hard to estimate from drawn samples when the variables live in a high dimensional space, which is generally the case of $L$. Indeed, computing $\mathcal{I}(Z, L)$ usually requires an estimation of the conditional density $\Pr(Z|L)$ which is hard because of the well-known curse of dimensionality. This explains why conventional leakage assessment tools and classical attack strategies typically focus on one (or a few) samples at a time in the trace. As a result, the amount of information effectively detected or used may be significantly lower than $\mathcal{I}(Z, L)$.

In a completely unrelated context, Belghazi *et al.* [Bel+18a] lately introduced a Mutual Information Neural Estimator (MINE) which uses the power of deep learning to compute mutual information in high dimension. They have proposed applications in a pure machine learning context but we argue that this tool might be of great interest in the side-channel domain. Indeed, as discussed in subsection 1.3.2, deep learning techniques seem to be promising and well-suited for SCA. Being able to efficiently compute $\mathcal{I}(Z, L)$ in an unsupervised way (no profiling of the target needed), no matter the target, the implementation, or the countermeasures used, would be highly relevant for all the different parties.

The analysis of MINE in a side-channel context is the first contribution of this thesis and is discussed in Chapter 2. We show that it constitutes a new multidimensional leakage assessment tool that outputs an interpretable number representing the upper bound of information one could potentially exploit from the target.

However, knowing how much information is leaking is not the same as knowing how to exploit it to retrieve a secret. If such an upper bound may be approached in the context of supervised attacks, it is much less straightforward for unsupervised attacks. Since MINE constitutes a new leakage assessment technique, we naturally derive it as an unsupervised model-based attack, the Neural Estimation Mutual Information Analysis (NEMIA) in Chapter 3. It revisits the classical MIA, mitigating some common misconceptions about its genericity with respect to the adversary's leakage model *a priori*, and conducting an in-depth analysis related to the optimal choice of the model in a multidimensional context.

While NEMIA bridges the good properties of mutual information and the power of the latest DL techniques, it appears that it has two major practical drawbacks: the time complexity (it requires as many network trainings as there are key candidates) and the need for an explicit model which requires a strong *a priori*. In Chapter 4, we derive from the mathematical framework developed for NEMIA, a new deep learning architecture, denoted the EVIL machine, able to automatically recover the leakage model of a device. Such a machine is then turned into an unsupervised attack: the EVIL Machine Attack (EMA). It combines the good properties of the stochastic attacks (flexibility on the *a priori*) and the potential of DL techniques while requiring only one network training, thus overcoming the two main issues of NEMIA and of DL model-based attack in general.

Eventually, the analysis of EMA in the context of masked implementations raised questions about higher-order generalizations of stochastic attacks. Some inconsistencies related to the genericity of the state-of-the-art generalizations with respect to the masking scheme are pointed out in Chapter 5. We then propose a new unsupervised strategy, the Joint Moment Regression (JMR), which aims at being generic regarding the masking scheme, while still benefiting from the flexibility of the stochastic attacks. As a final thought, we highlight the fact that EMA and JMR might be combined to form a deep learning-based unsupervised attack agnostic to the underlying masking scheme and requiring only one network training.

To ease the reading of this thesis and make the chapters standalone, there may be small overlaps between their introductory parts as we reintroduce notations that are useful for the understanding of the aforesaid chapters.

# Chapter 2

# Leakage Assessment through Neural Estimation of the Mutual Information

*"Truth is multi-dimensional."*

Peter Shepherd

*A large variety of side-channel attacks have been developed to extract secrets from electronic devices through their physical leakages. Whatever the utilized strategy, the amount of information one could gain from a side-channel trace is always bounded by the Mutual Information (MI) between the secret and the trace. This makes it, a key quantity for leakage evaluation. Unfortunately, traces are usually of too high dimension for existing statistical estimators to stay sound when computing the MI over full traces. However, recent works from the machine learning community have shown that it is possible to evaluate the MI in high dimensional space thanks to new deep learning techniques. This chapter explores how this new estimator could impact the side channel domain. It presents an analysis whose aim is to derive the best way of using this estimator in practice. Then, it shows how such a tool can be used to assess the leakage of any device.*

## 2.1 Introduction

### 2.1.1 Context

As explained in the introductory chapter, Side-Channel Analysis (SCA) is defined as the set of techniques aiming at gaining information on a secret, owned by a system, through an auxiliary leakage channel often related to its physical implementation. The secret is usually a cryptographic key but could be as well basic block execution, assembly instructions, or even the value of an arbitrary register. The basic assumption is that the secret and the side-channel data are statistically dependent. Many techniques have been developed to exploit part of this dependency and mount an attack but the incentive to develop a leakage assessment protocol can be found in several works [MSQ08; SM15; Bro+19]. From a purely theoretical point of view, the best metric to assess the quantity of leakage would be the mutual information $\mathcal{I}(Z, L)$ between the sensitive variable and the full trace as it measures the dependency between these two variables, by definition. However, such a metric has not really been used in the state-of-the-art. Indeed, it is famously hard to estimate due to the so called curse of dimensionality. Traces being usually of too high dimension, the majority of the leakage assessment techniques work in a univariate way, computing sample-wise statistics from the trace, as pointed out in subsection 1.2.2. Thus, they may underestimate the leakage or even state that there is no leakage when there is.

On the other hand, latest deep learning techniques have proved to be a very interesting tool for SCA since neural networks are able to automatically combine information from many samples of the traces. In addition, Belghazi *et al.* [Bel+18a] lately introduced a Mutual Information Neural Estimator (MINE) which uses deep learning techniques to produce an estimation of mutual information between high dimensional variables. MINE has been developed in a completely unrelated context, but this chapter proposes to analyze its applicability in the side-channel domain. The goal is to derive the best way to use MINE, in order to develop a new leakage assessment tool able to directly compute $\mathcal{I}(Z, L)$.

### 2.1.2 Chapter Organization

The general method and the mathematical ideas behind MINE are presented in section 2.2. Section 2.3 proposes an in-depth analysis of MINE in a side-channel context supported with synthetic traces, highlights the problem of overfitting and suggests ways of dealing with it. Section 2.4 provides some real case applications. It shows how this estimator constitutes a reliable leakage assessment tool that can be used to compare leakage from different implementations and devices. This section also shows that such an estimator can be used as a guide for an evaluator/attacker to maximize the MI captured from different hardware side-channel setups.

## 2.2 Mutual Information Neural Estimation

Mutual information is a powerful tool in data science since it measures the dependencies between two variables. In our case, it may be seen as an upper bound on the amount of information an attacker could gain on the secret from the target's leakage. The main problem is that side-channel data are often of too high dimension for classical estimators to stay sound when computing the MI with the full trace. The most common ways to estimate MI are the histogram method and the kernel density estimation both described in [PR09]. There also exists a non parametric estimation based on $k$-nearest neighbors [KSG04]. This chapter is interested in MINE [Bel+18a], a new estimator based on deep learning techniques, which claims to scale well with high dimensions. Technical details about MINE are given hereafter.

A well known property of $\mathcal{I}(Z, L)$ is its equivalence with the Kullback-Leibler (KL) divergence between the joint probability $p_{Z,X} = \Pr(Z, L)$ and the product of the marginals $p_Z \otimes p_L = \Pr(Z) \cdot \Pr(L)$:

$$\mathcal{I}(Z, L) = D_{KL}(p_{Z,L} \,||\, p_Z \otimes p_L) \tag{2.1}$$

where $D_{KL}(p, q)$ is defined as follow:

$$D_{KL}(p \,||\, q) = \mathbb{E}_p\left[log\left(\frac{p}{q}\right)\right] \tag{2.2}$$

whenever $p$ is absolutely continuous with respect to $q$. This property guarantees that when $q$ is equal to 0, $p$ is also equal to 0 and there is no division by 0 in the logarithm. By definition, $p_{Z,L}$ is absolutely continuous with respect to $p_Z \otimes p_L$.

The key technical ingredient of MINE is to express the KL-divergence with variational representations, especially the Donsker-Varadhan representation that is given hereafter. Let $p$ and $q$ be two densities over a compact set $\Omega \in \mathbb{R}^d$.

**Theorem 1.** *(Donsker-Varadhan, 1983) The KL-divergence admits the following dual representation:*

$$D_{KL}(p \,||\, q) = \sup_{T:\, \Omega \to \mathbb{R}} \mathbb{E}_p[T] - log(\mathbb{E}_q[e^T]) \tag{2.3}$$

*where the supremum is taken over all functions $T$ such that the two expectations are finite.*

A straightforward consequence of this theorem is that for any set $\mathcal{F}$ of functions $T : \Omega \to \mathbb{R}$ satisfying the integrability constraint of the theorem we have the following lower bound:

$$D_{KL}(p \,||\, q) \geq \sup_{T \in \mathcal{F}} \mathbb{E}_p[T] - log(\mathbb{E}_q[e^T]) \tag{2.4}$$

Thus, using Equation 2.1, one have the following lower bound for $\mathcal{I}(Z, L)$:

$$\mathcal{I}(Z, L) \geq \sup_{T \in \mathcal{F}} \mathbb{E}_{p_{Z,L}}[T] - log(\mathbb{E}_{p_Z \otimes p_L)}[e^T]) \tag{2.5}$$

**How to compute $\mathcal{I}(Z, L)$.** To put it short, the idea is to define $\mathcal{F}$ as the set of all functions $T_\theta$ parametrized by a neural network with parameters $\theta \in \Theta$ and to look for the parameters maximizing the loss function $\mathcal{L} : \Theta \to \mathbb{R}$:

$$\mathcal{L}(\theta) = \mathbb{E}_{p_{Z,L}}[T_\theta] - log(\mathbb{E}_{p_Z \otimes p_L)}[e^{T_\theta}]) \tag{2.6}$$

This loss function is itself bounded by $\mathcal{I}(Z, L)$. The universal approximation theorem for neural networks guarantees that this bound can be made arbitrarily tight for some well-chosen parameters $\theta$. The goal is then to find the best $\theta$, potentially using all the deep learning techniques and, more generally, all the tools for optimization problem-solving. The expectations in Equation 2.6 can be estimated using empirical samples from $p_{Z,L}$ and $p_Z \otimes p_L$ and the maximization can be done with the classical gradient ascent. A noticeable difference with a classical deep learning setup is that the trained network is not used for any kind of prediction. Instead, the evaluation of the loss function at the end of the training gives an estimation of $\mathcal{I}(Z, L)$. We give hereafter the formal definition of the estimator as stated in the original paper [Bel+18a].

**Definition 1.** *(MINE) Let $\mathcal{A} = \{(z_1, \ell_1), \ldots, (z_n, \ell_n)\}$ and $\mathcal{B} = \{(\widetilde{z}_1, \widetilde{\ell}_1), \ldots, (\widetilde{z}_n, \widetilde{\ell}_n)\}$ be two sets of n empirical samples respectively from $p_{Z,L}$ and $p_Z \otimes p_L$. Let $\mathcal{F} = \{T_\theta\}_{\theta \in \Theta}$ be the set of functions parametrized by a neural network. MINE is defined as follows:*

$$\widehat{\mathcal{I}(Z, L)}_n = \sup_{T \in \mathcal{F}} \overline{\mathbb{E}_{\mathcal{A}}[T]} - log(\overline{\mathbb{E}_{\mathcal{B}}[e^T]}) \tag{2.7}$$

*where $\overline{\mathbb{E}_{\mathcal{S}}[\cdot]}$ stands for the expectation empirically estimated over the set $\mathcal{S}$.*

The main theoretical result proved in [Bel+18a] is that MINE is strongly consistent.

**Theorem 2.** *(Strong consistency) For all $\epsilon > 0$ there exist a positive integer N such that:*

$$\forall n > N, \quad |\mathcal{I}(Z, L) - \widehat{\mathcal{I}(Z, L)}_n| < \epsilon, \text{ a.e.} \tag{2.8}$$

In practice, one often only have a set of samples from the joint distribution: $\mathcal{A} = \{(z_1, \ell_1), \ldots (z_n, \ell_n)\}$. Samples from the product of the marginals can be artificially generated by shuffling the variable $X$ using a random permutation $\sigma$: $\mathcal{B} = \{(z_1, \ell_{\sigma(1)}), \ldots, (z_n, \ell_{\sigma(n)})\}$. We provide a pseudo-code implementation of MINE, in algorithm 1, that uses minibatch gradient ascent. Note that $\mathcal{B}$ is regenerated after each epoch. Thus, this algorithm is not strictly implementing MINE as defined in Equation 2.7 because $\mathcal{B}$ is fixed in this definition. Theoretical arguments are provided in subsection 2.3.4 to explain why this regeneration limit overfitting in practice and is therefore mandatory.

**Algorithm 1:** Mine implementation

**Input:** $\mathcal{A} = \{(z_1, \ell_1), \dots (z_n, \ell_n)\}$
$\theta \leftarrow$ Initialize network parameters
Choose $b$ a batch size such that $b \mid n$
**repeat**
  Generate $\mathcal{B} = \{(z_1, \ell_{\sigma(1)}), \dots, (z_n, \ell_{\sigma(n)})\}$ with a random permutation $\sigma$
  Divide $\mathcal{A}$ and $\mathcal{B}$ into $\frac{n}{b}$ packs of $b$ elements: $A_1, \dots, A_{\frac{n}{b}}$ and $B_1, \dots, B_{\frac{n}{b}}$
  **for** $i = 1; i = \frac{n}{b}$ **do**
    $\mathcal{L}(\theta) \leftarrow \overline{\mathbb{E}_{A_i}[T_\theta]} - log(\overline{\mathbb{E}_{B_i}[e^{T_\theta}]})$ `// Evaluate the loss function`

    $G(\theta) \leftarrow \nabla_\theta \mathcal{L}(\theta)$ `// Compute the gradient`

    $\theta \leftarrow \theta + \mu G(\theta)$ `// Update the network parameters (`$\mu$ `is the learning rate)`
  **end**
**until** *convergence of* $\mathcal{L}(\theta)$

## 2.3 Analysis of MINE in a Side-Channel Context

MI has found applications in a wide range of disciplines and it is not surprising that it is also of great interest for side-channel analysis. Unlike Pearson correlation coefficient, it detects non-linear dependencies and thus does not require any assumptions on the leakage model. Another key property of the MI is that it is invariant to bijective transformations of the variables. This is of interest for side-channel as $Z$ usually represents the state of an internal variable (ex: $Z = k^* \oplus X$ for an AES) and is therefore unknown but bijectively related to a public variable $X$. In that case, there exists a bijective function $g_{k^*}$ such that $Z = g_{k^*}(X)$ and:

$$\mathcal{I}(Z, L) = \mathcal{I}(g_{k^*}^{-1}(Z), L) = \mathcal{I}(X, L) \tag{2.9}$$

Thus, one may estimate $\mathcal{I}(Z, L)$ with only the knowledge of $X$ and $L$ and therefore quickly get the amount of leakage an attacker could potentially exploit.

In what follows, we consider that we are granted $n$ samples $(z_1, \ell_1), \dots, (z_n, \ell_n)$ of traces associated to the sensitive variable being processed in the device (or as stated above, to any bijection of this variable). These samples will be either generated on simulation or measured from real case experiments. The goal is to derive the best way to use MINE in a side-channel context in order to compute a reliable estimation of $\mathcal{I}(Z, L)$.

### 2.3.1 Simulated Traces Environment

In order to assess the capabilities of MINE experiments on synthetic traces were first conducted. These traces have been generated using a leakage model which may seem awkward since the whole point of conducting MI analysis is to avoid any assumption on the leakage model. But we argue that as a first step, it brings

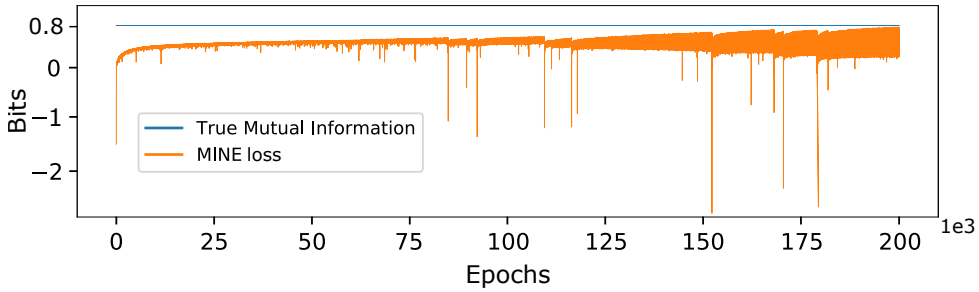



FIGURE 2.1: Evolution of MINE's loss function over time

a valuable advantage: the environment is perfectly controlled, thus the true MI is known and can be used to evaluate the results and compare different settings.

**Trace generation.** To generate traces, featuring $n_l + n_r$ independent samples, a sensitive byte $0 \leq s \leq 255$ was first drawn uniformly. The leakage was spread over the $n_l$ samples drawn from a normal distribution centered in the Hamming Weight (HW) of $s$ and with noise $\sigma \sim \mathcal{N}(\mathrm{HW}(z), \sigma)$. The $n_r$ remaining samples are random points added to the trace to artificially increase the dimension and be closer from a real scenario. Each of the $n_r$ samples is drawn from a normal distribution centered in $c$ and with noise $\sigma \sim \mathcal{N}(c, \sigma)$, where $c$ is an integer itself drawn uniformly between 0 and 8. A very simple yet informative case is to set $n_l = 1, n_r = 0$ and $\sigma = 1$. In that case, the true mutual information $\mathcal{I}(Z, L)$ is equal to:

$$\begin{aligned}
\mathcal{I}(Z,L) &= H(Z) - H(Z|L) \\
&= 8 - \sum_{z=0}^{255} \int_{-\infty}^{\infty} \Pr(z,l) \cdot log_2\left(\frac{1}{\Pr(z|l)}\right) dl \\
&= 8 - \sum_{z=0}^{255} \int_{-\infty}^{\infty} \frac{1}{2^8} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(l-\mathrm{HW}(z))^2} \cdot log_2\left(\frac{\sum_{z'=0}^{255} e^{-\frac{1}{2}(l-\mathrm{HW}(z'))^2}}{e^{-\frac{1}{2}(l-\mathrm{HW}(z))^2}}\right) dl \\
&\approx 0.8 \text{ bits}
\end{aligned} \tag{2.10}$$

As a first step, we applied MINE to a set of 10k synthetic traces generated with these parameters. The network was set to be a simple Multi Layer Perceptron (MLP) with two hidden layers of size 20. The Exponential Linear Unit (ELU) was used as the activation function. The input layer was composed of two neurons, representing the value of the sensitive variable $Z$ and the one-dimensional trace $L$. The output was a single neuron giving the value of the function $T_\theta$. The batch size was set to 500. The value of the loss function $\mathcal{L}(\theta)$ over time is plotted in Figure 2.1. An epoch represents the processing of all the data so the parameters are updated 20 times per epoch.

As shown, the results are mixed. On one hand, the loss function is always under the true MI and it seems that the limit superior of MINE is converging over time towards 0.8, *i.e.* the true MI. On the other hand, the loss function experiences a lot of drops and the convergence is very slow (above 200k epochs). Drops may be due to the optimizer used (ADAM [KB14]) and happens when the gradient is very close

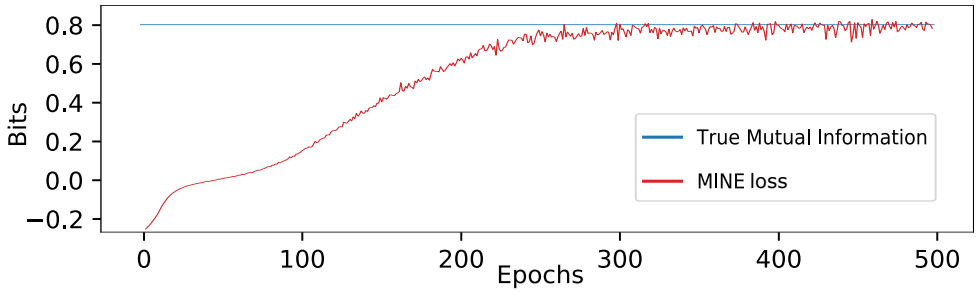



FIGURE 2.2: MINE with input decompression

to 0. Increasing the size/number of hidden layers did not produce any significantly better results. In that state MINE is clearly not of any use for side-channel: the convergence is not clear and a classical histogram method would compute the MI faster and better for one-dimensional traces.

### 2.3.2 Input Decompression

Trying to gain intuition about the reasons causing the network to perform poorly in this situation, we hypothesized that the information in the first layer, especially the value of $z$, could be too condensed in the sense that only one neuron is used to describe it. Intuitively, the information provided by $z$ about the corresponding trace $\ell$ is not continuous in $z$. The meaning of this statement is that there is no reason that two close values of $z$ induce two close values of $l$. For example, in a noise-free Hamming weight leakage model, the traces corresponding to a sensitive value of 127 and 128 would be very different since $\mathrm{HW}(127) = 7$ and $\mathrm{HW}(128) = 1$. Since neural networks are built using a succession of linear and activation functions which are all continuous, approximating functions with quick and high variations may be harder for them. Indeed, building a neural classifier that extracts the Hamming weight of an integer is not an easy task. However, if the value of this integer is split into multiple neurons holdings its binary representation, the problem becomes trivial as it ends up being a simple sum.

This observation led us to increase the input size to represent the value of $z$ in its binary form, thus using 8 neurons. However, computing $\mathcal{I}(Z, L)$ in that case gives an unfair advantage to the arbitrarily chosen Hamming weight model. Indeed, the value of $L$ would be closely related to the sum of the input bits. So we decided to compute $\mathcal{I}(Z \oplus k, L)$ instead of $\mathcal{I}(Z, L)$, with a fixed $k$. As stated above this bijective transformation does not change the MI anyway and removes a possible confusion factor in the analysis. Results with the same parameters as before ($n_l = 1, n_r = 0$, $\sigma = 1$) are presented in Figure 2.2. They bear no comparison with the previous ones. With this simple trick, MINE quickly converges toward the true MI. The estimation seems robust as restarting the training from different initializations of the network always produces the same results. The order of magnitude of the computational time for the 500 epochs is around two minutes.

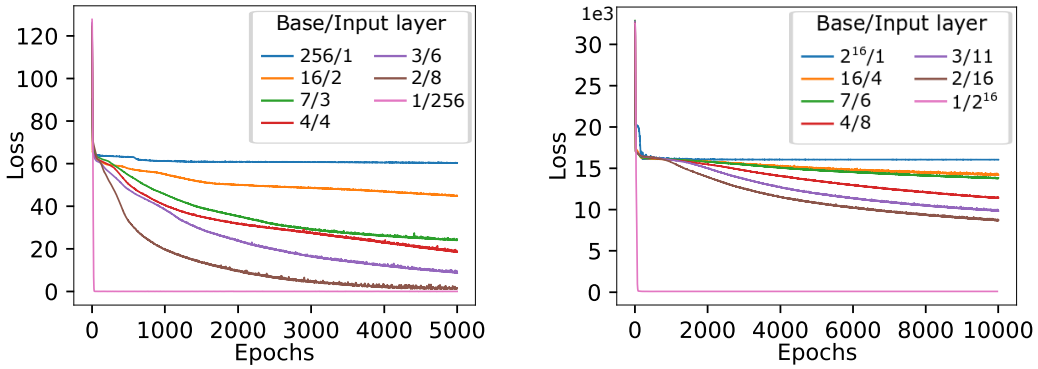
**Remark 2.** *Note that any constant function $T_\theta$ would produce a loss function of 0. We argue that this could explain the knee point around 0 bit in the learning curve: it is indeed, easy for the network to quickly reach 0 tuning the parameters towards a constant function, before learning anything interesting.*

Before testing this method for higher dimension traces, we propose to analyze more in-depth this Input Decompression (ID). The goal is to understand if this result was related in any way to our simulation setup or if ID could be applied to more generic cases. As a first step, we tried to decompress $L$ instead of $Z$, binning the value of $L$ to the closest integers, then using its binary representation as input neurons. As expected, it did not work: results looked like Figure 2.1, as $L$ is by essence a continuous variable. If there is no interest in splitting into multiple neurons an intrinsically continuous variable, our hypothesis is that, for categorical variables, the greater the decompression, the faster the training.

#### Learning Random Permutations

In order to test this hypothesis in a more generic case, this section proposes to build a neural network $P_\theta$ which goal is to learn a random permutation $P$ of $\{0, \dots, n-1\}$ and to analyze its performance in terms of ID. Permutations have been chosen because they are arbitrary functions with no relation between close inputs. For an integer $m < n$ the network returns a float $P_\theta(m)$ which has to be as close as possible to $P(m)$. The loss function was defined as error: $\mathcal{L}(\theta) = |P_\theta(m) - P(m)|$. Network architecture was again a simple MLP but with 3 hidden layers of size 100. To study the effect of ID the input layer was defined to be the representation of $m$ in different bases, with one neuron per digit. For example, with n = 256, all bases in $256, 16, 7, 4, 3, 2, 1$ were considered resulting in a first layer of respectively $\{1, 2, 3, 4, 6, 8, 256\}$ neurons (base 1 is actually the One Hot Encoding (OHE)). The training dataset was a list of 10k integers uniformly drawn from $\{0, \dots, n-1\}$. Loss functions in terms of ID are depicted in Figure 2.3. At the end of the training, plots are exactly ordered in the expected way: greater decompression leads to faster and better training.

In a recent analysis, Bronchain *et al.* [BS19] have shown that it was hard for a MLP to learn the Galois multiplication in $GF(2^n)$ when $n \geq 8$. As Galois multiplication suffers from the same non-continuity than random permutations, we argue that blue plots confirm this result. With no ID our MLP did not show the beginning of a convergence towards 0. But we do think their network may have been successful with ID. Pink plots show that the best choice is to use the OHE. The problem with OHE is that the number of neurons (and therefore the computational time) scales linearly with $n$ (the number of categories of the underlying problem), where it only scales logarithmically with any other base. In side-channel, one mostly deals with bytes (256 categories) and will therefore use the OHE. However, subsection 2.4.3




(A) $n = 2^8$ (B) $n = 2^{16}$

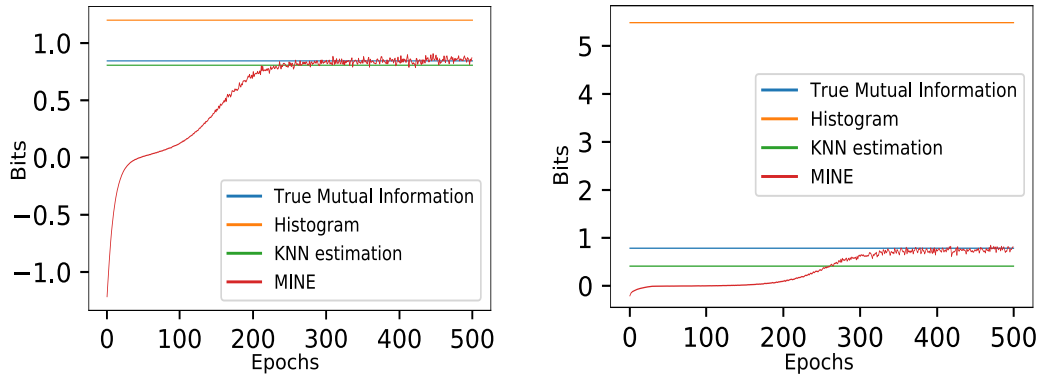
FIGURE 2.3: Impact of input decompression on learning random permutations

presents a scenario where using base 2 is a better choice, when computing the MI with assembly instructions.

**Remark 3.** *Note that the constant function $P_\theta = \frac{n}{2}$ would result in an average loss function of $\frac{n}{4}$, which explains the knee point around $\frac{n}{4}$ observable in most of the curves: quickly converging towards this function is an efficient strategy for the network to minimize its loss at the beginning of the learning phase. We verified this statement by looking at the predictions of the network which were all close from $\frac{n}{2}$ in the early stages of the training.*

### 2.3.3 MINE in Higher Dimension

This section presents results of simulations in higher dimension and compare MINE estimation to that provided by the classical histogram and KNN methods. The histogram estimator has been implemented following the description from [PR09] and Steeg's implementation [Ste14] has been utilized for KNN. Network architecture described in subsection 2.3.1 has been used with OHE to encode the $S$ variable. We have kept $n_l = \sigma = 1$ so the true MI is still around 0.8 bits but $n_r$ was no longer set to 0 in order to increase the traces dimension. Figure 2.4 shows the results for $n_r = 1$ and $n_r = 9$. In both case MINE correctly converges toward the true MI. The histogram method tends to overestimate the MI (as explained in [VEB10]) while KNN method underestimates it. With dimension greater than 10 these methods are not reliable anymore. One could argue that any dimension reduction technique applied in the above experiments would allow to compute the MI with classical estimators. While this is true in this case it may result in a loss of information in a real case scenario where the information could be split into multiple samples of the traces. We have conducted many experiments with different parameters and MINE always returned reliable estimations even in very high dimension (ex: Figure 2.5b with $n_l = 5$ and $n_r = 1000$).




(A) Trace dimension $= 2$ $(n_l = 1, n_r = 1, \sigma = 1)$

(B) Trace dimension $= 10$ $(n_l = 1, n_r = 9, \sigma = 1)$

FIGURE 2.4: Comparison of MINE with classical estimators in higher dimension

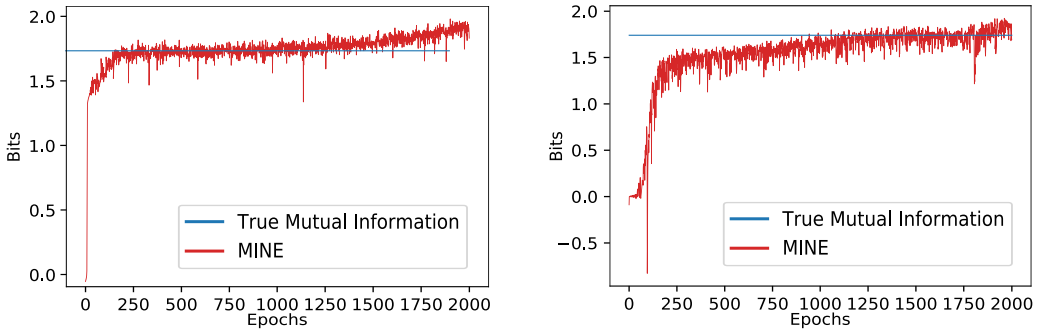
### 2.3.4 Analysis of the Overfitting Problem

Results of simulations are encouraging as they seem accurate with a lot of different parameters but one problem still has to be solved before testing MINE on real traces: when to stop the training ? Until now, training has been manually stopped when the loss had converged towards the true MI. No such threshold value will be granted in real cases. One could argue that since the loss function is theoretically bounded by the true MI, a good strategy would be to let the training happen during a sufficiently long time and to retain the supremum of the loss function as the MI estimation. We argue that this strategy is not viable: in practice the bound does not hold as expectations in the loss are not the true expectations but are only estimated through empirical data. Thus, MINE can still produce output above the true MI. Figure 2.5 shows this phenomenon: training has been intentionally let running for a longer time in these experiments and MINE overestimates the MI at the end of the training. In other terms, MINE is no exception to the rule when it comes to the overfitting problem: the network can learn ways to exploit specificities of the data it is using to train, in order to maximize its loss function. We propose hereafter a detailed analysis of this problem and answer to the following question: is it possible to control (for example to bound with a certain probability) the error made by the network ?

Let us return to the definition of MINE estimator:

$$\widehat{\mathcal{I}(Z,L)}_n = \sup_{\theta \in \Theta} \overline{\mathbb{E}_{\mathcal{A}}[T_\theta]} - log(\overline{\mathbb{E}_{\mathcal{B}}[e^{T_\theta}]}) \tag{2.11}$$

The problem comes from the fact that the two expectations are estimated over the set of empirical data $\mathcal{A}$ and $\mathcal{B}$. The error can not be controlled in the classical way with the central limit theorem because there is a notion of order that is important: the two sets $\mathcal{A}$ and $\mathcal{B}$ are selected *before* the network tries to find the supremum over $\Theta$. Thus, the network can exploit specificities of $\mathcal{A}$ and $\mathcal{B}$ in its research. We show in Theorem 3 that given two sets $\mathcal{A}$ and $\mathcal{B}$, the supremum may not even be bounded.




(A) Trace dimension = 15 $(n_l = 5, n_r = 10, \sigma = 1)$

(B) Trace dimension = 1005 $(n_l = 5, n_r = 1000, \sigma = 1)$

FIGURE 2.5: Over estimation of MINE at the end the training (overfitting)

**Theorem 3.** *Let $X$, $Y$ be two random variables over $\Omega$. Let $\vec{x} = (x_1, \ldots, x_n) \in \Omega^n$ and $\vec{y} = (y_1, \ldots, y_n) \in \Omega^n$ be two samples of $n$ realizations of respectively $X$ and $Y$. Then,*

$$\sup_{T:\Omega\to\mathbb{R}} \overline{\mathbb{E}_{\vec{x}}[T(X)]} - log(\overline{\mathbb{E}_{\vec{y}}[e^{T(Y)}]}) < \infty \Leftrightarrow \forall i, \exists j \text{ such that } x_i = y_j$$

*Proof.* Let us introduce two new random variables, $X'$ and $Y'$ defined as follows:

$$\forall \omega \in \Omega,\ \mathbb{P}(X' = \omega) = \frac{1}{n} \cdot |\{x_i = \omega\}| \quad \text{and} \quad \mathbb{P}(Y' = \omega) = \frac{1}{n} \cdot |\{y_i = \omega\}|$$

The samples $\vec{x}$ and $\vec{y}$ are perfect samples of $X'$ and $Y'$ (by definition of $X'$ and $Y'$), thus, the estimated expectations are equal to the true expectations computed over this new variables:

$$\sup_{T:\Omega\to\mathbb{R}} \overline{\mathbb{E}_{\vec{x}}[T(X)]} - log(\overline{\mathbb{E}_{\vec{y}}[e^{T(Y)}]}) = \sup_{T:\Omega\to\mathbb{R}} \mathbb{E}_{X'}[T(X')] - log(\mathbb{E}_{Y'}[e^{T(Y')}])$$

Now let us assume the right part of the equivalence. This condition means that there is no isolated $x_i$, or in other words: $\forall \omega, \Pr(Y' = \omega) = 0 \Rightarrow \Pr(X' = \omega) = 0$. This guarantees the absolute continuity of $p_{X'}$ with respect to $p_{Y'}$ and thus, that $D_{KL}(p_{X'} || p_{Y'})$ exists. Therefore, using Theorem 1:

$$\sup_{T:\Omega\to\mathbb{R}} \overline{\mathbb{E}_{\vec{x}}[T(X)]} - log(\overline{\mathbb{E}_{\vec{y}}[e^{T(Y)}]}) = D_{KL}(p_{X'} || p_{Y'}) < \infty$$

If, on the other hand, this condition is false: $\exists i$ such that $\forall j$, $x_i \neq y_j$. For any given function $T$ one can exploit this isolated $x_i$ modifying $T(x_i)$ without influencing the second expectation. In particular, if $T(x_i)$ tends towards infinity:

$$\begin{aligned}\lim_{T(x_i)\to\infty}\left[\overline{\mathbb{E}_{\vec{x}}[T(X)]} - log(\overline{\mathbb{E}_{\vec{y}}[e^{T(Y)}]})\right] &= \lim_{T(x_i)\to\infty}\left[\frac{1}{n}\sum_{k=1}^{n} x_k T(x_k) - log(\frac{1}{n}\sum_{k=1}^{n} y_k e^{T(y_k)})\right] \\ &= \lim_{T(x_i)\to\infty}\left[\frac{1}{n}\sum_{k=1}^{n} x_k T(x_k)\right] - log(\frac{1}{n}\sum_{k=1}^{n} y_k e^{T(y_k)}) \\ &= \infty\end{aligned}$$

So in that case:

$$\sup_{T:\,\Omega\to\mathbb{R}} \overline{\mathbb{E}_{\vec{x}}[T(X)]} - log(\overline{\mathbb{E}_{\vec{y}}[e^{T(Y)}]}) = \infty$$

□

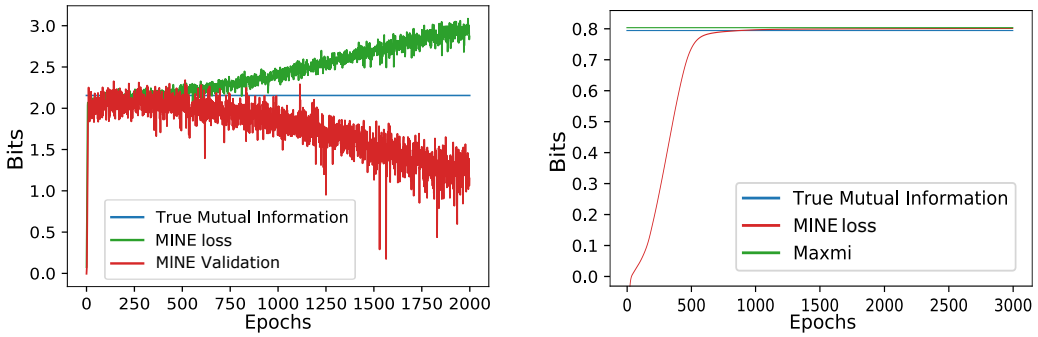
This theorem means that most of the time (and especially for high dimensional variables) the supremum is infinite and MINE is not even well defined. The natural question that comes now is: why does MINE seem to work in practice?

We claim that this is due to the implementation and especially to the randomization of the set $\mathcal{B}$ evoked in section 2.2: after each epoch a new permutation $\sigma$ is drawn to generate samples from $p_Z \otimes p_L$: $\mathcal{B} = \{(z_1, \ell_{\sigma(1)}), \ldots, (z_n, \ell_{\sigma(n)})\}$. Thus, the isolated samples from $\mathcal{A}$ are not always the same at each epoch which does not leave time for the network to exploit them. To verify that this was a key element, MINE was run without this randomization process. The loss function diverged towards infinity, as predicted by Theorem 3.

In the long run, the network can still learn statistical specificities of the dataset such as samples from $\mathcal{A}$ that has a greater probability of being isolated, and exploit them. This explains why MINE may overfit when it has a long time to train. That is why we suggest to add a validation loss function.

**Validation Loss Function**

A validation loss function is a common tool when it comes to detect overfitting and stop the training at the right time. The idea is to split the dataset $\mathcal{A}$ into a training dataset $\mathcal{A}_t$ and a validation one $\mathcal{A}_v$ and to only use $A_t$ for the training. At the end of each epoch, the loss function is computed both on $A_t$ and $A_v$. As the data from $A_v$ are never used during training, MINE cannot overfit on them. Thus, it is safe to take the supremum of the loss computed over $A_v$ as our MI estimation. It also provides a useful condition to stop the training as the decrease of the validation loss function is usually a sign of overfitting. Figure 2.6a shows an example where the training loss function and the validation one (computed on 80% and 20% of the data) respectively increase and decrease after a while. Training could have been stopped after the $500^{th}$ epoch.




(A) Validitaion strategy $(n_l = 10, n_r = 0, \sigma = 1)$

(B) Fill the holes strategy $(n_l = 1, n_r = 0, \sigma = 1)$

FIGURE 2.6: Two possible strategies against overfitting

### Fill the Holes

Theorem 3 states that there is still a case where the supremum is bounded: when $\forall a \in \mathcal{A}$, $\exists b \in \mathcal{B}$ such that $a = b$, or in other words, when there is no isolated value in $\mathcal{A}$. An alternative solution to prevent overfitting is thus to force this condition to be true instead of regenerating $\mathcal{B}$ after each epoch. Naively filling the holes by adding to $\mathcal{B}$ all the isolated values is not a good idea because the resulting set would be biased, not containing *stricto sensu* samples drawn from $p_Z \otimes p_L$. However, with $\mathcal{A} = \{(z_1, \ell_1), \dots (z_n, \ell_n)\}$, $\mathcal{A}_Z = \{z_1, \dots, \ell_n\}$ and $\mathcal{A}_L = \{z_1, \dots, \ell_n\}$ one can define $\mathcal{B}'$ as the Cartesian product[1] $\mathcal{B}' = \mathcal{A}_Z \times \mathcal{A}_L$ which is by definition a non-biased dataset that covers all the elements of $\mathcal{A}$. The problem is that its size is no longer $n$ but $n^2$ which drastically impacts the computational time of MINE as the network has to compute $T_\theta(b)$ for all $b \in \mathcal{B}'$ at each epoch. However, the number of network evaluations can be reduced to $c \cdot n$ where $c$ is the cardinality of the set $\mathcal{Z}$ made up of all the possible values taken by the sensitive variable $Z$. For example, if $S$ is a byte, $c = 256$. The idea is to evaluate $T_\theta$ on the $c \cdot n$ elements of the set $\mathcal{Z} \times \mathcal{A}_L$ which is sufficient to cover all the couples from $\mathcal{B}'$ as elements from $\mathcal{A}_Z \times \mathcal{A}_L$ can all be found in $\mathcal{Z} \times \mathcal{A}_L$.

With this implementation MINE is fundamentally bounded by a quantity denoted *Maxmi* equal to the KL-divergence between the empirical distributions associated to $\mathcal{A}$ and $\mathcal{B}$ as stated in the proof of Theorem 3. Figure 2.6b shows an example of MINE with this implementation applied to the already considered case $(n_l = 1, n_r = 0, \sigma = 1)$. One may observe that the loss function is a lot smoother and is effectively bounded by *Maxmi* (we tried to let the network train for more than 100k epochs) which is another empirical confirmation of Theorem 3.

However, when the dimension of the variables increases samples tend to be more and more unique. At the limit, they hold the full information about the corresponding sensitive variable $z$ which means that *Maxmi* will tends towards $H(Z)$. Knowing if the network will always converge towards his supremum or will stabilize to a

[1] These sets are actually multisets as they may contains repetitions of a single elements but the Cartesian product can be canonicaly extended to multisets.

value close to the true MI is an open question. We do think that the randomization proposed in the precedent strategy may help to that aim and that is why we will stick to the validation method for our real-life experiments, which is faster anyway.

## 2.4 Application of MINE in an Evaluation Context

This section provides real case examples where MINE could be useful especially in an evaluation context. Its most straightforward utilization is probably to assess the quantity of information leaking from a device when it computes a cryptographic algorithm. It can be seen as a first security metric, easy to compute whatever the target and the implementation, with low expertise required. However, MINE only returns an upper bound on the amount of leakage potentially usable. In practice, an attacker may not be able to fully exploit this information, depending on her strategy, and that is why classical evaluation methods still have to be performed.

That being said, MINE is also a great comparison tool. Indeed, its output is an interpretable number that allows to objectively rank different devices or implementations in terms of their leakage. It can be used to analyze the effect of a countermeasure or even to compare different hardware setup in order to maximize the MI for future attacks or evaluations.

### 2.4.1 Leakage Evaluation of an Unprotected AES

As a first real case experiment, our target was an unprotected AES implemented on a cortex M4 device. 20k EM traces centered on the first round of the AES have been acquired through a Langer probe (RF-B 0,3-3) linked to an low noise amplifier and a Tektronix oscilloscope (MSO64, 2.5 GHz) with a sample rate of 1 GS/s. Resulting traces had a length of 50k samples. They have been labeled with the first byte of the corresponding plaintext which was drawn randomly for each computation of the AES.

The main goal of this first experiment was to demonstrate how adding more samples to the analysis, which is the purpose of MINE, increases the amount of information one can recover. To this end, only the $n$ samples with the maximum SNR were kept in the traces, with $n$ in $\{1, 5, 500\}$. Network architecture was the same as in the simulated experiments. Results are presented in Figure 2.7. The thick blue plot shows that if one only uses one sample in his analysis (for example with a CPA or histogram-based MIA) he would be able to extract at most 1.15 bits about the secret, per trace. While it is a huge amount of information (it is an unprotected AES) it is possible to extract almost 4 times more using 500 samples. For clarity reasons, only the validation loss has been plotted. Training has been stopped after epoch 500 as these validations (especially the green one) started to decrease. Going further with $n \geq 500$ did not produce better results as it seems that the remaining samples were absolutely not informative about the secret.

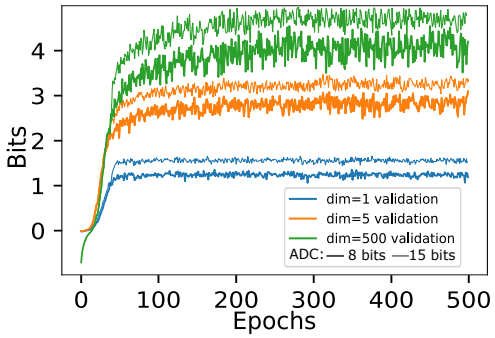



FIGURE 2.7: Leakage evaluation of an unprotected AES

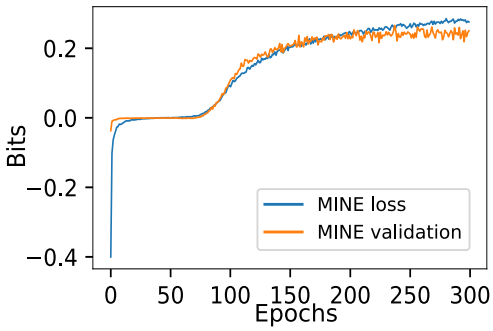



FIGURE 2.8: Leakage evaluation of a masked AES (ASCAD)

#### ADC Comparison

The oscilloscope used in the former experiment offers the possibility to set the ADC precision to either 8 or 15 bits. This is a good opportunity to show MINE comparative interest and its ability to answer questions such as "Is it really worth it to buy the newest scope with the enhanced ADC precision?" in a quantitative and objective way. 10k traces instead of 20k (so that the occupied memory stayed constant) were thus acquired with the 15 bits precision. Results are represented by the thin plots on Figure 2.7. In this case, the answer is that there is a slight improvement (around 10%) working with the 15 bits precision rather than the 8 bits one.

### 2.4.2 Leakage Evaluation of a Masked AES from the ASCAD Database

One of the main difficulties of side-channel analysis is to extract information even when the target algorithm has been masked. Indeed, masking removes all the first-order leakage and thus, obliges one to combine samples together to detect a dependency with the secret. This is usually very long as all the couples of samples (or $n$-tuple) have to be tested.

It is thus a great challenge for MINE to see if it is able to automatically perform this recombination, and detect higher-order leakages. For that purpose, the public dataset ASCAD [Ben+18] (with no jitter) has been used. It provides a database of 50k EM traces of 700 samples each, of an AES protected with a Boolean masking. A MI estimation, derived from deep learning attack results, has already been done on this dataset [MDP19]. They reported a MI of 0.065 bit between the traces and the third key byte (the first two were not masked) which provides a reference point.

At first, MINE was not successful: the loss function increased but the validation started to decrease very early which is a direct sign of overfitting. Intuitively, when the underlying problem is more complex, it may be easier for the network to learn properties of the empirical data before the true structure of these data. Then, classical solutions against overfitting have been applied to MINE. These include Batch Normalization (BN) layers, dropout, and regularization techniques. While the last, did not impact performances significantly, the combination of BN and dropout greatly

improved the results. A BN layer has been applied to the inputs in order to normalize them. This is known to make the loss function smoother and thus the optimization easier [San+18]. Dropout was activated with $p = 0.2$ so that each neuron has a probability $p$ of being set to 0 when an output of the network is computed (except when the validation is computed). This is also known to reduce overfitting and make the training more robust [Sri+14].

Results are presented in Figure 2.8: validation loss reached a value of 0.2 bits which is about three times bigger than the MI reported in [MDP19]. Due to how validation is computed this value cannot be an overestimation of the MI. Our MLP structure may be a little more adapted than the CNN used in [MDP19] as there is no jitter in that case. We also suggest that the input decompression technique, only usable with MINE, could help the network to learn, especially for complex problems such as when the algorithm is masked. This may explain why MINE was able to extract more information in that case. One can observe that it took 100 epochs for the network to start to learn something. It may seem random but this period of 100 epochs was surprisingly repeatable across experiments.

### 2.4.3 Instructions Leakage

Another advantage of MINE is that it cannot only compute MI for high dimensional traces but also for secrets with a high number of classes. This is the case if one is interested in recovering information about the raw assembly instructions that are being executed. This branch of SCA is called Side Channel Based Disassembling (SCBD) [GP08; EPW10; Str+15; CLH20] and the main difficulty in this domain is the size of the attacked variable, generally the couple (opcode, operands) which is no longer a simple byte. For example the target device in [EPW10; Str+15; CLH20] is a PIC16F from Microchip which encodes its instruction on 14 bits. Even though some opcodes are not valid, the number of possible couples (opcode, operands) is around $2^{12}$. It is even worse for more complex processors encoding their instruction on 16 or 32 bits. MINE treats the attacked variable as an input and the number of neurons used to encode it can be adjusted with ID as stated in subsection 2.3.2. Using base 2, one only need 14 neurons to encode an instruction in the PIC example.

In order to test MINE in this context, we have generated a program with 12k randoms instructions for the PIC. Using the same experimental setup described in section 4.2 of [CLH20], an EM trace of the whole execution has been acquired (it was averaged on 500 traces as the program is repeatable). This trace has then been separated into 12k sub-traces of 2000 samples each. Each sub-trace was labeled with the executed instruction. As it has been shown in [CLH20] that the probe position may be very important, MINE has been applied at 100 different positions (using a $(10 \times 10)$ grid) resulting in the MI cartography given in Figure 2.9. The value at each position is the mean of the network's validation computed over the 100 last epochs of the training (all the training lasted 500 epochs and a Gaussian filter has

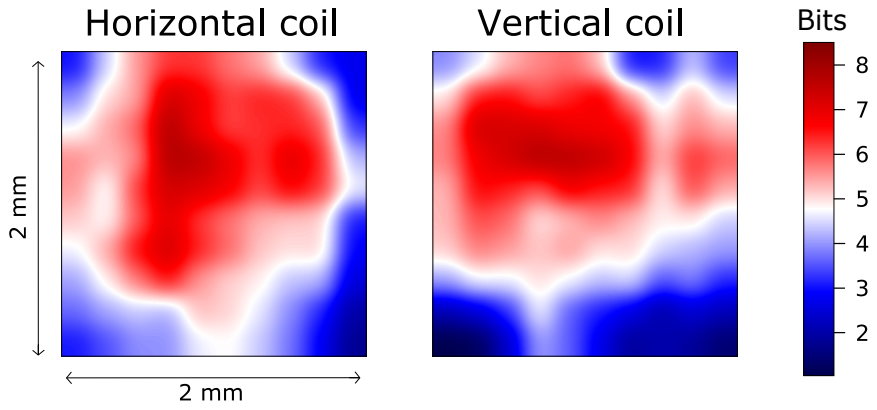



FIGURE 2.9: Cartography of the MI between instructions and traces estimated by MINE on a PIC16F

been applied to the figure). Up to 8 bits of information have been found for the best positions which is a high amount if one compares to the full entropy of an instruction which is approximately 12 bits. This shows that MINE stays sound even when the target variable has a high number of classes.

#### Coil Comparison

Similar to what has been done regarding the selection of the oscilloscope precision, another hardware comparative experiment was conducted. Two probes with two different coil orientations (Langer ICR HH and HV 100-27) have been used. While the "hot" zones are globally the same, one may observe that the coil orientation may have a significant impact on the captured information for some specific positions. This experiment suggests that MINE could be used to guide the positioning of EM probes during evaluations.

## 2.5 Conclusion

This chapter suggests ways MINE, a new deep learning technique to estimate mutual information, could constitute a new tool for side-channel analysis. The main advantage is its ability to estimate MI between high dimensional variables. Indeed, being able to consider full (or large part of) traces as a variable, allows to assess all potential leakage sources with no *a priori* on the leakage model neither on the implementation. It seems that MINE can be used as a very simple tool to obtain an objective leakage evaluation from traces. Thus, it may be employed for massive and quick evaluations for designers in their development process as well as for evaluators as a first leakage metric.

These suggestions result from a theoretical and practical analysis of MINE in a side-channel context. MINE's overfitting problem has been deeply investigated as well as the way input representation may have a big impact on performances. The next natural challenge, which is the object of the next chapter is to investigate

possible usages of MINE for extracting secrets in an unsupervised way *i.e.* in an attack context.

# Chapter 3

# Revisiting Mutual Information Analysis

## Multidimensionality, Neural Estimation and Optimality Proofs

*"Information is not knowledge."*

Albert Einstein

*The preceding chapter showed how Mutual Information Neural Estimation (MINE) could be applied to side-channel analysis in order to evaluate the amount of leakage of an electronic device. One of the main advantages of MINE over classical estimation techniques is to enable the computation between high dimensional traces and a secret, which is relevant for leakage assessment. However, optimally exploiting this information in an attack context in order to retrieve a secret remains a non-trivial task especially when a profiling phase of the target is not allowed. Within this context, the purpose of this chapter is to address this problem based on a simple idea: there are multiple leakage sources in side-channel traces and optimal attacks should necessarily exploit most/all of them. To this aim, a new mathematical framework, designed to bridge classical Mutual Information Analysis (MIA) and the multidimensional aspect of neural-based estimators, is proposed. One of the goals is to provide rigorous proofs consolidating the mathematical basis behind MIA, thus alleviating inconsistencies found in the state of the art. This framework allows to derive a new attack called Neural Estimated Mutual Information Analysis (NEMIA). To the best of our knowledge, it is the first unsupervised attack able to benefit from both the power of deep learning techniques and the valuable theoretical properties of MI. From simulations and experiments conducted in this paper, it seems that NEMIA performs better than classical and more recent deep learning based unsupervised side-channel attacks, especially in low-information contexts.*

## 3.1 Introduction

### 3.1.1 Context

The previous chapter took advantage of a new deep learning technique called Mutual Information Neural Estimation (MINE) [Bel+18a] to develop a side-channel tool able to reliably estimate the MI between the secret and full traces. It drastically reduces the impact of high dimensionality on the estimation reliability compared to classical estimators. This tool allows one to get an absolute leakage quantification from raw traces which is helpful for designers or evaluators to perform leakage assessment. However, knowing how much information is potentially usable is different from actually exploit it. This chapter analyzes the impact of this tool from the attacker's point of view. Is an adversary also able to use the inherent multidimensional properties of MINE to exploit at the same time all the potential leakage sources ? And if so, what is the optimal way to do it ? This chapter aims at answering these questions.

Side-channel attacks are mainly divided into two categories: supervised SCA, where the adversary can first perform a characterization of the target, and unsupervised SCA in which this profiling step is not possible. For profiled SCA, one is theoretically able to exploit all the information $\mathcal{I}(Z, L)$ by perfectly learning the target's leakage model during the characterization phase. Deep learning attacks have been shown to effectively extract all the available information when using the negative log likelihood as loss function [MDP19]. Therefore the problem is closed, at least in theory, for profiled SCA.

However, this is not the case for unsupervised attacks, where the true leakage model of the target is unknown to the adversary. In this situation, only a fraction of $\mathcal{I}(Z, L)$, which value depends on the correctness of one's *a priori* on the leakage model, can be exploited. For example, the Correlation Power Analysis (CPA) [BCO04] is efficient for linear dependencies between the leakage and a certain function of the intermediate variable (often being the Hamming weight function). The Linear Regression Analysis (LRA) [Dog+12] also assumes a linear dependency but can handle different weights for each bit of the intermediate variable.

Mutual Information Analysis (MIA), however, has been introduced as a generic strategy able to capture any kind of dependencies. Papers addressing the theoretical background behind MIA [Gie+08; PR09; VCS09; Bat+11] all present MIA as SCA distinguisher able to recover the correct key without any knowledge on the target nor on its leakage model. However, this leakage model free strategy only works to target non-bijective intermediate variables which makes it well suited for the DES (as the DES S-boxes are not bijectives) but less suited for more recent algorithms such as the AES. This explains why MIA has not often been used in practice.

A second version of the MIA allowing to target any intermediate variables (and is therefore applicable in many more contexts) has also been developed. These two versions are not separated in the literature but we decided to do so in this paper to clarify the relationship between MIA and leakage model *a priori*. Indeed, this second version is not leakage model free *i.e.* it requires an *a priori* on the leakage model to work. However, one of the main advantages of this attack is that it is not limited to linear leakage model and more generally, does not require any assumption on the leakage distribution (as long as the adversary's *a priori* is sufficiently correct). However, this gain in genericity comes at the cost of efficiency: CPA has always been proved to work better than MIA in classical attack scenarios since leakage models are often linear. Therefore MIA is more seen as a great tool in theory that does not offer much in practice.

However, one of the main advantages of MIA is that it generalizes well to higher dimension variables and offers a way to potentially use a bigger part of the information contained in a side-channel trace. This has not really been used in the literature (except to extend MIA for masked implementation [PR09; Bat+11]) due to MI estimators limitations. But recent breakthroughs regarding neural estimation encourages to revisit classical MIA in order to make it highly multidimensional, to get closer to an optimal attack regarding the amount of information being used from the traces.

Even if neural estimation techniques can be applied in the leakage model free version of the MIA, we are more interested in the second version of MIA since it does not impose restrictions on the targeted algorithm. However, we argue that the mathematical framework behind this version (developed in [Gie+08; PR09; VCS09; Bat+11]) is not complete and rely too much on intuition. As a result, it is difficult to derive the best way to use the new MI estimators, especially in the context of high dimensional variables, where intuition quickly falls short. That is why rebuilding a mathematical framework along with rigorous proofs on how to conduct an optimal multidimensional MIA is one of the contributions of this chapter.

### 3.1.2 Contributions

1. Clarifying the State Of The Art (SOTA) around the MIA.

   *We explicitly split MIA into two different versions (this is not explicitly done in the SOTA), to help understanding the need or not of an a priori on the leakage model (subsection 3.2.2). We then highlight inconsistencies with the second version mainly related to the fact that MIA relies on a distinguisher computing a score for each key hypotheses, but the wrong hypotheses scores are not taken into account in the analysis (subsection 3.2.3). This leads us to define a new generic version of MIA which objective is related to a maximization problem that includes the impact of the wrong hypotheses scores (subsection 3.2.4).*

2. Providing proofs to analytically solve the mathematical problems emerging from our new version of MIA.

   *One of the main contributions of this chapter, given by Theorem 4 (subsection 3.2.5), is to solve the optimization problem defined in subsection 3.2.4. Then, Theorem 5 provides an extension of the analysis in the context of masking (section 3.3). Both theorems are designed to take into account the potential multidimensionality of the leakage and therefore are suited to support the use of the new neural MI estimators.*

3. Presenting a new unsupervised multidimensional attack: the Neural Estimated Mutual Information Analysis (NEMIA).

   *Mathematical results are then combined with recent breakthroughs regarding neural MI estimation in high dimension. This allows to derive, to the best of our knowledge, the first unsupervised side-channel attack able to benefit from both deep learning techniques (highly multidimensional, no pre-processing of the data...) and the valuable theoretical properties of MI (section 3.4).*

4. Providing Simulations and experiments to support the analysis.

   *Simulations are provided both to empirically validate the mathematical analysis as well as to gain intuition about their meaning and about which situations are best suited for the use of NEMIA (section 3.5). Eventually, practical experiments on the ASCAD database (both on raw traces and on artificially noised traces) are conducted and show that this new attack seems outperform classical SCA strategies in terms of number of traces needed and noise resiliency (section 3.6).*

## 3.2 Mutual Information Analysis

### 3.2.1 Unsupervised attacks

Suppose an adversary wants to recover the secret key used by the physical implementation of a cryptographic algorithm. He has access to a set of traces $(L_i)_{1\leq i\leq n}$ labeled with a public variable $X_i$ used for the encryption/decryption. The general idea of an unsupervised side-channel attack is to make a series of hypotheses $k_i$, on a sub-part of the key and to use a distinguisher $\mathcal{D}(k)$ allowing to rank the different candidates. Distinguishers use statistical dependencies between traces and an intermediate variable $Z_{k^*} = g(X, k^*)$ that depends on a public variable $X$ and the correct key $k^*$ through a deterministic function $g : \mathcal{X} \times \mathcal{K} \to \mathcal{Z}$ related to the underlying algorithm. For simplicity, $g(X, k)$ is denoted $g_k(X)$ in the rest of the chapter.

Common distinguishers such as Pearson's coefficient or coefficient of determination in a linear regression exploit some *a priori* on the leakage model. A common intuition about mutual information used as a distinguisher [Gie+08] is that it has been introduced precisely to reduce the need to have an *a priori*. It is often found in the literature (*e.g.* [Bat+11]) that it aims at generality, leading to successful attacks

without requiring specific knowledge or assumptions on the target. While this is true in some sense, this assertion is mitigated hereafter.

### 3.2.2 State of the Art

This section presents the state of the art of MIA [Gie+08; PR09; VCS09; Bat+11] and is organized to discuss and clarify the importance of the adversary's leakage model *a priori*.

MIA uses a distinguisher $\mathcal{D}$ which takes the following form[1]:

$$\mathcal{D}(k) = \mathcal{I}\big(f(Z_k), L\big) \tag{3.1}$$

with $f$ being a function transforming the guessed intermediate variables $Z_k$. This function is one of the main concerns of this chapter. It is often called the "model" of the adversary. The requirement of a model may seem contradictory with the claims of genericity of the MIA. Actually, this model can be replaced by the identity function making the MIA independent of any *a priori* on the leakage model. This version of the MIA is presented hereafter.

**MIA Version 1 (Leakage model *a priori* free)**

In its most basic form, MIA uses $\mathcal{I}(Z_k = g_k(X), L)$ as a distinguisher, making hypotheses on $k$. With $\varphi : \mathcal{Z} \to \mathbb{R}^n$ representing the leakage model of the target, $L$ can be written as $L = \varphi(Z_{k^*}) + N$, with $N$ being a random variable independent of $Z_k$ for all $k$, and representing the noise. With these notations, the distinguisher becomes:

$$\mathcal{D}(k) = \mathcal{I}\big(Z_k, \varphi(Z_{k^*}) + N\big) \tag{3.2}$$

**Proposition 1.** *This distinguisher is maximized for the correct key hypothesis $k^*$.*

*Proof.* Using Equation 1.24, for any $k \in \mathcal{K}$:

$$\begin{aligned} \mathcal{D}(k^*) - \mathcal{D}(k) &= \mathcal{H}(L) - \mathcal{H}(L \mid Z_{k^*}) - \big[\mathcal{H}(L) - \mathcal{H}(L \mid Z_k)\big] \\ &= \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid Z_k\big) - \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid Z_{k^*}\big) \end{aligned} \tag{3.3}$$

Since adding knowledge can only decrease entropy:

$$\mathcal{D}(k^*) - \mathcal{D}(k) \geq \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid Z_k, Z_{k^*}\big) - \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid Z_{k^*}\big) \tag{3.4}$$

[1] Due to MI estimator limitations, $\mathcal{D}(k)$ is often replaced in practice by $\max_i \mathcal{I}(f(Z_k), L[i])$, where $L[i]$ represents the $i$-th sample of the trace. This does not affect the theory described in this section so we decided to keep it as described in Equation 3.1 for the sake of simplicity. More details are provided in subsection 3.4.1.

Now using the independence of $N$ and the fact that $\varphi(Z_{k^*})$ is entirely determined by $Z_{k^*}$:

$$\begin{aligned} \mathcal{D}(k^*) - \mathcal{D}(k) &\geq \mathcal{H}(N) - \mathcal{H}(N) \\ &\geq 0 \end{aligned} \tag{3.5}$$

which concludes the proof. □

This strategy does not require any assumption on the leakage model of the target. However, it only works if the correct key hypothesis is distinguishable from the other ones, or, in other words, if $\mathcal{D}(k) < \mathcal{D}(k^*), \forall k \neq k^*$, which is not guaranteed by Proposition 1. An important property of the MI is that it is preserved by injective transformations of the input variables. So if different key hypotheses yield $Z_k$ variables differing from each other only by a permutation (for example if the $g_k$ functions are bijective), $\mathcal{I}(Z_k, L)$ would be constant for all $k$ and the distinguisher could not discriminate key candidates. Therefore, $g_k$ has to be non-injective. For example, one could target the output of the first round DES S-box.

While this form of MIA is effectively leakage model free, it comes with a huge constraint since in many interesting cases $g_k$ is bijective. In the AES case, this means that one cannot target the output of the first S-box since $Sbox[k^* \oplus P]$ is bijective with $P$. In [PR09] and [RGV14b], authors suggest to target the bitwise addition between two S-box outputs during the *MixColumns* operation. This requires making hypotheses on 16 bits of the key (leading to $2^{16}$ MI computations). Moreover, it is only feasible if this operation leaks enough information which may not be the case in practice. Indeed, for hardware implementations, this step is usually fully combinatorial and does not use any register. This explains why most of the MIA experiments in the literature have been performed on the DES.

**MIA Version 2 (Leakage model *a priori* dependent)**

It is still possible to target $Z_{k^*} = g_{k^*}(P)$ for bijective $g_k$ functions. The idea is to apply a non-injective function $f$ to $Z_k$ and use $\mathcal{I}(f(Z_k), L)$ as distinguisher. The application of $f$ create a partition of $\mathcal{Z}$ so $f$ will be called the "partition function" in the rest of this thesis. Since no data transformation can create information (this is the so called data processing inequality [BR12]), the application of $f$ inevitably decreases the initial information: $\forall f, \forall k$, $\mathcal{I}(f(Z_k), L) \leq \mathcal{I}(Z_k, L)$. The goal is then to find a function that decreases more $\mathcal{I}(Z_k, L)$ than $\mathcal{I}(Z_{k^*}, L)$ and therefore, enhance the discriminating power of the analysis.

For example, assuming that bits leak independently, [Gie+08] proposes to drop one bit of $Z$. This is equivalent to redefine the intermediate variable as a restrictive number of bits of $g_{k^*}(P)$, and apply MIA version 1 with no partition function. Another idea is to use a guessed version $\bar{\varphi}$ of the leakage model $\varphi$. Indeed, $\mathcal{I}\big(\varphi(Z_k), \varphi(Z_{k^*}) + N\big)$ is clearly maximized for $k = k^*$. Therefore, if $\bar{\varphi}$ is not too far from $\varphi$, $\mathcal{I}\big(\bar{\varphi}(Z_k), \varphi(Z_{k^*}) + N\big)$ may still be maximized for $k = k^*$. It is shown

in [VCS09] that error in the approximation of $\varphi$ may be less penalizing than for other attacks.

In addition, MIA is more flexible in the sense that it is not limited to exploit linear dependencies and gives an option to mount a successful attack with any leakage model. However, it should be emphasized that, for this version, the adversary must have a good enough *a priori* on the leakage, otherwise, the attack is unsuccessful. A suitable choice for the partition function necessarily uses assumptions on $\varphi$.

While we think this point needed to be clarified, we do not see this as a criticism of MIA. As stated in[WOS14], hopes of finding a leakage model free strategy able to target a bijective intermediate variable are vain, even outside the context of MIA. We present hereafter a synthetic proof of the main result of [WOS14].

**Proposition 2.** *Let $g_k$ be a bijective map for all $k$. For any strategy $\mathfrak{S}$ which takes as input a set of traces $\vec{L} = \left(\varphi(g_{k^*}(P_i))\right)_{1\leq i\leq n}$ and outputs a ranking of the different key hypotheses, there exists a leakage model $\tilde{\varphi}$ that would rank $k^*$ in the last position such that the attack completely fails.*

*Proof.* First, apply $\mathfrak{S}$ on traces obtained through any leakage model $\varphi_0$ and denote by $\bar{k}$ the last key returned by $\mathfrak{S}$. Now, for all $P$, define $\tilde{\varphi}_0(g_{k^*}(P)) = \varphi_0(g_{\bar{k}}(P))$, which completely defines $\tilde{\varphi}_0$ as $g_{\bar{k}}$ is bijective. Applying $\mathfrak{S}$ on traces obtained through $\tilde{\varphi}_0$ would now rank $k^*$ in the last position. □

This proposition shows that there does not exist any generic distinguisher, that would both:

1) Exploit statistical dependencies between traces and an intermediate variable bijectively related to the plaintext.

2) Work whatever the leakage model of the target.

Since MIA version 2, with a fixed partition function, verifies 1), it necessarily fails for some leakage models or, in other words, has to use an assumption on the leakage model to succeed. Even though it requires an analysis on what partition function should be used, the rest of this chapter is more focused on MIA version 2 since it is more generic in the sense that it can be applied in many more situations.

### 3.2.3 About the Distinguishability

As stated in [WO11], even if $\mathcal{D}(k)$ is maximized for $k = k^*$, it is not enough to guarantee a successful attack in practice, when noise comes into play. What is really important is the difference between $\mathcal{D}(k^*)$ and the others, or in other words, the distinguishability of the correct hypothesis through the distinguisher $\mathcal{D}$. The idea found in the literature is that for a wrong key hypothesis:

«false predictions will form a partition corresponding to a random sampling of [L] and therefore simply give scaled images of the global side-channel probability

density function. Hence, the estimated mutual information will be equal (or close) to zero in this case.» [Bat+11].

We do not agree with this fact since the wrong hypotheses scores totally depend on the partition function $f$ and on $g_k$. As explained in the previous section, if the $g_k$'s are bijective, all the scores would be equal if $f$ is also bijective. This fact is well noted in all the papers about MIA but we would like to emphasize that even for non-bijective $f$ the wrong hypotheses score depends on the "degree of bijectiveness" of $f$. Intuitively, the more compact $f$ is (in the sense of more collisions through $f$) the smaller the wrong hypotheses scores would be. But the same is true for the correct score which means that there is a trade-off between how much one wants to decrease $\mathcal{I}\big(f(Z_k), L\big)$ for the wrong $k$ and keep $\mathcal{I}\big(f(Z_{k^*}), L\big)$ high, to enhance the distinguishability.

### 3.2.4 Towards an Optimal Partition Function

In the SOTA, the partition function is not seen as a parameter on which a maximization research could be done. Therefore, no research on finding the optimal function $f$ has been conducted. It is generally fixed to one or two constant choices, except in [PR09] where authors proposed that $f$ could be a generic function. However, it is stated that the adversary:

«does not need a good linear approximation of $\varphi$ but only a function [$f$] such that the mutual information $\mathcal{I}\big(f(Z_{k^*}), \varphi(Z_{k^*})\big)$ is non-negligible » [PR09].

Again, this condition is necessary but not sufficient. Even if bijective functions are excluded one can create the following $f_0$ function such that:

$$f_0(x) = \begin{cases} 0, \text{ if } x \in \{0, 1\} \\ x, \text{ else} \end{cases} \tag{3.6}$$

Being almost the identity function, $f_0$ is such that $\mathcal{I}\big(f_0(Z_{k^*}), \varphi(Z_{k^*})\big)$ is high but would have a very low discriminating power. This shows that the wrong hypotheses scores cannot be left out of the analysis. One typically wants to find the $f$ function maximizing the distinguishability of the correct hypothesis. Several criterion has been studied in the literature [WO11; RGV14a]. In this chapter we chose to use the nearest rival criterion[2]. Therefore, let us define the optimal set of functions $\mathcal{F}_{opt}$ as:

$$\mathcal{F}_{opt} = \underset{f\colon \mathcal{Z} \to \mathbb{R}^n}{\arg\max} \left\{ \mathcal{I}\big(f(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(f(Z_k), L\big)\big] \right\} \tag{3.7}$$

$\mathcal{F}_{opt}$ is a set since the maximum is reached by an infinite amount of functions. Indeed, if $f_{opt} \in \mathcal{F}_{opt}$, for any bijection $b$, $b \circ f_{opt}$ is also in $\mathcal{F}_{opt}$ since bijections do not affect MI. Note that $f$ is not restricted to be one-dimensional. Its domain is set to be $\mathbb{R}^n$ where $n$ can be any positive integer.

[2] Note that other criterion such as the distance with the mean of the wrong hypotheses could also have been used without modifying the analysis as discussed in remark 4.

### 3.2.5 Analytical Resolution

Being consistent with Proposition 2, $\mathcal{F}_{opt}$ depends on $L$ and therefore on the leakage model. Since knowledge on $\varphi$ is required anyway, this section assumes a full knowledge on $\varphi$ in order to conduct an analytical analysis to find the optimal $f$ function. Traces are also supposed to be acquired in an ideal set-up, without noise, so that, at least for a significant sub-part of the trace, $L = \varphi(Z_{k^*})$. Bounds taking into account imperfect knowledge on $\varphi$ as well as noise will be given in subsection 3.2.7.

A natural choice for the partition function would be to take $f = \varphi$ because it maximizes the left term in Equation 3.7: $\mathcal{I}\big(f(Z_{k^*}), \varphi(Z_{k^*})\big)$. But it may be possible to find a function that would maximize the global objective without maximizing the left term of Equation 3.7 (we emphasize that $f$ and $\varphi$ can be multi-dimensional which make the intuition harder to have). Theorem 4 actually proves that it is not possible and that whatever the leakage model, $\varphi \in \mathcal{F}_{opt}$. The main demonstration requires the use of a helper which is introduced in the form of a lemma hereafter.

**Lemma 1.** *Let $f$: $\mathcal{Z} \to \mathbb{R}^n$ be any function. For any leakage model $\varphi$: $\mathcal{Z} \to \mathbb{R}^n$ there exists a decomposition of $f$ into $f = f_2 \circ f_1$, with $f_1 : \mathcal{Z} \to \mathbb{N}$, $f_2 : \mathbb{N} \to \mathbb{R}^n$, satisfying the two following properties:*

*1)* $\exists\, f_3 : \mathrm{Im}\, f_1 \to \mathbb{R}^n$ *such that* $f_3 \circ f_1 = \varphi$

*2)* $\forall z \in \mathcal{Z}, f_2|_{f_1\left(\varphi^{-1}(\{\varphi(z)\})\right)}$ *is bijective of reciprocal* $f_2^{-1}|_{f_2 \circ f_1\left(\varphi^{-1}(\{\varphi(z)\})\right)}$

*Proof.* The proof is given in Appendix A. □

**Theorem 4.** *Let $P$ follow a uniform distribution. Let $Z_k$ represent the hypothetical intermediate variables such that $Z_k = g_k(P)$ with bijective $g_k$'s. Let $\varphi$: $\mathcal{Z} \to \mathbb{R}^n$ be the leakage model of the target so that $L = \varphi(Z_{k^*})$. Then, $\varphi \in \mathcal{F}_{opt}$.*

*Proof.* Let $\mathcal{S}_f = \mathcal{I}\big(f(Z_{k^*}), L\big) - \max_{k \neq k^*}\big[\mathcal{I}\big(f(Z_k), L\big)\big]$ represent the distinguishability score for a given function $f$ such that:

$$\mathcal{F}_{opt} = \underset{f:\, \mathcal{Z} \to \mathbb{R}^n}{\arg\max}\{\mathcal{S}_f\}$$

Since all the $Z_k$ follow a uniform distribution ($P$ follows a uniform distribution and the $g_k$ functions are bijective), the entropy $H(f(Z_k))$ is equal for all $k$. Then using $\mathcal{I}(A, B) = H(A) - H(A \mid B)$:

$$\mathcal{S}_f = -H\big(f(Z_{k^*}) \mid L\big) + \min_{k \neq k^*}\big[H\big(f(Z_k) \mid L\big)\big] \tag{3.8}$$

Symmetrically, using $\mathcal{I}(A, B) = H(B) - H(B \mid A)$:

$$\mathcal{S}_f = -H\big(L \mid f(Z_{k^*})\big) + \min_{k \neq k^*}\big[H\big(L \mid f(Z_k)\big)\big] \tag{3.9}$$

Let $f\colon \mathcal{Z} \to \mathbb{R}^n$ be any function. Applying Lemma 1, there exist $f_1$ and $f_2$ satisfying the two properties given in Lemma 1, such that $f = f_2 \circ f_1$. The goal is to show that $\mathcal{S}_f \leq \mathcal{S}_\varphi$. The proof is divided into two phases: first show that $\mathcal{S}_f \leq \mathcal{S}_{f_1}$ using Equation 3.8, then show that $\mathcal{S}_{f_1} \leq \mathcal{S}_\varphi$ using Equation 3.9. Let us start with Equation 3.8:

$$\begin{aligned}\mathcal{S}_f &= -H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) + \min_{k \neq k^*} \big[H\big(f_2 \circ f_1(Z_k) \mid L\big)\big] \\ &\leq -H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) + \min_{k \neq k^*} \big[H\big(f_1(Z_k) \mid L\big)\big]\end{aligned} \tag{3.10}$$

since applying $f_2$ in the second term can only decrease entropy (see Lemma 2). The goal is now to remove $f_2$ in the first term:

$$-H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) = \sum_{\substack{l \in \mathcal{L} \\ \bar{f}_2 \in \operatorname{Im} f_2}} P(l) \cdot P(\bar{f}_2 \mid l) \cdot log(P(\bar{f}_2 \mid l)) \tag{3.11}$$

$$\begin{aligned}P(\bar{f}_2 \mid l) &= P\big(f_2 \circ f_1(Z_{k^*}) = \bar{f}_2 \mid \varphi(Z_{k^*}) = l\big) \\ &= P\big(f_1(Z_{k^*}) \in f_2^{-1}(\bar{f}_2) \mid \varphi(Z_{k^*}) = l\big)\end{aligned} \tag{3.12}$$

Knowing that $\varphi(Z_{k^*}) = l$ implies that $Z_{k^*} \in \varphi^{-1}(\{l\})$ and also that $f_1(Z_{k^*}) \in f_1(\varphi^{-1}(\{l\}))$. Let $\mathcal{A}_l$ denotes $f_1(\varphi^{-1}(\{l\}))$ to avoid heavy notations. Then:

$$\begin{aligned}\varphi(Z_{k^*}) = l &\implies f_1(Z_{k^*}) \in \mathcal{A}_l \\ &\implies f_1(Z_{k^*}) \in f_2^{-1}(f_2(\mathcal{A}_l))\end{aligned} \tag{3.13}$$

which means that:

$$P(\bar{f}_2 \mid l) = \begin{cases} P\big(f_1(Z_{k^*}) \in f_2^{-1}|_{f_2(\mathcal{A}_l)}(\bar{f}_2) \mid l\big) & \text{if } \bar{f}_2 \in f_2(\mathcal{A}_l) \\ 0 & \text{else} \end{cases} \tag{3.14}$$

Lemma 1 states that $f_2|_{\mathcal{A}_l}$ is bijective of reciprocal $f_2^{-1}|_{f_2(\mathcal{A}_l)}$, so if $\bar{f}_2 \in f_2(\mathcal{A}_l)$:

$$P(\bar{f}_2 \mid l) = P\big(f_1(Z_{k^*}) = f_2^{-1}|_{f_2(\mathcal{A}_l)}(\bar{f}_2) \mid l\big) \tag{3.15}$$

Let us plug this result back into Equation 3.11:

$$\begin{aligned}-H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) = \sum_{l \in \mathcal{L}} \sum_{\bar{f}_2 \in f_2(\mathcal{A}_l)} &P(l) \cdot P\big(f_1(Z_{k^*}) = f_2^{-1}|_{f_2(\mathcal{A}_l)}(\bar{f}_2) \mid l\big) \cdot \\ &log\Big(P\big(f_1(Z_{k^*}) = f_2^{-1}|_{f_2(\mathcal{A}_l)}(\bar{f}_2) \mid l\big)\Big)\end{aligned} \tag{3.16}$$

Applying the following change of variable in the second sum: $\bar{f}_1 = f_2^{-1}|_{f_2(\mathcal{A}_l)}(\bar{f}_2)$:

$$\begin{aligned}-H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) = \sum_{l \in \mathcal{L}} \sum_{\bar{f}_1 \in \mathcal{A}_l} &P(l) \cdot P\big(f_1(Z_{k^*}) = \bar{f}_1\big) \mid l) \cdot \\ &log\Big(P\big(f_1(Z_{k^*}) = \bar{f}_1\big) \mid l)\Big)\end{aligned} \tag{3.17}$$

Finally, since $P\big(f_1(Z_{k^*}) = \bar{f}_1) \mid l\big) = 0$ when $\bar{f}_1 \in \operatorname{Im} f_1 \setminus \mathcal{A}_l$, one can artificially add some terms equal to 0 in the second sum:

$$\begin{aligned} -H\big(f_2 \circ f_1(Z_{k^*}) \mid L\big) &= \sum_{l \in \mathcal{L}} \sum_{\bar{f}_1 \in \operatorname{Im} f_1} P(l) \cdot P(\bar{f}_1 \mid l) \cdot log\big(P(\bar{f}_1 \mid l)\big) \\ &= -H\big(f_1(Z_{k^*}) \mid L\big) \end{aligned} \tag{3.18}$$

Applying this result to Equation 3.10 gives:

$$\begin{aligned} &\mathcal{S}_f \leq -H\big(f_1(Z_{k^*}) \mid L\big) + \min_{k \neq k^*} \big[H\big(f_1(Z_k) \mid L\big)\big] \\ &\mathcal{S}_f \leq \mathcal{S}_{f_1} \end{aligned} \tag{3.19}$$

which concludes the first step of the demonstration.

Now the goal is to show that $\mathcal{S}_{f_1} \leq \mathcal{S}_\varphi$. Lemma 1 guarantees that there exists $f_3$ such that $f_3 \circ f_1 = \varphi$. Let us use this in Equation 3.9:

$$\begin{aligned} \mathcal{S}_{f_1} &= -H\big(L \mid f_1(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(L \mid f_1(Z_k)\big)\big] \\ &\leq -H\big(L \mid f_1(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(L \mid \underbrace{f_3 \circ f_1}_{\varphi}(Z_k)\big)\big] \end{aligned} \tag{3.20}$$

since applying $f_3$ to the known variable can only increase the global entropy (see Lemma 3). Now using $L = \varphi(Z_{k^*})$:

$$\begin{aligned} &-H\big(L \mid f_1(Z_{k^*})\big) \leq 0 \\ &-H\big(L \mid f_1(Z_{k^*})\big) \leq -H\big(\varphi(Z_{k^*}) | \varphi(Z_{k^*})\big) = 0 \end{aligned} \tag{3.21}$$

Therefore:

$$\begin{aligned} &\mathcal{S}_{f_1} \leq -H\big(L \mid \varphi(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(L \mid \varphi(Z_k)\big)\big] \\ &\mathcal{S}_{f_1} \leq \mathcal{S}_\varphi \end{aligned} \tag{3.22}$$

Finally, using both part of the demonstration:

$$\mathcal{S}_f \leq \mathcal{S}_{f_1} \leq \mathcal{S}_\varphi \tag{3.23}$$

which ensures that $\varphi$ is better than any other functions and so that $\varphi \in \mathcal{F}_{opt}$. □

**Remark 4.** *Demonstration of Theorem 4 would have worked exactly the same if one had first fixed a particular hypothesis k, and tried to maximize $\mathcal{S}_{f,k} = \mathcal{I}\big(f(Z_{k^*}), L\big) - \mathcal{I}\big(f(Z_k), L\big)$. Therefore, for each k, $\varphi$ maximizes the distance between the score of $k^*$ and k which is an even stronger version of the theorem. One could not be sure that such a function would exist a priori, that is why $\mathcal{F}_{opt}$ has not been defined with this criterion. However, this shows a posteriori that Theorem 4 is still valid even if one decides to redefine $\mathcal{F}_{opt}$, for example using the distance with the mean (instead of the maximum) of the wrong hypotheses scores.*

**Interpretation.** This theorem tells that to conduct an optimal MIA, one has to transform the targeted variable $Z_k$ by applying the leakage model $\varphi$ (or any bijection of $\varphi$) and use $\mathcal{I}(\varphi(Z_k), L)$ as a distinguisher. Note the multidimensional aspect of this theorem since both $\varphi(Z_k)$ and $L$ can live in high dimensional space. This is a key point in this chapter that will be discussed in detail in subsection 3.4.1 which bridges this theorem with newest multidimensional MI estimators in order to derive a new attack. Note that this theorem also implies that if the leakage model is itself bijective, MIA is not a valid strategy since the distinguishability score would be bounded by 0.

### 3.2.6 Selecting Leakage Model *a Priori*

In a real-life experiment, one might not perfectly know the leakage model $\varphi$ but only an estimation $\bar{\varphi}$. This is especially true when working in an unsupervised context. This section provides a procedure to evaluate the correctness of $\bar{\varphi}$, helping to choose from multiple guesses $\bar{\varphi}_1, \ldots, \bar{\varphi}_n$. This test relies on the following observation:

**Proposition 3.** *Let $L = \varphi(Z_{k^*}) + N$, with $N$ an independent random variable representing the noise. Then: $\varphi \in \arg\max_f [\mathcal{I}\big(f(Z_{k^*}), L\big)]$*

*Proof.* On one hand:

$$\begin{aligned} \mathcal{I}\big(f(Z_{k^*}), L\big) &= \mathcal{H}(L) - \mathcal{H}\big(L \mid f(Z_{k^*})\big) \\ &\leq \mathcal{H}(L) - \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid f(Z_{k^*}), \varphi(Z_{k^*})\big) \\ &\leq \mathcal{H}(L) - \mathcal{H}(N) \end{aligned} \tag{3.24}$$

and on the other hand:

$$\begin{aligned} \mathcal{I}\big(\varphi(Z_{k^*}), L\big) &= \mathcal{H}(L) - \mathcal{H}\big(L \mid \varphi(Z_{k^*})\big) \\ &= \mathcal{H}(L) - \mathcal{H}\big(\varphi(Z_{k^*}) + N \mid \varphi(Z_{k^*})\big) \\ &= \mathcal{H}(L) - \mathcal{H}(N) \end{aligned} \tag{3.25}$$

Then:

$$\mathcal{I}\big(\varphi(Z_{k^*}), L\big) \geq \mathcal{I}\big(f(Z_{k^*}), L\big) \tag{3.26}$$

which concludes the proof. □

The identity function obviously also maximizes: $\mathcal{I}\big(f(Z_{k^*}), L\big)$ so combining this with Proposition 3:

$$\mathcal{I}\big(Z_{k^*}, L\big) = \mathcal{I}\big(\varphi(Z_{k^*}), L\big) \tag{3.27}$$

or,

$$\mathcal{I}\big(Z_{k^*}, L\big) = \max_k [\mathcal{I}\big(\varphi(Z_k), L\big)] \tag{3.28}$$

Then, if $k^*$ is known (for example in an evaluation setup) one can use Equation 3.27 and estimate $\mathcal{I}\big(Z_{k^*}, L\big)$ and $\mathcal{I}\big(\bar{\varphi}(Z_{k^*}), L\big)$ and compare them. If the latest is too far from $\mathcal{I}\big(Z_{k^*}, L\big)$, one may reject $\bar{\varphi}$ as being a good approximation of

the true underlying leakage model. If $k^*$ is unknown, the adversary can still use Equation 3.28 estimating $\mathcal{I}\big(\bar{\varphi}(Z_k), L\big)$ for all $k$, and comparing the maximum with $\mathcal{I}\big(Z_{k_0}, L\big)$ ($k_0$ can be chosen randomly since all the $Z_k$ variables are just permutation of each other which does not affect MI). Note that this test is only a rejection test since passing the test does not guarantee a good estimation of $\varphi$: for example, the identity function always passes the test.

### 3.2.7 Leakage Model Uncertainty and Noise

Let assume that the adversary has chosen a given estimation $\bar{\varphi}$ of $\varphi$. Let also assume that the ideal data $L = \varphi(Z_{k^*})$, used in Theorem 4, are now noisy so that the acquired data takes the following form: $\bar{L} = \varphi(Z_{k*}) + N$, with $N$ an independent random variable. This section aims at complementing Theorem 4 by lower bounding the distinguishably score $\bar{\mathcal{S}}_{\bar{\varphi}}$ that one would get in practice in such a context:

$$\bar{\mathcal{S}}_{\bar{\varphi}} = \mathcal{I}\big(\bar{\varphi}(Z_{k^*}), \bar{L}\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(\bar{\varphi}(Z_k), \bar{L}\big)\big] \tag{3.29}$$

Our goal is to compare $\bar{\mathcal{S}}_{\bar{\varphi}}$ with the optimal score $\mathcal{S}_{\varphi}$ (from Theorem 4) that one would get with the perfect knowledge of $\varphi$ and un-noised data such that:

$$\mathcal{S}_{\varphi} = \mathcal{I}\big(\varphi(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(\varphi(Z_k), L\big)\big] \tag{3.30}$$

**Proposition 4.** *$\bar{\mathcal{S}}_{\bar{\varphi}}$ is lower-bounded by the following inequality:*

$$\bar{\mathcal{S}}_{\bar{\varphi}} \geq \mathcal{S}_{\varphi} - H(N) - H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big) - \max_{k \neq k^*} \big[H\big(\bar{\varphi}(Z_k) \mid \varphi(Z_k)\big)\big] \tag{3.31}$$

*Proof.* Using the same argument as in Equation 3.9 one has:

$$\bar{\mathcal{S}}_{\bar{\varphi}} = -H\big(\varphi(Z_{k^*}) + N \mid \bar{\varphi}(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(\varphi(Z_{k*}) + N \mid \bar{\varphi}(Z_k)\big)\big] \tag{3.32}$$

Since removing noise on the right term can only decrease entropy:

$$\bar{\mathcal{S}}_{\bar{\varphi}} \geq -H\big(\varphi(Z_{k^*}) + N \mid \bar{\varphi}(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_k)\big)\big] \tag{3.33}$$

Now since $H(A + B) \leq H(A) + H(B)$ and using the independence of $N$:

$$\bar{\mathcal{S}}_{\bar{\varphi}} \geq -H(N) - H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_k)\big)\big] \tag{3.34}$$

Using $H(A \mid B) \geq H(A \mid C) - H(B \mid C)$ which can be shown through information Venn diagram:

$$\begin{aligned}\bar{\mathcal{S}}_{\bar{\varphi}} &\geq -H(N) - H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \varphi(Z_k)\big) - H\big(\bar{\varphi}(Z_k) \mid \varphi(Z_k)\big)\big] \\ &\geq -H(N) - H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big) + \min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \varphi(Z_k)\big)\big] \\ &\quad - \max_{k \neq k^*} \big[H\big(\bar{\varphi}(Z_k) \mid \varphi(Z_k)\big)\big]\end{aligned} \tag{3.35}$$

Now let $\mathcal{S}_\varphi$ appear in the equation:

$$\begin{aligned}\min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \varphi(Z_k)\big)\big] &= \min_{k \neq k^*} \big[H\big(\varphi(Z_{k^*}) \mid \varphi(Z_k)\big)\big] - \overbrace{H\big(\varphi(Z_{k^*}) \mid \varphi(Z_{k^*})\big)}^{0} \\ &= \mathcal{S}_\varphi\end{aligned} \tag{3.36}$$

So:

$$\bar{\mathcal{S}}_{\bar{\varphi}} \geq \mathcal{S}_\varphi - H(N) - H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big) - \max_{k \neq k^*} \big[H\big(\bar{\varphi}(Z_k) \mid \varphi(Z_k)\big)\big] \tag{3.37}$$

which concludes the proof. □

This proposition describes the impact of the noise and leakage model approximation in a quantitative way. Its qualitative interpretation is fairly intuitive. It clearly shows that one has two strategies to get closer to the optimal score: reducing the noise entropy or improving his guess on $\bar{\varphi}$. When $H(N)$ tends towards 0 and $\bar{\varphi}$ gets closer to $\varphi$ , $\bar{\mathcal{S}}_{\bar{\varphi}}$ tends towards the optimal score $\mathcal{S}_\varphi$. It also captures the fact that bijective errors do not impact the outcome of the attack since if there exists a bijection between $\bar{\varphi}(Z_k)$ and $\varphi(Z_k)$, both terms $H\big(\varphi(Z_{k^*}) \mid \bar{\varphi}(Z_{k^*})\big)$ and $\max_{k \neq k^*}[H\big(\bar{\varphi}(Z_k) \mid \varphi(Z_k)\big)]$ would be equal to 0.

## 3.3 MIA Against Masked Implementations

Masking is one of the most widely used countermeasures to protect implementations of block ciphers against side-channel analysis [Cha+99]. The idea is to split each sensitive intermediate value $Z$, into $d$ shares, following the relation:

$$Z_1 = Z * Z_2 * \cdots * Z_d \tag{3.38}$$

for a group operation $*$. The $d-1$ shares $Z_2, ..., Z_d$ are randomly chosen and the last one, $Z_1$, is processed such that Equation 3.38 is satisfied. Assuming the masks are uniformly distributed, the knowledge of $d-1$ shares does not tell anything about $Z$. However, partial knowledge on the $d$ shares can be exploited to retrieve information on $Z$. That is why, to defeat masking, one should use a distinguisher able to combine

the leakage of at least $d$ samples of the traces (assuming masks do not leak at the same time). Higher-order correlation attacks [Mes00] exploit a combining function, $C : \mathbb{R}^d \to \mathbb{R}$ , which transforms a multidimensional leakage into a single value such that the output of $C$ correlates with $Z$. The optimal combining function is unknown but most of the time, the centered product between the shares is used [PRB09].

### 3.3.1 MIA, a Natural Choice Against Masking

Although higher-order CPA attacks lead to successful key recoveries, they are not optimal from an information-theoretic point of view. Indeed, by the data processing inequality [BR12], the application of the combining function leads to an information loss. Opposed to Pearson's correlation, mutual information can deal with dependencies of multidimensional variables. Therefore, no combining function is required which makes MIA a very natural strategy against masked implementations. An extension of MIA in the context of masking has been proposed in [PR09] and [Bat+11]. The idea is very similar to the non-masked case. Concepts of MIA versions 1 and 2 still apply and one can use $\mathcal{I}(f(Z_k), L)$ as a distinguisher.

### 3.3.2 About the Partition Function in the Presence of Masking

Using $\mathcal{I}(f(Z_k), L)$ as distinguisher still raises the question of the optimal $f$ function. Theorem 4 cannot be applied straightforwardly since, for masked implementation, the leakage cannot be expressed as a deterministic function $\varphi(Z_{k^*})$ modulo some noise. Instead, with $Z_i$ representing the shares, one now has:

$$L = \sum_i \varphi_i(Z_i) \tag{3.39}$$

for some functions $\varphi_i : \mathcal{Z} \to \mathbb{R}^n$. Note that, as for the unmasked case, a noise-free version of the leakage is first considered to simplify the analysis. Noise will be added in subsection 3.3.3. Most of the time, the $\varphi_i$ supports can be supposed disjoint (*i.e.* leakages of the shares do not overlap). In that case, the leakage vector could be summarized as:

$$L = [\varphi_1(Z_1), \ldots, \varphi_d(Z_d)] \tag{3.40}$$

with $\varphi_i$ taking its values in a subspace of $\mathbb{R}^n$. Even with this simplification, we could not solve analytically the problem of finding an optimal partition function, or, in other words, a function $f \in \mathcal{F}_{opt}$ as defined in Equation 3.7. However, we still give some useful insights in the common case of Boolean masking on a device leaking the Hamming weight (or Hamming distance with a known value) of the shares.

For this specific case, [Bat+11] tried to use the Hamming weight as well as the identity function for $f$ (they were attacking the output of a DES S-box, therefore a non-injective intermediate variable). The Hamming weight produced better results. Their justification is that the Hamming weight is closer to the underlying leakage

model of the circuit. We do not find this justification straightforward especially in a multivariate context since even in the ideal case where the leakage could be expressed as:

$$L = [\mathrm{HW}(Z_{k^*} \oplus M), \mathrm{HW}(M)] \tag{3.41}$$

$\mathrm{HW}(Z_{k^*})$ is not directly related to any physical leakage. More generally, there is no proof that if all shares leak with the same leakage model $\varphi$, taking $f = \varphi$ is the optimal (or even a good) option. However, in the specific case of a Hamming weight leakage model, [PRB09] has shown that their exists a linear correlation between $\mathrm{HW}(Z_{k^*})$ and the covariance: $\mathrm{cov}\big(\mathrm{HW}(Z_{k^*} \oplus M), \mathrm{HW}(M)\big)$ which is a clue that there exists a non-negligible mutual information between $\mathrm{HW}(Z_{k^*})$ and $L$. However, we go further in this chapter by showing in Theorem 5 that there is actually no loss of information when applying the Hamming weight function to the $Z_{k^*}$ variable. This result can then be used to give a formal justification for using $f = \mathrm{HW}$, as done hereafter.

Let us introduce $\mathcal{F}_{Left}$ as the left part of Equation 3.7:

$$\mathcal{F}_{Left} = \underset{f:\, \mathcal{Z} \to \mathbb{R}^n}{\arg\max} \left\{ \mathcal{I}\big(f(Z_{k^*}), L\big) \right\} \tag{3.42}$$

This set does not consider the wrong hypotheses. Therefore it is not hard to find a function $f \in \mathcal{F}_{Left}$: the identity or any bijective function works. The problem is that with a bijective map, $\mathcal{I}\big(f(Z_{k^*}), L\big) = \mathcal{I}\big(f(Z_k), L\big)$ for any $k$. However, a non-injective function $f$ such that $f \in \mathcal{F}_{Left}$ would naturally decrease $\mathcal{I}\big(f(Z_k), L\big)$ and create some distinguishability. Such a function is not *a priori* likely to exist. But the following theorem shows that, while being highly non-injective, $\mathrm{HW} \in \mathcal{F}_{Left}$.

**Theorem 5.** *Let $L$ represent the leakage of a masked variable $Z_{k^*}$ with a mask $M$. Let both shares follow any bijection $b_1$ and $b_2$ of a Hamming weight leakage model so that:*

$$L = \big[b_1\big(HW(Z_{k^*} \oplus M)\big), b_2\big(HW(M)\big)\big] \tag{3.43}$$

*Then, $HW \in \mathcal{F}_{Left}$ or in other words: $\mathcal{I}\big(HW(Z_{k^*}), L\big) = \mathcal{I}\big(Z_{k^*}, L\big)$.*

The following proof is generalized to higher-order in section A.4, (see subsection 3.5.4 for an empirical validation), but for simplicity, only a first-order masking is considered here.

*Proof.* Since bijective transformations do not impact mutual information, one can consider without loss of generality that:

$$L = \big[\mathrm{HW}(Z_{k^*} \oplus M), \mathrm{HW}(M)\big] \tag{3.44}$$

Now let us evaluate $\mathcal{I}\big(f(Z_{k^*}), L\big)$ using Equation 1.26:

$$\mathcal{I}\big(f(Z_{k^*}), L\big) = \sum_{\bar{f} \in f(\mathcal{Z})} \sum_{l \in \mathcal{L}} P(\bar{f}, l) \cdot log\left(\frac{P(\bar{f}, l)}{P(\bar{f}) \cdot P(l)}\right) \tag{3.45}$$

One can split the first sum by summing on $z$ instead of $\bar{f}$:

$$\begin{aligned} \mathcal{I}\big(f(Z_{k^*}), L\big) &= \sum_{z \in \mathcal{Z}} \sum_{l \in \mathcal{L}} P(z, l) \cdot log\left(\frac{P(l \mid f(z))}{P(l)}\right) \\ &= \sum_{z \in \mathcal{Z}} \sum_{l \in \mathcal{L}} P(z) \cdot P(l \mid z) \cdot log\left(\frac{P(l \mid f(z))}{P(l)}\right) \end{aligned} \tag{3.46}$$

Since the identity function is bijective and maximizes this quantity, it would be enough to show that $P(l \mid \mathrm{HW}(z)) = P(l \mid z)$ for any given $z$ and a given $l = [\mathrm{HW}(z \oplus m), \mathrm{HW}(m)]$ for a fixed $m$. Let us start by the latter term:

$$P(l \mid z) = P(\mathrm{HW}(m)) \cdot P(\mathrm{HW}(z \oplus m) \mid z, \mathrm{HW}(m)) \tag{3.47}$$

To compute the right term one can evaluate the cardinal of the set $\mathfrak{M}$ of all the masks $m'$ satisfying the following conditions:

1) $\mathrm{HW}(m') = \mathrm{HW}(m)$

2) $\mathrm{HW}(z \oplus m') = \mathrm{HW}(z \oplus m)$

and divide by the number of byte with a Hamming Weight of $\mathrm{HW}(m)$ which is $\binom{8}{\mathrm{HW}(m)}$.

To evaluate this cardinal, we first show an invariance property. For any $m' \in \mathfrak{M}$, let $n_{m'}$ denotes the number of bits set to 1 in $m'$ such that there is also a bit set to 1 at the same position (0 to 7) in $z$. Then:

$$\begin{aligned} \mathrm{HW}(z \oplus m') = \mathrm{HW}(m') + \mathrm{HW}(z) - 2 \cdot n_{m'} &\iff \\ n_{m'} = \frac{\mathrm{HW}(m') + \mathrm{HW}(z) - \mathrm{HW}(z \oplus m')}{2} & \end{aligned} \tag{3.48}$$

Now since $m'$ satisfies the above two conditions:

$$n_{m'} = \frac{\mathrm{HW}(m) + \mathrm{HW}(z) - \mathrm{HW}(z \oplus m)}{2} \tag{3.49}$$

which does not depend on $m'$ anymore. As $n_{m'}$ has to be a positive integer, the above equation shows that:

$$\mathrm{HW}(m) + \mathrm{HW}(z) - \mathrm{HW}(z \oplus m) \notin 2\mathbb{N} \implies \mathfrak{M} = \emptyset \tag{3.50}$$

This allows us to define a generic $n$ as:

$$n = \begin{cases} \frac{\mathrm{HW}(m)+\mathrm{HW}(z)-\mathrm{HW}(z\oplus m)}{2}, \text{ if } \mathrm{HW}(m)+\mathrm{HW}(z)-\mathrm{HW}(z\oplus m) \in 2\mathbb{N} \\ -1, \text{ otherwise} \end{cases} \tag{3.51}$$

so that $\forall m' \in \mathfrak{M}, n_{m'} = n$.

Reciprocally, one can see that each byte $m'$ such that $\mathrm{HW}(m') = \mathrm{HW}(m)$ and $n_{m'} = n$ is in $\mathfrak{M}$. So to form a valid $m' \in \mathfrak{M}$ one has to choose first the position of the $n$ '1s' superposing with the '1s' in $z$, which lead to $\binom{\mathrm{HW}(z)}{n}$ possibilities. Then, choose the positions of the remaining '1s', which lead to $\binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n}$ possibilities. Therefore, with the convention $\binom{l}{k} = 0$ when $k$ is strictly negative:

$$P(\mathrm{HW}(z\oplus m) \mid z \text{ and } \mathrm{HW}(m)) = \binom{\mathrm{HW}(z)}{n} \cdot \binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n} \cdot \frac{1}{\binom{8}{\mathrm{HW}(m)}} \tag{3.52}$$

Injecting this into Equation 3.47 gives:

$$\begin{aligned} P(l \mid z) &= \frac{\binom{8}{\mathrm{HW}(m)}}{2^8} \cdot \binom{\mathrm{HW}(z)}{n} \cdot \binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n} \cdot \frac{1}{\binom{8}{\mathrm{HW}(m)}} \\ &= \frac{1}{2^8} \cdot \binom{\mathrm{HW}(z)}{n} \cdot \binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n} \end{aligned} \tag{3.53}$$

Now let us evaluate $P(l \mid \mathrm{HW}(z))$:

$$P(l \mid \mathrm{HW}(z)) = P(\mathrm{HW}(m)) \cdot \overbrace{P(\mathrm{HW}(z\oplus m) \mid \mathrm{HW}(z) \text{ and } \mathrm{HW}(m))}^{A} \tag{3.54}$$

And,

$$A = \sum_{\substack{z' \text{ s.t.} \\ \mathrm{HW}(z')=\mathrm{HW}(z)}} P(z' \mid \mathrm{HW}(z)) \cdot P(\mathrm{HW}(z'\oplus m) \mid z' \text{ and } \mathrm{HW}(m)) \tag{3.55}$$

Now using result from Equation 3.52:

$$\begin{aligned} A &= \sum_{\substack{z' \text{ s.t.} \\ \mathrm{HW}(z')=\mathrm{HW}(z)}} \frac{1}{\binom{8}{\mathrm{HW}(z)}} \cdot \binom{\mathrm{HW}(z')}{n} \cdot \binom{8-\mathrm{HW}(z')}{\mathrm{HW}(m)-n} \cdot \frac{1}{\binom{8}{\mathrm{HW}(m)}} \\ &= \binom{\mathrm{HW}(z)}{n} \cdot \binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n} \cdot \frac{1}{\binom{8}{\mathrm{HW}(m)}} \end{aligned} \tag{3.56}$$

since all the terms are constant in the sum and there are exactly $\binom{8}{\mathrm{HW}(z)}$ of them. Now plugging this into Equation 3.54 gives:

$$P(l \mid \mathrm{HW}(z)) = \frac{1}{2^8} \cdot \binom{\mathrm{HW}(z)}{n} \cdot \binom{8-\mathrm{HW}(z)}{\mathrm{HW}(m)-n} = P(l \mid z) \tag{3.57}$$

Thus,

$$\mathcal{I}\big(\mathrm{HW}(Z_{k^*}), L\big) = \mathcal{I}\big(Z_{k^*}, L\big) \tag{3.58}$$

which ensures that $\mathrm{HW} \in \mathcal{F}_{left}$ and concludes the proof. □

**Interpretation.** This theorem shows that when the shares leak in Hamming weight, it is sound to use $f = \mathrm{HW}$ in practice because it creates some distinguishability by decreasing the information only for the wrong hypotheses. Since the Hamming distance with a computable value can be rewritten as a Hamming weight, it also works in that case. However, Theorem 5 is not generalizable to any leakage model $\varphi$ (for example on 3 bits words, $\varphi = 2b_1 + b_2 + b_3$ gives a counter-example). Knowing if there exists a generic strategy against masking (depending on $\varphi$ but working for any $\varphi$) or if one will always be condemned to work on a case-by-case basis is an interesting question and may be handled in future works.

**Remark 5.** *Note that since* $\mathcal{I}\big(Z_{k^*}, L\big) = \mathcal{I}\big(HW(Z_{k^*}), L\big) = \max_k[\mathcal{I}\big(HW(Z_k), L\big)]$, *the procedure described in subsection 3.2.6 can also be applied on a masked implementation, to test the validity of the Hamming weight leakage model hypothesis. If the Hamming weight is too far from the true model, a practical alternative is to use only specific bits of the unmasked variable as partition function. An example of this is given in section 3.6.*

Considering the distinguishability score:

$$\mathcal{S}_f = \mathcal{I}\big(f(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(f(Z_k), L\big)\big] \tag{3.59}$$

HW has not been shown to be optimal. However, a partial result can be given introducing the concept of "wider" function.

**Definition 2.** *A function $f$ is said wider than $g$ if there exists another function $h$ such that: $h \circ f = g$.*

**Corollary 1.** *Let $L$ be defined as in Equation 3.43. Then, for any function $\bar{h}$ wider than HW, $\mathcal{S}_{HW} \geq \mathcal{S}_{\bar{h}}$.*

*Proof.* The proof is given in Appendix A. □

Even though we do not conjecture so, a function with a better distinguishability than the HW may exist. But a straightforward consequence of Theorem 5, given by Corollary 1, is that HW has a better distinguishability score than any other wider function.

### 3.3.3 Noise and Multidimensionality

The advantage of MINE is to be able to exploit the information contained in multiple samples at the same time. In a Hamming weight leakage scenario, the Hamming

weight of a variable is probably not going to leak perfectly on a single sample. Instead, multiple samples may leak a noisy version of it. To ensure that it is sound to use MINE and its multidimensional capabilities to mount an attack in the case of masking, one would need a multidimensional version of Theorem 5. This is exactly the purpose of Corollary 2, in which the noise is directly included.

In the context of masking the actual useful part of the leakage could be expressed as:

$$\begin{aligned} L = \big[&b_1\big(\mathrm{HW}(Z_{k^*} \oplus M)\big) + N_1, \ldots, b_{m_1}\big(\mathrm{HW}(Z_{k^*} \oplus M)\big) + N_{m_1}, \\ &b'_1\big(\mathrm{HW}(M)\big) + N'_1, \ldots, b'_{m_2}\big(\mathrm{HW}(M)\big) + N'_{m_2}\big] \end{aligned} \tag{3.60}$$

with $b_i$ and $b'_j$ being bijective maps, and $N_i$ and $N'_j$ being discrete noise variables independent of the shares. The following corollary shows that Theorem 5 is still valid in that case.

**Corollary 2.** *Let $L$ be defined as in Equation 3.60. Then, one still has $HW \in \mathcal{F}_{Left}$ as defined in Equation 3.42.*

*Proof.* As for Theorem 5, one can drop, without loss of generality, the bijections in $L$ as they do not affect the MI. Let $N$ be the noise vector $[N_1, \ldots, N_{m_1}, \bar{N}_1, \ldots, \bar{N}_{m_2}]$ and $\bar{L}$ the noise-free version of the leakage so that $L = \bar{L} + N$. As for Theorem 5, it is enough to show that $P(l \mid \mathrm{HW}(z)) = P(l \mid z)$ for any given $l$ and $z$. Decomposing on all the possible values of the noise one has:

$$\begin{aligned} P(l \mid z) &= \sum_{n \in \mathcal{N}} P(n) \cdot P(L = l \mid z \text{ and } n) \\ &= \sum_{n \in \mathcal{N}} P(n) \cdot P(\bar{L} = l - n \mid z) \end{aligned} \tag{3.61}$$

Since $\bar{L}$ is noise free, it consists of the repetition of the same two variables: $\mathrm{HW}(Z_{k^*} \oplus M)$ ($m_1$ times) and $\mathrm{HW}(M)$ ($m_2$ times). So for the probability $P(\bar{L} = l - n \mid z)$ to be non-zero, the vector $l - n$ should be constant on its first $m_1$ coordinates, and constant on its $m_2$ last one. Let $\mathcal{N}_c$ be the subset of $\mathcal{N}$ verifying the precedent property. If $n \notin \mathcal{N}_c$, then:

$$P(\bar{L} = l - n \mid z) = P(\bar{L} = l - n \mid \mathrm{HW}(z)) = 0 \tag{3.62}$$

Else, if $n \in \mathcal{N}_c$, then, with $a_n = (l - n)[1]$, $b_n = (l - n)[m_1 + m_2]$ and $\tilde{L} = [\mathrm{HW}(Z_{k^*} \oplus M), \mathrm{HW}(M)]$:

$$P(\bar{L} = l - n \mid z) = P(\tilde{L} = [a_n, b_n] \mid z) \tag{3.63}$$

So Equation 3.61 can be rewritten as:

$$P(l \mid z) = \sum_{n \in \mathcal{N}_c} P(n) \cdot P(\tilde{L} = [a_n, b_n] \mid z) \tag{3.64}$$

Since, Theorem 5 tells that $P(\tilde{L} = [a_n, b_n] \mid z) = P(\tilde{L} = [a_n, b_n] \mid \mathrm{HW}(z))$:

$$\begin{aligned} P(l \mid z) &= \sum_{n \in \mathcal{N}_c} P(n) \cdot P(\tilde{L} = [a_n, b_n] \mid \mathrm{HW}(z)) \\ P(l \mid z) &= \sum_{n \in \mathcal{N}_c} P(n) \cdot P(\bar{L} = l - n \mid \mathrm{HW}(z)) \\ P(l \mid z) &= P(l \mid \mathrm{HW}(z)) \end{aligned} \tag{3.65}$$

which concludes the proof. □

This corollary shows that it is sound to use $\mathcal{I}(\mathrm{HW}(Z_k), L)$ as distinguisher even when considering a noisy multidimensional leakage vector. Theorem 5 still applies and MINE may benefit from the different leakage sources resulting in an attack (presented in the next section) exploiting a bigger amount of the available information.

## 3.4 Neural Estimated Mutual Information Analysis (NEMIA)

This section aims at formally describing the new attack proposed in this chapter. Note that throughout this work, a tool able to compute $\mathcal{I}(Z, L)$ with high dimensional variables has been assumed to exist. This research has been driven by recent progress regarding neural estimation techniques. However, this work is not absolutely related to MINE. It would stay sound with any MI estimator able to work in high dimension. In particular, any progress in the field, which is likely to happen since it is a very active domain, would instantly impact the attack efficiency.

### 3.4.1 Multidimensional Paradigm

MINE is by essence a tool that estimates MI in a multidimensional way, enabling to compute the MI between $f(Z_k)$ and full (or at least large part of) traces. This was not possible with classical MI estimators which do not scale with high dimensional variables. Until now, MIA was only performed with the following distinguisher:

$$\mathcal{D}_{old}(k) = \max_i \mathcal{I}(f(Z_k), L[i]) \tag{3.66}$$

where $L[i]$ represents the $i$-th sample of the trace. This way, trace dimension is kept low, allowing methods such as the histogram or the kernel density estimation [PR09] to produce reliable results. However, this comes at the cost of sacrificing some, and maybe a large part, of the available information. MINE allows to directly use:

$$\mathcal{D}_{new}(k) = \mathcal{I}(f(Z_k), L) \tag{3.67}$$

as a distinguisher. This comes with two main advantages:

- Intermediate variables often leak at multiple instants in the trace. MINE allows to exploit all these leakage sources at the same time.

- Other intermediate variables, statistically dependent from the first one, can also leak information. For example, there could be some useful information about an AES key, before and after the application of the first S-box. In this context, MINE could exploit leakage from both intermediate variables at the same time, without any assumption related to the kind of link between these variables.

Theorem 4 states that the optimal distinguisher is $\mathcal{I}(\varphi(Z_k), L)$ with $\varphi$ being the leakage model. It is important to note that $\varphi(Z_k)$ itself can be multidimensional. Therefore, an optimal MI attack should exploit this multidimensionality of the leakage model to increase the distinguishability of the correct hypothesis. However, it is frequent that multiple samples leak with the same underlying model: for example, a noisy version of the Hamming weight of $Sbox[k^* \oplus P]$ can leak multiple times in the trace. In such a context, the deterministic parts of the leakage of all these samples are all bijectively related. As adding bijection of the same variables multiple times would not change the MI, one can keep only one version of each different sub-leakage model. For example, if the target leaks (maybe multiple times) the Hamming weight of the first S-box of an AES and the Hamming distance between the S-box and $k \oplus P$, $Z_k$ could be defined as $k \oplus P$ and one could replace $\varphi(Z_k)$ by the two-dimensional vector:

$$\Big[\mathrm{HW}\big(Sbox[Z_k]\big), \mathrm{HW}\big(Sbox[Z_k] \oplus Z_k\big)\Big] \tag{3.68}$$

**Remark 6.** *In practice, one may deliberately drop some intermediates variables for not being enough discriminating for wrong key candidates making them less tolerant regarding errors in the estimation of $\varphi$. For example, it is theoretically possible to use leakage on a xor: $HW(k \oplus P)$ (assuming a Hamming weight a priori) but it is preferable to use intermediate variables where each bit depends on multiple bits of k such as the output of an S-box. Indeed, these variables are more discriminating since single bit errors on k are diffused to the whole variable which prevents from rewarding wrong hypotheses with several correct bits.*

**Scalability with Masking Order.** In the context of masking, another advantage of multidimensionality emerges. In a classical $d$-order attack one often does not know the exact leakage time of each share, and therefore, has to compute the value of the distinguisher for each possible tuple $(i_1, \ldots, i_d)$ and select the maximum. In the case of MIA the old distinguisher takes the following form:

$$\mathcal{D}_{old}(k) = \max_{i_1,\ldots,i_d} \{\mathcal{I}(f(Z_k), L[i_1, \ldots, i_d])\} \tag{3.69}$$

For long traces, this can become a huge constraint since the total number of tuples grows exponentially with the masking order. Our version of the MIA which uses $\mathcal{I}(f(Z_k), L)$ as distinguisher, does not suffer from this since it does not require any kind of recombination between time samples. Note that it does not mean that

masking is useless: it still decreases exponentially the information contains in side-channel traces [PR13] and an attack may require exponentially more traces to succeed. However, for a fixed number of traces, the computational effort required to mount a NEMIA does not scale exponentially with the order of the attack.

### 3.4.2 Attack Description

A step-by-step description of the NEMIA is given hereafter. It takes as input a set of traces and outputs a ranking of the key hypotheses.

1. Define an *a priori* $\bar{\varphi}$ on the leakage model. It can be multidimensional if multiple intermediate variables related to the key leak information. Also, a single intermediate variable can have different leakage models at different times. The test described in subsection 3.2.6 can be used to detect wrong *a priori*. Even if MIA is tolerant regarding estimation errors on $\varphi$, better *a priori* lead to more efficient attacks.
2. Compute, for all $k$, the hypothesis vectors: $H_k = \bar{\varphi}(Z_k)$.
3. Compute $\mathcal{I}(H_k, L)$, for all $k$, with MINE[3]. This implies to run 256 neural network trainings.
4. Rank the key hypotheses.

For masked implementation, the only step that changes is the construction of $H_k$. If the shares have a Hamming weight leakage model, Theorem 5 proves that it is sound to use the Hamming weight of the corresponding unmasked intermediate variable in $H_k$ (one may do this for multiple intermediate variables). For a generic leakage model of the shares, the best strategy to adopt remains an open question. It appears that, in some cases, it is efficient to keep a restrictive number of bits of the unmasked variable as partition function, for example in a situation where some bits of the shares leak much more information than the others (an example of this is given in section 3.6).

## 3.5 Simulation Experiments

In order to gain confidence in the mathematical results presented in this chapter, as well as to gain intuition about their implications, this section presents experiments on synthetic data. These experiments all use the same architecture of MINE. It should be seen as a proof of concept with almost no hyper-parameters tuning and without considering recent optimizations nor improvements in the technique (non-exhaustively: [CL20; Lin+19; Cha+19]). A study focused on deep learning optimizations would be interesting but is out of the scope of this thesis.

[3] For robustness, the MI estimation is not set to be the supremum of the validation loss, but instead, the supremum of a moving average along the epochs off the validation loss with a window size of $w$ which depends on the variability between epochs ($w = 10$ in this chapter)

### 3.5.1 About the Network's Architecture

The network's input layer consists of a concatenation of both $Z$ and $L$ variables. We have shown in Chapter 2 that the representation of $Z$ is important and that one should use the One-Hot Encoding (OHE) or a binary encoding of $Z$ (unless otherwise specified we used the OHE in this chapter). The output layer is a single neuron. Other layers are not specified by the method and should be adapted to the underlying problem (*e.g.* convolutional layers to counter jitter or traces misalignment).

For our experiments, we used a Convolutional Neural Network (CNN) where a batch normalization layer is added after the first layer and dropout layers are inserted after each hidden layer in order to mitigate overfitting. The activation function is set to the Exponential Linear Unit (ELU) and the batch size to 1000. The precise architecture is depicted in Appendix B. The validation dataset represents 20 percent of the full dataset.

### 3.5.2 On the Importance of the *a Priori*

The main message of Theorem 4 is that, to maximize the distinguishability of the correct hypothesis, one should use the leakage model $\varphi$ to create the hypothesis vectors $H_k$. In a classical side-channel scenario, with no other specific information, one may often guess a Hamming weight leakage of the intermediate variables. This is justified by electronic arguments. However, it has been shown that bits may have different leakage behaviours, such as leakage weighting or even sign inversions [CLH20]. To illustrate Theorem 4, 10k synthetic traces leaking a slightly modified version $\varphi_0$ of the Hamming weight have been generated. They consist of a single sample leaking the Hamming weight of $Z_{k^*} = Sbox(k^* \oplus P)$ but with a flipped sign for bit 0 so that:

$$\varphi_0(z) = -z_0 + \sum_{i=1}^{7} z_i \tag{3.70}$$

with $z_i$ representing the $i$-th bit of $z$. Some Gaussian noise has been added to the traces so that $L = \varphi_0(Z) + \mathcal{N}(0,1)$. Figure 3.1 shows the results of a NEMIA with $k^* = 0$, both with HW and $\varphi_0$ as partition function. As predicted by Theorem 4, the distinguishability score:

$$\mathcal{S}_f = \mathcal{I}\big(f(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(f(Z_k), L\big)\big] \tag{3.71}$$

is higher for $f = \varphi_0$ than for $f = \mathrm{HW}$. Obviously, an attacker may not know $\varphi_0$ and an attack with the Hamming weight still succeeds in that case. However, this shows that, if by any means, an adversary knows the particularity of bit 0 of such a target, he can perform more efficient attacks.

**Semi-Supervised Attacks.** This opens the idea of semi-supervised attacks. One of the main problems of profiling attacks is the portability [EG12b]. Indeed, during the characterization phase, the adversary learns a perfect representation of the

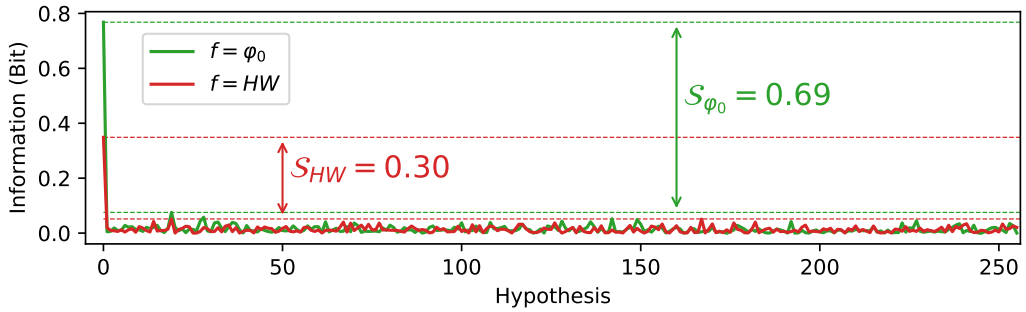



FIGURE 3.1: $\mathcal{I}\big(f(Z_k), L\big)$ in terms of $k$, with $k^* = 0$

leakage model which may overfit on the particular target which is profiled. It has been shown that portability to other targets is not trivial. Therefore NEMIA could be turned into a semi-supervised attack where the purpose of the characterization phase is only to learn general leakage characteristics, such as the sign or weighting of each bit, and use them as an improved *a priori* for a NEMIA. Since NEMIA is agnostic towards bijective errors in the leakage model estimation, it has a better chance of being portable on many other targets similar to the one used for profiling.

### 3.5.3 The Potential of Multidimensionality

One of the main advantages of NEMIA is its potential to exploit at the same time, multiple leakage sources. It is possible that multiple intermediate variables leak information on the key and each particular variable may leak multiple times in the traces. This section aims at showing how NEMIA could exploit all these leakage sources as well as to compare it with other state of the art attacks.

#### Traces Generation

To this aim, a dataset of 100k synthetic traces have again been generated. These traces represent the leakage of an AES that both leaks $A_{k^*} = \mathrm{HW}(Sbox[k^* \oplus P])$ and $B_{k^*} = \mathrm{HW}(Sbox[k^* \oplus P] \oplus (k^* \oplus P))$. One could imagine that the bus leaks the Hamming weight of the S-box data and that the update of the state register leaks the Hamming distance with its precedent value (*e.g.* [Mor+08]).

One of the strength of using deep learning in an unsupervised attack is the absence of need for preprocessing techniques. To highlight this fact we also added 90 % of uninformative samples as well as some misalignment in the traces following the shifting deformation procedure introduced in [CDP17] which simulates a random delay effect of maximal amplitude $T$ by shifting each trace by a random number uniformally drawn between 0 and $T$. The procedure for the trace generation is depicted in algorithm 2.

**Algorithm 2:** Generate Traces

**Output:** *L, a (100k, 1010) array*
**Output:** *P, a (100k array*
$P \leftarrow$ Draw 100k plaintexts uniformly from $[\![0, 255]\!]$
$A \leftarrow \mathrm{HW}(\texttt{Sbox}[P \oplus k^*])$
$B \leftarrow \mathrm{HW}(\texttt{Sbox}[k^* \oplus P] \oplus (k^* \oplus P))$
$S \leftarrow$ Draw 1010 samples from a Gaussian$\mathcal{N}(0, 10^2)$ `// Generate a baseline shape`

$L \leftarrow$ Repeat $S$ 100k times to form a (100k, 1010) array
**for** $1 \leq i \leq 100k$ **do**
  **for** $1 \leq j \leq 50$ `// Add leakage one every 10 samples` **do**
    $L[i, 10 * j] \leftarrow L[i, 10 * j] + A[i]$
    $L[i, 10 * j + 500] \leftarrow L[i, 10 * j + 500] + B[i]$
  **end**
**end**
$R \leftarrow$ Draw an array (100k, 1010) of random number from a Gaussian $\mathcal{N}(0, 20^2)$
$L \leftarrow L + R$ `// Add some noise`
**for** $1 \leq i \leq 100k$ **do**
  $sh \leftarrow$ Draw a random integer uniformly from $[\![0, 10]\!]$
  $L[i] \leftarrow \mathrm{Roll}(L[i], sh)$ `// Apply the jitter (Roll shift the array by` $sh$`)`

**end**
**return** L, P

**Compared Strategies**

We used the generated dataset to compute and compare guessing entropies for the following attack strategies:

1. A classical CPA [BCO04] with a Hamming weight model.
2. A classical MIA with a Hamming weight model computing the MI with the histogram method described in [Bat+11] with 9 bins.
3. NEMIA$_{Partial}$, only considering the Hamming weight leakage ($A_k$) to construct the hypothesis vectors $H_k = A_k$:
4. NEMIA$_{Full}$, considering both leakages ($A_k$ and $B_k$) to construct the hypothesis vectors $H_k = [A_k, B_k]$.
5. The Differential Deep Learning Analysis (DDLA) introduced in [Tim19]. It is sound to compare NEMIA to DDLA since both methods use deep learning with an unsupervised approach. It builds 256 classifiers, one for each key hypothesis, and uses a metric (we used the accuracy as suggested in [Tim19]) as a distinguisher. Note that a partition function also has to be applied to the intermediate variables but its optimal choice has not been discussed in [Tim19]. We use the Hamming weight function in this experiment.
6. A classical deep learning supervised attack [MPP16], denoted DL-supervised, where a network is train to classify amoung the 256 classes. The total number of traces is divided into 80% for training and 20% for the actual attack. The architechture of the network is depicted in Appendix B.
7. The same deep learning attack but in a non-limited setup regarding the number of traces during profiling. In practice we have trained the network using another dataset of 100k traces generated with algorithm 2. This attack is denoted DL-supervised$_{\infty}$.

Figure 3.2 shows the evolution of the average rank of $k^*$ for each attack. Each point represents the average over 100 attacks computed with traces randomly drawn from the 100k traces dataset. It appears that for low numbers of traces, CPA performs the best amoung the unsupervised attacks but this is not very meaningful since attacks with such guessing entropies (greater than 20 on a single key byte) are not really exploitable for a full key recovery. Deep learning attacks behave more like if they had a threshold: after a certain number of traces, one can observe a quick drop in their guessing entropies.

As predicted by the theory, $NEMIA_{Full}$ converges faster towards a ranking of 0 than $NEMIA_{Partial}$, and both converge faster than CPA. $NEMIA_{Partial}$ outperforms DDLA and also the supervised DL attack with a restricted number of traces for profiling. This may seem counter-intuitive but in this case we argue that the learning

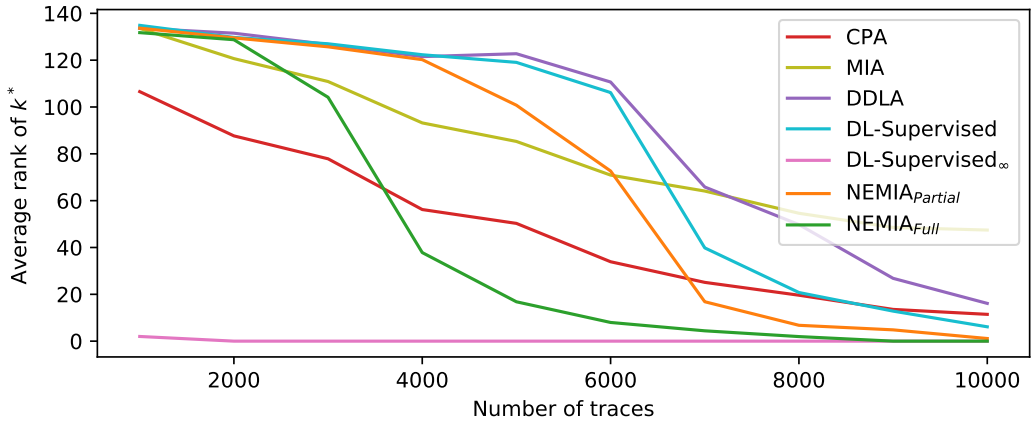



FIGURE 3.2: Guessing entropies for the considered attacks

problem is simpler for NEMIA since it has to deal with 9 different classes instead of 256 for the DL model. This may result into succesfull profiling with less traces. In this case, the application of the partition function is only beneficial and does not induce information loss since the true leakage model is known.

To the best of our knowledge, classical MI-based attacks always performed worse than CPA in the literature, when considering the Hamming weight model, which is again confirmed by our results. This experiment shows that in a low-information scenario (noisy traces with jitter), NEMIA may be worth considering among the other unsupervised attacks.

### 3.5.4 Empirical Validation of Theorem 5

Theorem 5 may seem very counterintuitive since it basically states that: when shares of a Boolean masking leak in a Hamming weight model, one has:

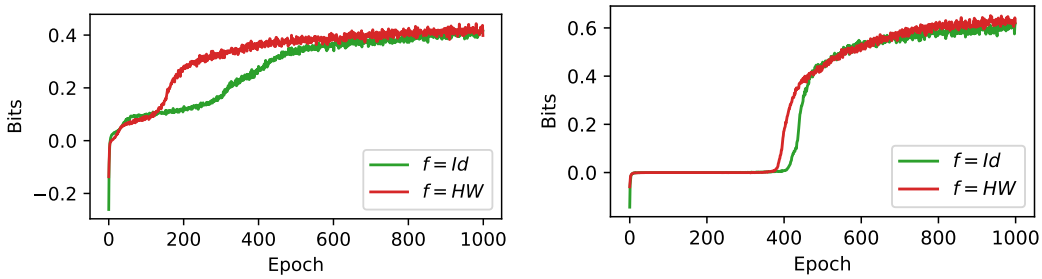
$$\mathcal{I}\big(\mathrm{HW}(Z_{k^*}), L\big) = \mathcal{I}\big(Z_{k^*}, L\big) \tag{3.72}$$

which is surprising since HW is highly non-injective and should at first glance, decrease the information. Corollary 2 says that this is even true when multiple samples leak a noised version of the Hamming weight of the shares. To verify this claim, 100k synthetic traces have been generated considering the following leakage:

$$\begin{aligned} L = \big[&\mathrm{HW}(Z_{k^*} \oplus M) + N_1, \ldots, \mathrm{HW}(Z_{k^*} \oplus M) + N_{10}, \\ &\mathrm{HW}(M) + N_{11}, \ldots, \mathrm{HW}(M) + N_{20}\big] \end{aligned} \tag{3.73}$$

with $Z_{k*} = Sbox(k^* \oplus P)$, $N_i = \mathcal{N}(0,1)$ and $M$ being uniformly distributed in $\mathbb{Z}/256\mathbb{Z}$.

Figure 3.3a shows the evolution of the loss function for both the HW and the identity function for the correct key hypothesis. As predicted, both converge towards the same value which confirms experimentally that the application of the HW




(A) First order masking ($\sigma = 1$)

(B) Second order masking ($\sigma = 0.5$)

FIGURE 3.3: Comparison of $\mathcal{I}(Z_{k^*}, L)$ and $\mathcal{I}(\mathrm{HW}(Z_{k^*}), L)$ on masked synthetic traces

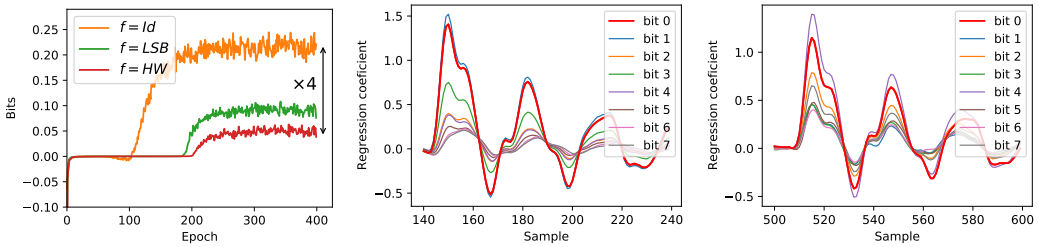
does not alter information. The HW function is even doing a little better which can be explained with practical machine learning considerations. Indeed, the information being constant, it is easier for the network to learn with a 9-classes variable than with a 256 classes variable (note that in this experiment, $id(Z_{k^*})$ has been encoded in binary rather than in OHE, because it produced slightly better results). Also, since overfitting was not really a problem in this experiment, the dropout parameter has been set to $p = 0.1$.

Figure 3.3b shows the result of the same experiment performed on a second-order masking, with three shares and 10 leakage samples for each. Noise has been a bit decreased ($\sigma = 0.5$ instead of 1) to keep comparable level of information. This result confirms that Theorem 5 is generalizable to higher-order masking, as shown in section A.4, and that MINE is able to extract information even with a second-order masking.

## 3.6 A practical Case: Attack on ASCAD

This section provides a real case experiment on the public dataset of ASCAD [Ben+18]. We only considered the training dataset composed of 50k traces composed of 700 samples focusing on the processing of the third byte (the first two are not masked) of the masked state $Sbox(k^*[3] \oplus P[3]) \oplus r[3]$, with $r$ being the mask variable and with a fixed key $k^*[3]$.

Since it is a masked implementation, the test described in remark 5 has first been conducted. Results are presented in Figure 3.4a. $\mathcal{I}(Z_{k^*}, L)$ is more than four times greater than $\mathcal{I}(\mathrm{HW}(Z_{k^*}), L)$ which indicates that the underlying leakage of the shares is far from a pure Hamming weight model. In parallel to this, authors in [Tim19] applied the DDLA strategy which also requires a partition function and they reported that, for the ASCAD database, only keeping the value of the Least Significant Bit (LSB) produced better results than the Hamming weight without giving further explanations.




(A) $\mathcal{I}(f(Z_{k^*}), L)$ (B) $r[3]$ (C) $Sbox(k^*[3] \oplus P[3]) \oplus r[3]$

FIGURE 3.4: Analysis of the ASCAD leakage model:
a) Test from remark 5 - b) & c) Coeficients of a linear regression on the given variable

In a real attack scenario, an adversary mounting a NEMIA could obviously try to use every single bit of the unmasked variable as partition function. But in order to gain some intuition, and since the masks values are given in the database, we first performed a linear regression on both shares, assuming bits leak independently so that the actual leakage of share $s$ is: $\sum_{i=0}^{7} \alpha_i s_i + \beta$. Figures 3.4b and 3.4c show the evolution of the $\alpha_i$ coefficients, on a leakage window for both shares. Since the implementation is protected by a Boolean masking, a mono-bit leakage is exploitable only if it is present on the same bit of both shares. Out of the 8 bits, bit 0 (LSB) is clearly the one that leaks the most information since its coefficients are among the greatest ones in both shares. Thus, we computed with MINE $\mathcal{I}(Z_{k^*}[0], L)$ where $Z_{k^*}[0]$ represents the LSB of $Sbox(k^*[3] \oplus P[3])$. It returned 0.09 bit, which is two times more than the information left with the Hamming weight (see Figure 3.4a). This indicates that the LSB may be a good partition function since it is highly non-injective and still keep a decent amount of information for the correct hypothesis. We also tried with other bits but the information, while being non-zero, was significantly lower. Even though attacks with the Hamming weight were successful, we decided to use the LSB as partition function for the rest of our analysis. The attacks presented in this section uses the whole 700 samples as input. We compared the following attacks:

1. A classical second-order CPA [PRB09] with a Hamming weight model.
2. A second-order MIA with a LSB model computing the MI with the histogram method described in [Bat+11] with 9 bins.
3. NEMIA with LSB as a partition function. The architechture of the network is depicted in Appendix B.
4. The Differential Deep Learning Analysis (DDLA) using the accuracy as distinguisher and with LSB as partition function. The architechture of the network is depicted in Appendix B.

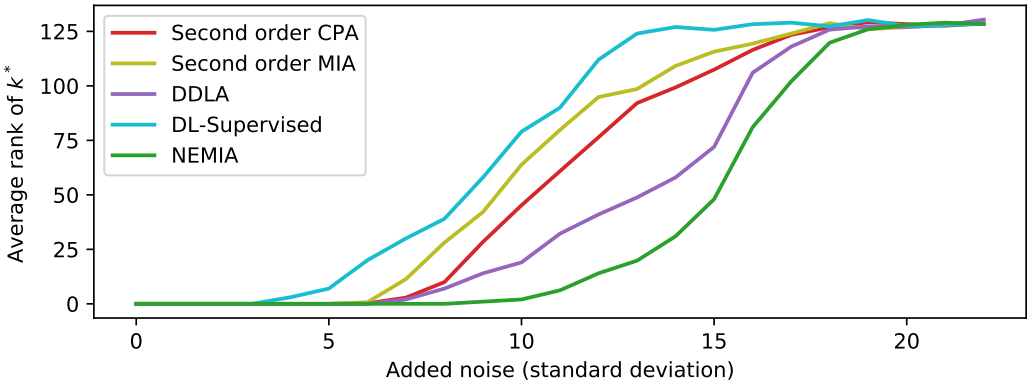



FIGURE 3.5: Guessing entropies for the considered attacks on ASCAD with added noise

5. A deep learning supervised attack [MPP16], denoted DL-supervised, where a network is train to classify among the 256 classes (we do not apply any partition functions because it is not required in a supervised context). The total number of traces is divided into 80% for training and 20% for the actual attack. The architechture of the network is depicted in Appendix B.

**Results.** In order to evaluate the potential of NEMIA to exploit leakage even in very low information context, the dataset has been artificially degraded adding Gaussian noise $\mathcal{N}(0, \sigma^2)$ to each sample. All the attacks have been performed with $\sigma$ going from 0 to 20, using the whole 50k traces. For each level of noise, the attacks have been repeated 10 times (with different random sampling of the noise) in order to compute the average rank of the correct hypothesis. Results are presented in Figure 3.5. They confirm that NEMIA is able to succeed in situations where the considered state of the art attacks would not.

As for the experiment in subsection 3.5.3, the DL-Supervised attack performs worse than the unsupervised attack which is non-intuitive. However, an adversary performing a supervised attack would likely have an unlimited amount of traces for profiling which will give rise to the best attack in terms of attack traces. We lack traces to compute the equivalent of DL-Supervised$_\infty$ for such noise level. It appears that the application of the partition function (the LSB which only has two classes) makes the training easier for the networks which explain why a DL model, with a restricted number of traces for profiling, underperforms compared to the supervised attacks. Obvioulsy the partition function could be applied even in the supervised case (*i.e.* building a two classes classifier) but one would then loose the interest of being in a supervised context where no assumption has to be done on the leakage model.

## 3.7 Conclusion and Perspectives

This chapter first proposes a clarification of the state of the art around the MIA. It provides rigorous proofs whose goal is to derive the optimal MI-based attack working with high-dimensional traces. Combined with recent breakthroughs on neural MI estimation techniques, this allows to mount a new attack: the NEMIA, which benefits from both the strength of deep learning and information theory. Being able to exploit at the same time multiple leakage sources, it pushes the amount of effectively used information (depending on the strength of the attacker *a priori*) closer to the actual existing information between traces and secret. Simulations and real case experiments are presented to support the mathematical theory developed in this chapter. They also show that NEMIA outperforms classical uni/bivariate side-channel attacks and that this strategy may be worth to consider in low-information/high-noise situations, where all (or a large part of) the available information contained in traces need to be used to mount a successful attack.

Several lines of research emerge from this chapter. The mathematical analysis could be further extended, especially in the context of masking, in order to develop strategies for generic leakage model of the shares or for other masking schemes such as arithmetic masking. On the practical side, integrating the latest optimization on neural estimation techniques, as well as deep learning research on optimal networks architecture and hyper-parameters would allow to mount more efficient attacks, taking as input larger portion of the traces, leading to better/easier attacks.

The two main difficulties of conducting a NEMIA in practice is the choice of the partition function, which can be hard when the adversary has very low knowledge on how the device leaks, and the time complexity of the attack since it requires 256 neural network trainings. The next chapter proposes a solution to tackle both problems at the same time.

# Chapter 4

# The EVIL Machine

## Encode, Visualize and Interpret the Leakage

*"Insanity is applying a singular dimensional answer to a multi dimensional experience."*

David Ault

*Unsupervised side-channel attacks allow extracting secret keys manipulated by cryptographic primitives through leakages of their physical implementations. As opposed to supervised attacks, they do not require a preliminary profiling of the target, constituting a broader threat since they imply weaker assumptions on the adversary model. Their downside is their requirement for some a priori knowledge on the leakage model of the device. On one hand, stochastic attacks such as the Linear Regression Analysis (LRA) allow for a flexible a priori, but are mostly limited to a univariate treatment of the traces. On the other hand, model-based attacks require an explicit formulation of the leakage model but have been extended to multidimensional versions allowing to benefit from the potential of Deep Learning (DL) techniques (see NEMIA in the preceding chapter). The EVIL Machine Attack (EMA), introduced in this chapter, aims at taking the best of both worlds. Inspired by generative adversarial networks, its architecture is able to recover a representation of the leakage model, which is then turned into a key distinguisher allowing flexible a priori. In addition, state-of-the-art DL techniques require 256 network trainings to conduct the attack. EMA requires only one, scaling down the time complexity of such attacks by a considerable factor. Simulations and real experiments show that EMA is applicable in cases where the adversary has very low knowledge on the leakage model, while significantly reducing the required number of traces compared to a classical LRA. Eventually, a generalization of EMA, able to deal with masked implementation, is introduced.*

## 4.1 Introduction

### 4.1.1 Context

Side-Channel Analysis (SCA) is mainly divided into two categories: supervised and unsupervised SCA and their utilization depends on the considered threat model. In the first one, the adversary is supposed to be able to conduct a profiling step of the target, most likely on a clone device, in which she learns the leakage model of the intermediate variables and then adopts a maximum likelihood approach to recover the secret key. This includes strategies such as Gaussian template attacks [CRR02] or deep learning profiled attacks [MDP19]. If the model is perfectly learned during the characterization phase, these attacks are known to be optimal from an information theory point of view.

However, being able to conduct a sound profiling step is not always possible and refers to a strong threat model for the adversary. Indeed, the latter one may not be able to obtain a clone with full control on the device and even in cases where she could, template portability issues [EG12a] may still stand in the way. In this case, the adversary can "replace" the profiling of the target by its *a priori* on the leakage model to mount what is called unsupervised SCA. Such an *a priori* usually comes from physical assumptions and is central to unsupervised SCA. Indeed, as shown in [WOS14], there does not exist a generic unsupervised strategy that would work without requiring such an *a priori*. However, *a priori* is a very vague term that only captures the fact that the adversary has to have some knowledge about the leakage model for the attack to work. This knowledge can take many different forms.

In a first type of unsupervised SCA, known under the stochastic attacks, the *a priori* takes the form of a parametric model. For example, the well-known Linear Regression Analysis (LRA) [Dog+12] falls into this category. The advantage of such attacks is their ability to work with weak *a priori* such as the only assumption that the bits of the sensitive variable leak independently. This makes them robust and applicable in a wide range of cases.

Another type of strategy, denoted the model-based attacks, requires the *a priori* on the model to be expressed as an explicit function (a.k.a. the partition function in Chapter 3). A famous example would be the classical Mutual Information Analysis (MIA) [Gie+08]. Some attacks of this category have recently been extended in order to use deep learning techniques (NEMIA or [Tim19; ZLG21]) allowing to take advantage of the multidimensional treatment of the traces. It reduces at the same time the need for preprocessing with for example filtering or realignment techniques. The preceding chapter showed that such attacks can exploit a larger part of the information contained in the traces and may outperform state-of-the-art stochastic attacks. However, they suffer from two major drawbacks:

- **The choice of the partition function** which is closely related to the leakage model *a priori*. Indeed, each bit of the intermediate variable can have very different leakage behavior (especially when it comes to Electro-Magnetic (EM)

measurements) and even sign inversions of their coefficients [CLH20]. In these cases, a classical Hamming weight *a priori* may lead to unsuccessful attacks where an LRA would work.

- **The time complexity of the attack** which requires as many network trainings as there are key hypotheses (which means 256 trainings when attacking a key byte). If the adversary wants to test multiple *a priori*, this issue is then amplified which leads to complex or even unpractical attacks for highly noisy target requiring a lot of traces and therefore long trainings.

### 4.1.2 Contributions

This chapter presents a new unsupervised strategy, denoted the EVIL Machine Attack (EMA), which allows to use a flexible *a priori* (as in the stochastic attack) while still being able to exploit the power of deep learning. We show that it only requires one network training whatever the number of key hypotheses. Therefore, EMA overcomes the two main issues of unsupervised deep learning attacks.

The architecture of the EVIL machine is presented in section 4.2. It is a Generative Adversarial Network (GAN) like structure whose goal is to derive the leakage model of the target under a key assumption. The actual attack and how the number of network training is reduced to only one is described in section 4.3. Results on synthetic and real traces are also presented in this section. Eventually, section 4.4 gives an introduction to a higher-order version of the attack, when dealing with masked implementation.

## 4.2 Learning a Leakage Model Representation

As shown in chapter 3, one of the main difficulties of the model-based SCA is to have a sound representation of the leakage model to use it as a partition function. The original idea of this work is to transfer the task of finding such a representation to a neural network without having to use any *a priori*. This representation is conditioned by a key assumption otherwise it would provide an a priori free strategy contradicting [WOS14]. So to ease the reading of the chapter, the correct key is first assumed to be known in this section. However, section 4.3 shows how such a tool can be turned into an actual unsupervised attack.

### 4.2.1 Notations and SCA framework

We recall here the SCA notations used in this chapter. In order to gain information on the secret key the adversary targets the manipulation of a sensitive variable $Z \in \mathcal{Z} = \mathbb{F}_2^n$, for a given $n \in \mathbb{N}$. This variable is supposed to functionally depends on a public variable $X \in \mathcal{X} = \mathbb{F}_2^m$, for a given $m \in \mathbb{N}$, and a secret key $k^* \in \mathcal{K} = \mathbb{F}_2^m$ through the relation: $Z = g(X, k^*)$ where $g : \mathcal{X} \times \mathcal{K} \to \mathcal{Z}$ is a known function

depending on the underlying cryptographic algorithm. If an hypothesis $k$ is made on the secret key, one can define $Z_k = g(X, k)$ such that $Z = Z_{k^*}$.

For a fixed $k$ we denote by $g_k$ the $g(\cdot, k)$ function. All the $g_k$ are supposed to be bijective in this chapter of reciprocal $g_k^{-1}$ (for example, in the classical first round AES case one would have $g_k(X) = \texttt{SBOX}[X \oplus k]$ of reciprocal $g_k^{-1}(Z) = \texttt{SBOX}^{-1}[Z \oplus k]$. Eventually, traces $L$ leaking information about $Z$ through a leakage model $\varphi$ can be expressed as $L = \varphi(Z) + \mathcal{E}$ where $\mathcal{E}$ represent an independent noise variable.

### 4.2.2 Building the Network's Architecture

Classical Mutual Information (MI) attacks rank the key hypotheses with the following distinguisher:

$$\mathcal{D}(k) = \mathcal{I}\big(f(Z_k), L\big) \tag{4.1}$$

where $\mathcal{I}(X, Y)$ stands for the MI between $X$ and $Y$ and $f$ is the partition function transforming the guessed intermediate variable. The starting point of the reasoning behind the EVIL machine is the Theorem 4 (the main theoretical result of the Chapter 3) which states that the leakage model $\varphi$ belong to the set of optimal partition functions as recalled hereafter:

**Theorem 1.** *(Leakage model optimality)*

$$\varphi \in \mathcal{F}_{opt} = \underset{f:\, \mathcal{Z} \to \mathbb{R}^p}{\arg\max} \left\{ \mathcal{I}\big(f(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(f(Z_k), L\big)\big] \right\} \tag{4.2}$$

Note that traces can be multidimensional. Therefore the output domain of $\varphi$ is $\mathbb{R}^p$ where $p$ can be any positive integer.

Our goal is to derive from this theorem a neural network architecture able to produce an encoding function $\mathbf{E} : \mathcal{Z} \to \mathbb{R}$ of the leakage model $\varphi$. The main challenge is to define the network's objective function. Optimally, $\mathbf{E}$ should carry the same information as $\varphi$ and therefore be a bijection of $\varphi$ encoding it into one dimension[1]. Combined with Equation 4.2, this property could be derived as an objective for E. Indeed, since bijective transformations do not affect the MI:

$$\mathbf{E} \in \mathcal{B}(\varphi) \implies \mathbf{E} \in \mathcal{F}_{opt} \tag{4.3}$$

where $\mathcal{B}(\varphi)$ stands for the set of all bijections of $\varphi$. The reciprocal of Equation 4.3 is not formally proven even though we conjecture so. Thus, we make the hypothesis here that a function belonging to $\mathcal{F}_{opt}$ would give a valuable representation of the leakage model and try to find such a function thanks to deep learning tools. A naive

[1] One could design an encoding function encoding the leakage model in more than one dimension. However, preliminary analyses did not show any real value of increasing the output dimension. In addition, since one-dimensional data are easier to interpret and better suited for the attack phase presented in section 4.3, we only focus on functions with a one-dimensional output in this work.

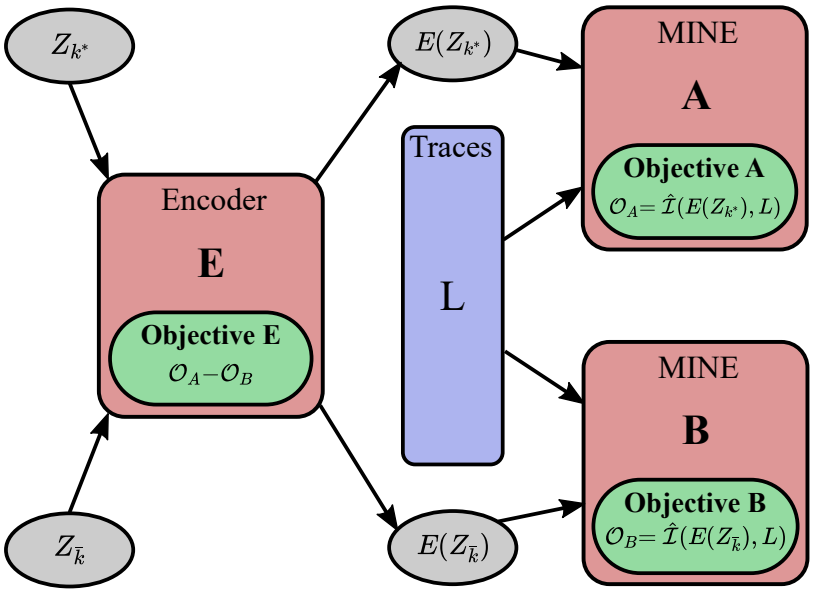



FIGURE 4.1: The EVIL Machine Architecture

idea would be to use the difference of the MI term in Equation 4.2 as an objective function for E. However, such an expression can hardly be used in a deep learning framework.

The first problem is the presence of the max function which is not differentiable. This problem can be solved using Remark 4 which states that the theorem is still valid if one fixes $\bar{k} \in \mathcal{K} \setminus k^*$. So $\forall \bar{k} \in \mathcal{K} \setminus k^*$:

$$\varphi \in \mathcal{F}_{opt} = \underset{f:\, \mathcal{Z} \to \mathbb{R}^p}{\arg\max} \left\{ \mathcal{I}\big(f(Z_{k^*}), L\big) - \mathcal{I}\big(f(Z_{\bar{k}}), L\big) \right\} \tag{4.4}$$

The second problem is the computation of the MI terms which are known to be hard, especially for high-dimensional traces. However, we showed in Chapter 2 that these MI terms could be estimated using deep learning tool and more precisely a Mutual Information Neural Estimator (MINE) [Bel+18a]. Therefore such a tools could be incorporated into the EVIL machine architecture in order to compute the objective function of the encoder **E**. In this chapter it is enough to see MINE as a black-box deep learning method that defines a network, taking as input the traces and label variable, with an objective function converging toward the MI between these two variables. Further details on the theory and implementation in an SCA context can be found in Chapter 2.

**Architecture.** The architecture of the EVIL Machine is depicted in Figure 4.1. It is composed of three neural networks interacting with each other:

- An encoder **E** which takes as input a label (for instance, the value of $Z_{k^*}$ or $Z_{\bar{k}}$) and outputs a value in $\mathbb{R}$ (its final layer is a single neuron).
- MINE **A** which objective function produces an estimation $\hat{\mathcal{I}}(E(Z_{k^*}), L)$ of $\mathcal{I}(\mathbf{E}(Z_{k^*}), L)$.

- MINE **B** which objective function produces an estimation $\hat{\mathcal{I}}(E(Z_{\bar{k}}), L)$ of $\mathcal{I}(\mathbf{E}(Z_{\bar{k}}), L)$, where $\bar{k}$ is fixed to any value in $\mathcal{K} \setminus k^*$.

The encoder **E** is applied on both $Z_{k^*}$ and $Z_{\bar{k}}$ where $\bar{k}$ can be fixed to any value except $k^*$. Then $\mathbf{E}(Z_{k^*})$ and $\mathbf{E}(Z_{\bar{k}})$ are both concatenated with the traces (each trace is concatenated with its associated encoded label) and provided to the MINE networks which estimate both MI terms of Equation 4.4. Their difference is used as an objective function for the encoder. If all the objective functions are correctly optimized (with usual deep learning techniques), **E** should converge toward an element of $\mathcal{F}_{opt}$, potentially close to a bijection of $\varphi$. In theory, on entire execution of MINE A and B should be run after each epoch of **E** which would be very long. So we decided to mimic the idea found in the field of GANs [Goo+14] and run successively one epoch of **E** and one epoch of A and B (in parallel) hoping that this strategy accelerates the convergence without worsening the results.

### 4.2.3 Simulation Experiments

We have implemented the generic architecture described in the previous section using the TensorFlow library [al.15]. The precise architecture of each network is depicted in Appendix C. In order to validate the methodology and gain intuition about the network behavior, this section provides simulation experiments on synthetic traces generated with different leakage models. The idea is to observe the evolution of the output of **E** over the training epochs and to compare it to the known underlying leakage model to assess if that is converging toward a bijection of $\varphi$.

#### Hamming Weight Leakage Model

For the first experiment, we have generated the most basic side-channel traces composed of one sample leaking a noised version of the Hamming weight of $Z_{k^*} = SBOX[P \oplus k^*]$, with P and $k^*$ respectively representing a plaintext and a key byte. The leakage traces can then be expressed as:

$$L = \mathrm{HW}(Z_{k^*}) + \mathcal{N}(0, 1) \tag{4.5}$$

with $\mathcal{N}(0, 1)$ representing the standard normal distribution. The evolution of the output of **E** is depicted in Figure 4.2. For each epoch, we plot $\mathbf{E}(z)$ for all $z \in \mathcal{Z}$. Each of these 256 values are colored according to their Hamming weight. The first observation is that the encoder is learning to cluster the different values of $Z$ according to their Hamming weight. It thus correctly learns a bijection of the true leakage model. It should be noted that **E** clusters the Hamming weight classes in order which was not guaranteed by the mathematical analysis. The latter result being robust over multiple simulations, we conjectured that such a representation is "simpler" in a way and naturally emerges from the gradient descent algorithm.

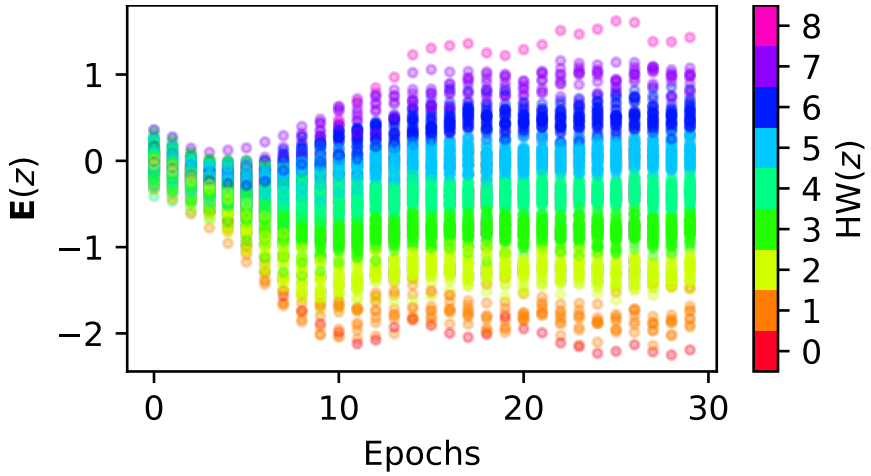



FIGURE 4.2: Evolution of the encoder's output: $\mathbf{E}(z), \forall z \in \mathcal{Z}$ versus epochs (Hamming weight leakage model).

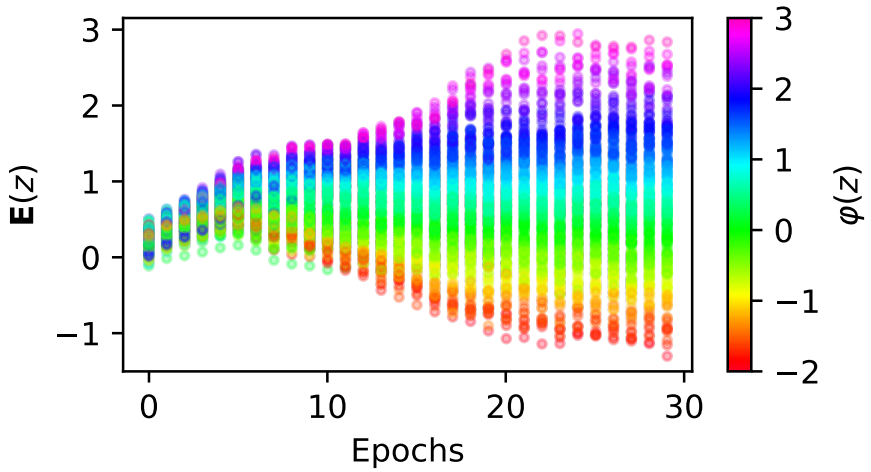


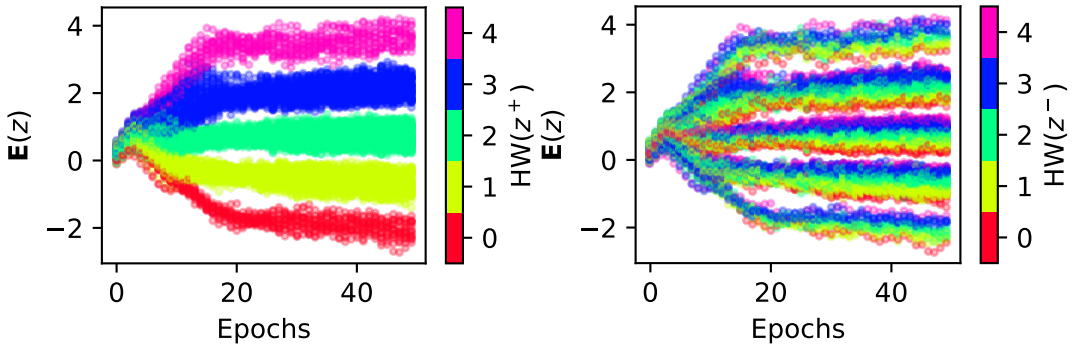

FIGURE 4.3: Evolution of the encoder's output: $\mathrm{E}(z), \forall z \in \mathcal{Z}$ versus epochs (linear leakage model).

### Linear Leakage Model

To assess the EVIL machine capabilities on a more generic leakage model, we repeated the previous experiment emulating a linear leakage with respect to the bits of the sensitive variable. In this case, the leakage traces can then be expressed as:

$$L = \varphi_0(Z_{k^*}) + \mathcal{N}(0,1) \tag{4.6}$$

with $\varphi_0 = \sum_1^8 \alpha_i b_i$ where $b_i$ represents the projection on the $i^{th}$ bit and $\alpha_i$ a random coefficient uniformly drawn from $[-1,1]$. Results are depicted in Figure 4.3. Each point is colored according to the value of $\varphi_0(z)$. The encoder is again converging towards a meaningful representation of the leakage model. Indeed, the figure looks like a uniform rainbow which highlights the fact that the relation between $\mathbf{E}(Z_{k^*})$ and $\varphi_0(Z_{k^*})$ is quasi-linear.




(A) Colored according to $HW(Z_{k^*}^+)$ (B) Colored according to $HW(Z_{k^*}^-)$

FIGURE 4.4: Evolution of the encoder's output: $\mathrm{E}(z), \forall z \in \mathcal{Z}$ versus epochs (multidimensional leakage model).

### Multidimensional Leakage Model

One of the main advantages of the EVIL machine is its ability to take large parts of the trace as input and to compress int one neuron multiple leakage sources. To highlight this multidimensional capability we design a simple example where the leakage is split on two samples. Each sample leaks respectively the Hamming weight of $Z_{k^*}^+$ and $Z_{k^*}^-$ which represent the four most and least significant bits of $Z_{k^*}$. The leakage traces can then be expressed as:

$$L = [\mathrm{HW}(Z_{k^*}^+) + \mathcal{N}(0,1),\ \mathrm{HW}(Z_{k^*}^-) + \mathcal{N}(0,1)] \tag{4.7}$$

Results are presented in Figure 4.4. The two plots are the same but colored differently. Figure 4.4a is colored according to $\mathrm{HW}(Z_{k^*}^+)$ while Figure 4.4b is colored according to $\mathrm{HW}(Z_{k^*}^-)$. It appears that the encoder uses the macro structure (the five big branches) to encode the information on $Z_{k^*}^+$ and the micro structure (the position in the branch) to encode the information on $Z_{k^*}^-$. Therefore, the encoder successfully learned a compressed version of a multidimensional leakage model. In this case it could almost be expressed as linear combination of both leakage sources: $\mathbf{E}(z) \approx \alpha \mathrm{HW}(Z_{k^*}^+) + \beta \mathrm{HW}(Z_{k^*}^-)$, for some $\alpha, \beta \in \mathbb{R}$.

### Non-Linear Leakage Model

Finally, to show that the EVIL machine is not limited to linear leakage models, we present an experiment with a non-linear model $\varphi_{NL}$. We used a linear leakage model $\varphi_0$ on the 7 MSBs of $Z_{k^*}$ where the sign of the leakage is determined by the value of the LSB: $\varphi_{NL} = (-1)^{b_8} * \varphi_0$ with $\varphi_0 = \sum_1^7 \alpha_i b_i$, where $\alpha_i$ are random coefficient uniformly drawn from $[0,1]$. The leakage traces can then be expressed as:

$$L = \varphi_{NL}(Z_{k^*}) + \mathcal{N}(0,1) \tag{4.8}$$

Results are depicted in Figure 4.5. Each point is colored according to the value of $\varphi_0(z)$. In this case, it appears that the encoder learned a representation of the leakage

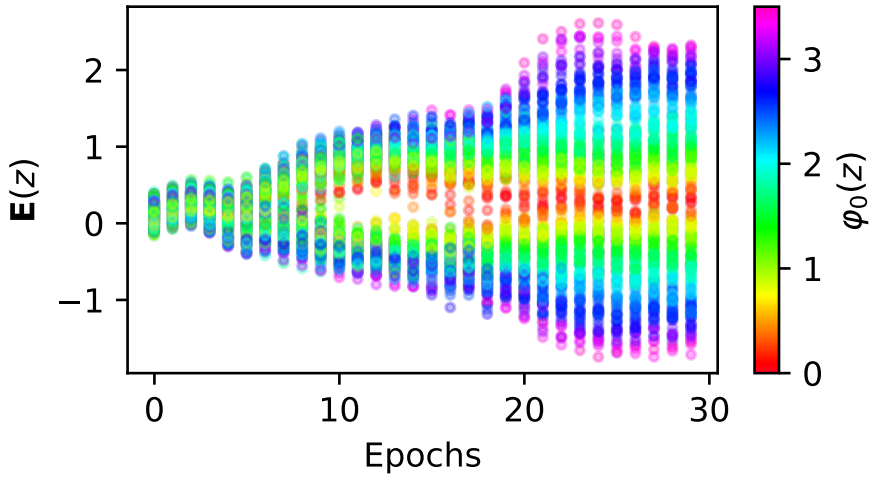



FIGURE 4.5: Evolution of the encoder's output: $\mathrm{E}(z), \forall z \in \mathcal{Z}$ versus epochs (non-linear leakage model).

model $\varphi_{NL}$ composed of two symmetric branches, each one encoding the linear part of the model $\varphi_0$ but in the opposite direction due to the sign inversion related to the value of the LSB of $Z_{k^*}$. This confirms that the EVIL machine can still learn sound representations of the leakage even if the latter is non-linear.

This section has shown that the EVIL machine could learn a sound representation of the leakage model. Such a tool can be useful in itself, for example, for designers to gain intuition on the leakage of their devices. It does require the knowledge of the key to work, however, it is possible to use key guesses as explained in the next section which aims at turning this tool into an unsupervised attack.

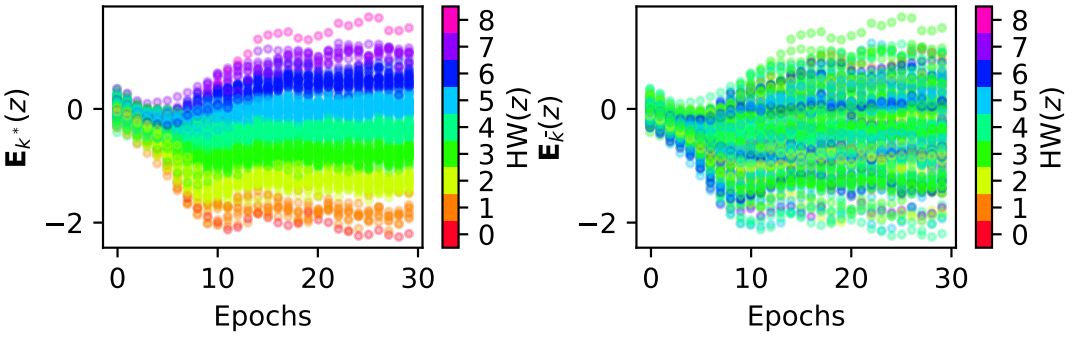
## 4.3 The EVIL Machine Attack

The EVIL machine presented in the previous section produces a representation of the leakage model assuming that the correct key is known. However, making hypotheses on the key allows to convert this tool into a key recovery strategy, denoted the EVIL Machine Attack (EMA). For any $k \in \mathcal{K}$ one can run an iteration of the EVIL machine producing a representation $\mathbf{E}_k$ of the leakage model $\varphi_k$ representing the leakage model under the assumption that the correct key is $k$. All of these models $\varphi_k$ mathematically exist. They are functions representing a hypothetical leakage model that would be the real one if the processed variable was the wrongly guessed $Z_k = g(X, k)$ instead of $Z_{k^*}$. Thus, they satisfy the following property:

$$\varphi_k(Z_k) = \varphi_{k^*}(Z_{k^*}) \tag{4.9}$$

They are therefore, very likely, complicated functions (with a high algebraic degree) due to cryptographic properties of the $g_k$ functions[2] linking $Z_k$ and $Z_{k^*}$ together

[2] Cryptographic algorithms should embed highly non-linear transformations to avoid algebraic attacks and we assume here that the targeted sensitive variable has undergone this non-linear transformation, for example, $Z = \texttt{SBOX}(X \oplus k^*)$.




(A) Correct key (B) Wrong key

FIGURE 4.6: Evolution of the encoders' output for the correct and a wrong key guess.

by the relation: $Z_k = g_k(g_{k^*}^{-1}(Z_{k^*}))$. The main idea of EMA is then to choose among all the possible representations of the leakage model $\mathbf{E}_k$, the simplest one, implementing a form of Occam's razor principle. This choosing step involves physical knowledge on what a leakage model should look like and this is where the adversary's *a priori* knowledge comes into play.

To illustrate the point, we repeated the Hamming weight leakage model experiment of section 4.2 but with a wrong key guess $k_0$. The evolution of $\mathbf{E}_{k_0}$ and $\mathbf{E}_{k^*}$, the one obtained with the correct key guess, are plotted in Figure 4.6 in order to compare them. One can visually note the difference: $\mathbf{E}_{k^*}$ is related to the Hamming weight function while $\mathbf{E}_{\bar{k}}$ seems unstructured and close to a random output. Such property can be used to define a distinguisher between key guesses as explained in subsection 4.3.2.

### 4.3.1 One Training to Rule them All

Obtaining all the $\mathbf{E}_k$ can be done by running $|\mathcal{K}|$ times the EVIL machine with different key assumptions. However, this may be very long, especially for low-information scenarios requiring a lot of traces which are precisely the use cases considered in this work. This section shows that the adversary can in fact make a first guess $k_0$, which may be wrong, compute $\mathbf{E}_{k_0}$, and mathematically derive $\mathbf{E}_k$ for any $k \in \mathcal{K}$ from $\mathbf{E}_{k_0}$. This method allows running only once the EVIL machine to rule all cases.

As explained in the previous section $\mathbf{E}_{k_0}$ gives a representation of $\varphi_{k_0}$, the leakage model if the correct key was $k_0$. For any $k \in \mathcal{K}$ Equation 4.9 gives:

$$\varphi_k(Z_k) = \varphi_{k^*}(Z_{k^*}) \tag{4.10}$$

which is also true for $k_0$:

$$\varphi_{k_0}(Z_{k_0}) = \varphi_{k^*}(Z_{k^*}) \tag{4.11}$$

So one can deduce from Equations 4.10 and 4.11 that:

$$\begin{aligned} \varphi_k(Z_k) &= \varphi_{k_0}(Z_{k_0}) \\ \varphi_k\big(g_{k_0}(g_k^{-1}(Z_k))\big) &= \varphi_{k_0}\big(g_{k_0}(g_k^{-1}(Z_{k_0}))\big) \\ \varphi_k(Z_{k_0}) &= \varphi_{k_0}\big(g_{k_0}(g_k^{-1}(Z_{k_0}))\big) \end{aligned} \tag{4.12}$$

which completely define $\varphi_k$ from $\varphi_{k_0}$. Since $\mathbf{E}_k$ stands for a representation of $\varphi_k$, one has by analogy:

$$\mathbf{E}_k(Z_{k_0}) = \mathbf{E}_{k_0}\big(g_{k_0}(g_k^{-1}(Z_{k_0}))\big) \tag{4.13}$$

And since $Z_{k_0}$ and $Z$ live in the same set $\mathcal{Z}$, one can simplify the previous equation:

$$\mathbf{E}_k(Z) = \mathbf{E}_{k_0}\big(g_{k_0}(g_k^{-1}(Z))\big) \tag{4.14}$$

So one can quickly and easily deduce any $\mathbf{E}_k$ including $\mathbf{E}_{k^*}$ from the only knowledge of $\mathbf{E}_{k_0}$ with $k_0$ arbitrary chosen by the adversary. This reduces by a factor $|\mathcal{K}|$ (when targeting a key byte, $|\mathcal{K}| = 256$) the time complexity of the attack.

**Example.** *In the AES case with $g_k(P) = \texttt{SBOX}[P \oplus k]$ where $P$ represents a plaintext byte, one can deduce $\boldsymbol{E}_k$ from $\boldsymbol{E}_{k_0}$ with the following formula:*

$$\boldsymbol{E}_k(Z) = \boldsymbol{E}_{k_0}\big(\texttt{SBOX}\big[\texttt{SBOX}^{-1}[Z \oplus k] \oplus k_0\big]\big) \tag{4.15}$$

### 4.3.2 About the Distinguisher

In a real attack scenario, one does not know the true leakage model so it is impossible to use the visualization technique used in the previous section (where each sample is colored according to the true leakage model) to discriminate between the key candidates. Therefore, one would need a distinguisher $\mathcal{D}$ scoring each leakage model representation $\mathbf{E}_k$ to rank the key hypotheses. The intuition behind EMA is that $\mathcal{D}$ should give a score reflecting the plausibility of the leakage representation $\mathbf{E}_k$ according to physical *a priori* on the leakage model and on which properties the latter should follow.

#### Assumption on the Degree of $\mathbf{E}_{k^*}$

We herein give a proposal of distinguisher for the common *a priori* stating that the leakage model $\varphi_{k^*}$ should be a function from $\mathcal{Z} = \mathbb{F}_{2^n} \to \mathbb{R}^p$ such that the $p$ coordinate functions have a low algebraic degree (bonded by a degree $d$). This is the multivariate version of assumption 3 (Leakage Interpolation Degree) followed in [Dog+12] for the setup of the well-known LRA. For example, a degree $d = 1$ traduces the fact that, for each time sample, all the bits of the processed variable leak independently. We propose to extend the assumption on the degree of $\varphi_{k^*}$ (precisely of its coordinate functions) to the degree of its representation $\mathbf{E}_{k^*}$, produced by the EVIL machine.

### Distinguisher

If this assumption is true, then one can perform a polynomial regression of order $d$ to find the polynomial $p_k$ of degree $d$ minimizing the quadratic error $||\mathbf{E}_k(Z_k) - p_k(Z_k)||^2$ for all $k$. As explained in [Dog+12], such a regression can be seen as a linear regression by choosing an appropriate basis and is easily solvable using the least square method. A measure of fitness such as the coefficient of determination $R^2$ is then used as a score for each key:

$$\mathcal{D}(k) = 1 - \frac{||\mathbf{E}_k(Z_k) - p_k(Z_k)||^2}{\mathrm{var}\big(\mathbf{E}_k(Z_k)\big)} \tag{4.16}$$

Since the assumption on the degree of $\mathbf{E}_k$ is true only for $k = k^*$, the correct key should have the highest score leading to a successful attack.

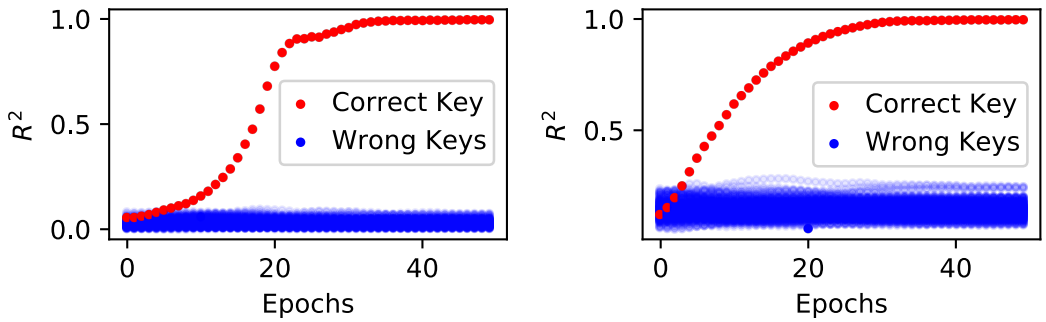
### Intuition Behind the Assumption

The justification of the assumption on the degree of $\mathbf{E}_{k^*}$ is empirical. Indeed two functions in bijection do not necessarily have the same algebraic degree. However, we observed in the experiments of section 4.2 that the encoder naturally converges towards a "smooth" representation of the leakage model. Our intuition is that since neural networks are composed of a succession of continuous functions it is easier for it to map two values $z_1$ and $z_2$ that have a close leakage model ($\varphi_{k^*}(z_1)$ close to $\varphi_{k^*}(z_2)$), to close encoding values $\mathbf{E}_{k^*}(z_1)$ and $\mathbf{E}_{k^*}(z_2)$. Such continuous representation has more chance of preserving the degree. In addition, in Figure 4.4, it seems that when dealing with multidimensional leakages, the encoder produces a linear combination of the representation of univariate leakages. Such additive behavior would also preserve the degree since by hypothesis all the coordinate functions are of degree less or equal than $d$.

### Experiments Supporting the Assumption

One may argue that the intuitions shared in the previous paragraph come from simple experiments and may not scale with high dimensional traces containing much more leaking samples as well as uninformative samples. Therefore we designed two experiments to assess the assumption in these harder cases. We have generated 10000 synthetic traces of dimension 1000 for both experiments. They contains 100 informative samples randomly distributed in the traces leaking information about $Z = \texttt{Sbox}[P \oplus k^*]$ where $P$ represents a plaintext byte uniformly drawn from $[\![0, 255]\!]$. In the first experiments, a linear leakage model is assigned to each sample, thus, the leakage $L$ of each sample can be expressed as:

$$L = \alpha_0 + \sum_{i=1}^{8} \alpha_i \cdot bit_i(Z) + \mathcal{N}(0,1) \tag{4.17}$$




(A) Linear leakages (B) Quadratic leakages

FIGURE 4.7: Evolution of the distinguisher $\mathcal{D}(k)$ versus epochs for all the key candidates.

where $bit_i(Z)$ stands for the $i^{th}$ bit of $Z$. The $\alpha_i$ are random coefficient uniformly drawn from $[-1, 1]$. These coefficients are redrawn for each leaking sample. The second experiment emulates quadratic leakages (with no linear coefficients) such that for each sample:

$$L = \alpha_0 + \sum_{i=1}^{8} \sum_{j=i+1}^{8} \alpha_{i,j} \cdot bit_i(Z) \cdot bit_j(Z) + \mathcal{N}(0,1) \tag{4.18}$$

In both experiments, the 900 non-informative samples follow a normal distribution $\mathcal{N}(0,1)$. We ran the EVIL machine assuming $k_0 \neq k^*$ and derived all the $\mathbf{E}_k$ as explained in subsection 4.3.1. Figure 4.7 reports the evolution of $\mathcal{D}(k)$ for each $k$ computed with linear and quadratic regression respectively for the first and second experiments. In both cases, the coefficient of determination converges toward 1 for the correct key. This means that after some epochs, $\mathbf{E}_{k^*}$ can be perfectly regressed by a linear/quadratic model. These two results support the assumption on the degree of $\mathbf{E}_{k^*}$ regarding the maximum degree of the coordinate function of the leakage model $\varphi_{k^*}$. In addition, this is not at all the case of $\mathbf{E}_k$ for $k \neq k^*$ leading to a high distinguishability between the correct and the wrong keys.

### 4.3.3 Attack Description

The steps required to perform the EMA are summarized hereafter.

1. Choose any $k_0$ and run the EVIL machine for $n_e$ epochs with $k_0$ as a key assumption. This gives a representation $\mathbf{E}_{k_0}^{(e)}$ for each epoch $e$.

2. Derive $\mathbf{E}_k^{(e)}$ for all $e$ and $k$ using the formula:

$$\mathbf{E}_k^{(e)}(Z_{k_0}) = \mathbf{E}_{k_0}^{(e)}\left(g_{k_0}(g_k^{-1}(Z_{k_0}))\right) \tag{4.19}$$

3. Compute for all $e$ and $k$ the distinguishing score:

$$\mathcal{D}^{(e)}(k) = 1 - \frac{||\mathbf{E}_k^{(e)}(Z_k) - p_k(Z_k)||^2}{\mathrm{var}\big(\mathbf{E}_k^{(e)}(Z_k)\big)} \tag{4.20}$$

4. Derive a single score for each key:

$$\mathcal{D}(k) = \max_{n_s \leq e \leq n_e} \mathcal{D}^{(e)}(k) \tag{4.21}$$

5. Rank the keys according to their distinguishing score.

The $n_s$ and $n_e$ parameters are hyper-parameters set by the adversary. The purpose of $n_s$ is to not take into account potential early fluctuation where the networks might be not stable yet. This turned out to be useful for very noisy situations (for our experiments, we used $n_s = 50$ and $n_e = 100$).

### 4.3.4 Experimental Results

EMA uses the classical LRA distinguisher on the encoder's output whose purpose is to encapsulate all the informative components of the traces. The classical LRA computes the regression directly on the traces in a univariate way (selecting the maximum score along all samples as a distinguisher). This section provides simulations and real case experiments aiming at comparing both attacks in order to assess to what extent the deep learning step is valuable. Both attacks are unsupervised and require very limited knowledge on the leakage model of the target. The main advantage of EMA over LRA is the multidimensional treatment of the traces. The latter offers the capability to potentially exploit multiple leakage sources at the same time while reducing the need of preprocessing techniques thanks to the power of DL techniques. For example, convolutional layers may handle desynchronized traces [CDP17]. To confirm that these advantages translate into objective improvements in terms of noise resilience or number of traces required, we present hereafter two experiments.

#### Experiments on Synthetic traces

For the first experiment, we generated multidimensional synthetic traces subject to desynchronization. These traces contains 20 informative samples, leaking information about $Z = \texttt{Sbox}[P \oplus k^*]$ trough linear leakage models, with randomly chosen coefficients, to which we added Gaussian noise with a standard $\sigma$. Traces contains also 80 uninformative samples and are artificially desynchronized. Each trace is shifted by a random number *sh* uniformly drawn from $[\![0, 50]\!]$. The precise generation procedure is depicted in algorithm 3. We ran the classical LRA and the EMA following the procedure explained in subsection 4.3.3 for each value of $0 \leq \sigma \leq 25$. Results are presented in Figure 4.8a. Each point represents the average rank of the

**Algorithm 3:** Generate Traces

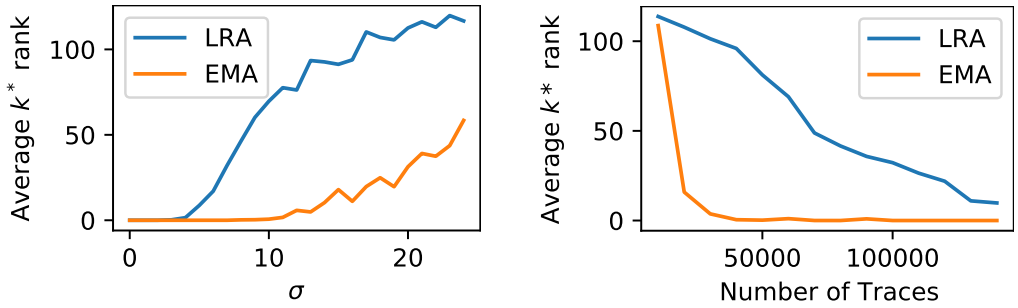
**Input:** $\sigma$
**Output:** $L$, *a* $(10k, 100)$ *array*
**Output:** $P$, *a* $(10k)$ *array*
$P \leftarrow$ Draw 10k plaintext uniformly from $[\![0, 255]\!]$
$Z \leftarrow \texttt{Sbox}[P \oplus k^*]$
$L \leftarrow$ Draw a (10k, 100) array from a Gaussian $\mathcal{N}(0, \sigma^2)$
$R \leftarrow$ Draw a (20, 8) array of random coefficients from $\mathcal{U}_{[-1,1]}$
**for** $1 \leq i \leq 10k$ **do**
  **for** $1 \leq j \leq 20$ `// Add leakage one every 5 samples`
  **do**
    $L[i, 5 * j] \leftarrow L[i, 5 * j] + \sum_{l=1}^{8} R[j, l] \cdot bit_l(Z[i])$
  **end**
**end**
**for** $1 \leq i \leq 10k$ **do**
  $sh \leftarrow$ Draw a random integer uniformly from $[\![0, 50]\!]$
  $L[i] \leftarrow$ Roll$(L[i], sh)$ `// Apply the jitter (Roll is a function shifting the array by` $sh$`, in a looping way)`
**end**
**return** L, P

correct key computed over 100 independent experiments realized with a given value of $\sigma$. It appears that the EMA is way more resilient to the noise added to the traces resulting in successful attacks where the LRA fails. We highlight the fact that with such leakage models, (linear model with random coefficient from $[-1, 1]$) it would be hard to run an effective model-based attack since it would require an explicit model assumption.

### Experiments on Real Traces

In order to validate our methodology on a real case scenario, we have acquired one million AES traces from a cortex A7-based System-on-Chip (SoC) running a Linux OS. The AES implementation comes from the OpenSSL library [The03]. We chose to run the analysis on a very noisy SoC subject to desynchronization, thus, offering an interesting case. Traces[3] are composed of 1500 samples corresponding to the execution of the first round `Sbox`. Guessing entropies of LRA and EMA are presented in Figure 4.8b. Each point represents the average rank of the correct key computed over 100 experiments in which traces are chosen randomly from the dataset. It appears that the EMA converges toward the correct key faster than the LRA ranking it always at the first position with 40k traces where the LRA does not with 150k traces.

[3] For reproducibility reasons and potential use in the SCA community, the dataset with the associated labels is accessible on this link: `removed for blind reviews`.




(A) Synthetic traces

(B) Real traces (A7 based SoC)

FIGURE 4.8: Guessing entropies of EMA and LRA.

This confirms that the EMA may be worth considering in very noisy scenarios, offering an unsupervised deep learning-based attack requiring only one network training and very little knowledge on the device leakage model.

## 4.4 Introduction to Higher-Order Generalization

One of the most widely used countermeasures to prevent instantaneous leakages and mitigate first-order SCA is to mask the sensitive data using secret sharing techniques [Cha+99]. The idea is to split each sensitive intermediate value $z$, into $d$ shares: $(z_i)_{1\leq i\leq d}$. The $d-1$ shares $z_2, ..., z_d$ are randomly chosen and the last one, $z_1$ is processed such that:

$$z_1 = z * z_2 * \cdots * z_d \tag{4.22}$$

for a group operation $*$ of $\mathcal{Z}$. This led to the emergence of the so called higher-order attacks which combine multiple samples from the same traces leaking information from each share to deduce information on $Z$. The EVIL machine is, by nature, multidimensional and may therefore be extended to masked implementations. Indeed, we have shown in Chapter 2 that MINE can automatically recombine samples to find information in masked cases. It would therefore be possible to run straightforwardly the EVIL machine on masked implementations. However, the main theoretical problem is that in masked case the leakage model cannot be expressed as a deterministic function $L = \varphi(Z)$ with some noise. Thus, there is no equivalent of Equation 4.2 for masked cases. A natural question is then to ask towards what kind of representation the encoder typically converge.

### 4.4.1 Encoder's Output and Joint Moments

When dealing with masked implementations, the leakage variable is not separable into a deterministic and a noise part anymore. Instead, traces correspond to realizations of a leakage variable $L$ coming from a stochastic process $\mathcal{S}$, $Z \xrightarrow{\mathcal{S}} L$ which encapsulates the random choice of the masks and the leakage model of each share.

$L$ follows a multivariate probability distribution which, as any distribution, is completely determined by the list of all its joint moments. First-order SCA exploit information in the first-order moment: the leakage model being the mean per class:

$$\varphi(z) = \mathbb{E}[L \mid Z = z] \tag{4.23}$$

For a masked implementation the discriminating information, if it exists, is necessarily hidden in higher-order joint moments since lower-order leakages are prevented by masking. Therefore, the idea of higher-order attacks [PRB09; DDP13; Cri+22] is to exploit the lowest-order[4] centered joint moment containing information, thus replacing Equation 4.23 by the joint moment per class. For a $d$-order masked implementation and $(L_1, \ldots, L_d)$ representing one leakage sample for each share, such a joint moment can be expressed as:

$$jm_L(z) = \mathbb{E}\left[\prod_{i=1}^{d}(L_i - \mu_{L_i} \mid Z = z)\right] \tag{4.24}$$

where $\mu_{L_i}$ stands for the expectation of $L_i$. For example, the classical second-order CPA [PRB09] combines samples with the centered product function which happens to be the covariance, *a.k.a.* the second order joint moment.

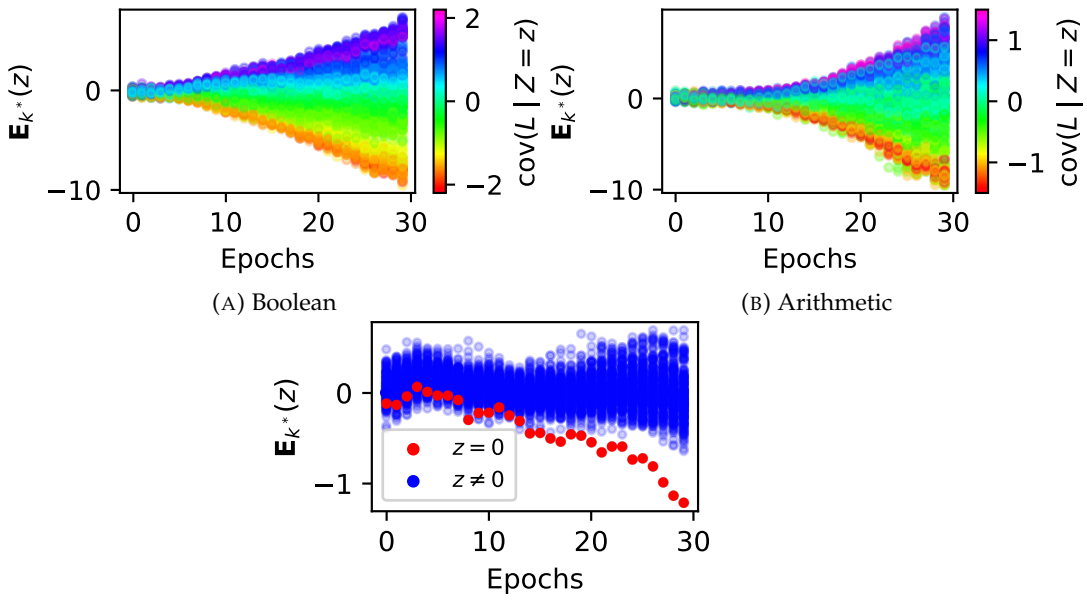
#### Hypothesis on the Encoder's Output

By analogy with the unmasked case where the encoder of the EVIL machine converged towards a representation of $\varphi$ (Equation 4.23) our hypothesis is that, for masked implementations, the encoders will converge towards a representation of $jm_L$ (Equation 4.24). Our intuition is that $jm_L(Z)$ is a sound way to encode $Z$ to keep information between $jm_L(Z)$ and $L$ while limiting the information $\mathcal{I}\big(jm_L(Z_{\bar{k}}), L\big)$ computed with a wrong key candidate $\bar{k}$.

#### Experiment Supporting the Hypothesis

To validate this hypothesis, we present experiments on masked synthetic traces for three different second-order masking schemes: Boolean, arithmetic and multiplicative. The group operations are respectively the xor $\oplus$, the addition over $\mathbb{Z}/n\mathbb{Z}$ $+[n]$ and the multiplication over the Galois field $\otimes$. The multiplication by $m$ being reversible for $m \neq 0$ the multiplicative mask has to be chosen in $\mathbb{F}_{2^n}^*$. In addition, the multiplicative scheme is biased since $Z = 0$ implies $Z \otimes m = 0$ which induces a first-order leakage. For each masking scheme, we have generated 50k traces containing two samples representing the leakage of each share, $Z_1$ and $Z_2$, through a

[4] Higher-order moments being harder to estimate due to noise amplification.




(A) Boolean

(B) Arithmetic

(C) Multiplicative

FIGURE 4.9: Evolution of the encoder's output: $\mathrm{E}(z), \forall z \in \mathcal{Z}$ on masked synthetic traces (linear leakage model of the shares).

linear model. The leakage $L$ can be expressed as:

$$L = \left[ \sum_{i=1}^{8} \alpha_i bit_i(Z_1) + \mathcal{N}(0,1), \sum_{i=1}^{8} \beta_i bit_i(Z_2) + \mathcal{N}(0,1) \right] \tag{4.25}$$

where the $\alpha_i$ and $\beta_i$ coefficients are uniformly drawn from $[-1, 1]$. We have run the EVIL machine as explained in section 4.2 and present the result in Figure 4.9. For the Boolean and arithmetic schemes, each point is colored according to the theoretical covariance per class: $\mathrm{cov}(L \mid Z = z)$ which is computable knowing the $\alpha_i$ and $\beta_i$. It appears that the encoder is converging towards a smooth representation of covariance per class (the second-order joint moment), thus, supporting the hypothesis. For the multiplicative scheme, it seems that the encoder learned to distinguish the class $Z = 0$ from the others, confirming that it is focusing on lower-order leakages, if it exists, in priority.

### About the Distinguisher

Stochastic attacks (with flexible *a priori*) have been extended to masked implementations [DDP13]. Authors presents a higher-order version of the LRA, the HO-LRA, by proposing to perform the LRA on the estimated joint moment per class instead of the raw traces. In Chapter 5, we highlight some limitations of the previous works and propose a new strategy which directly regresses the leakage model of each share with the Joint Moment Regression (JMR) strategy. Both attacks use an estimation of the $jm_L$ function calculated under a key assumption as input to the method. Therefore we argue that these attacks could be conducted on the output of the encoders

of the EVIL machine. Indeed, as suggested in the previous section, $\mathbf{E}_{k^*}(Z)$ could be seen as a representation of the $jm_L$ function under the correct key assumption. The trick introduced in subsection 4.3.1 still apply and all the $\mathbf{E}_k$ can be derived from only one network training.

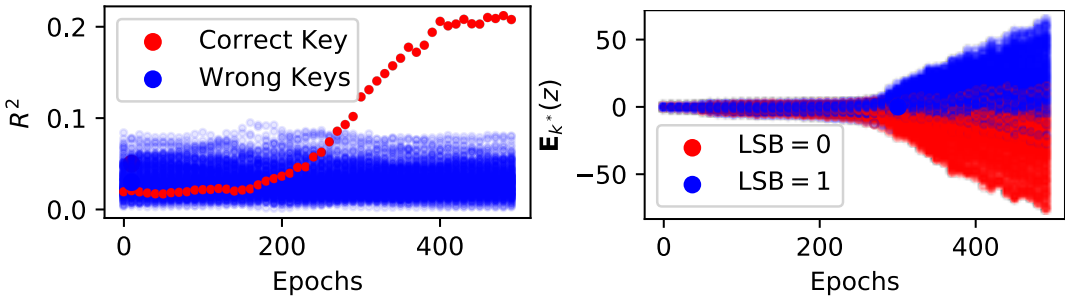
Again, the main advantage of the EMA is that these representations benefit from all the deep learning techniques. In addition, in classical $d$-variate attacks, the adversary should select the $d$ samples from the traces to combine and perform the joint moment estimation. In a complete black-box setting, with no information on the point of interests, the latter has to enumerate all the possible $d$-(upplet) samples and select the best one which can be very long and scale exponentially with $d$. The EMA does not suffer from the same problem since the whole (or at least a big part of) traces can be fed to the network.

### 4.4.2 A Practical Case on ASCAD

As a proof of concept of the higher-order version of the EMA, this section presents an attack on the public ASCAD dataset [Ben+18]. It is a common set of side-channel traces, introduced for research purposes on deep learning-based side-channel attacks. The targeted implementation is a software AES, protected with a first-order Boolean masking, running on an 8-bit ATMega8515 board. Since JMR and the HO-LRA are equivalent for the second-order Boolean case, as shown in the next chapter, we adopt the HO-LRA strategy which is simpler. It just consists in conducting an LRA on the estimated covariance or on $\mathbf{E}_k$ for the EMA case. It is therefore the exact same attack as for the first-order case described in subsection 4.3.3. We ran the EMA on 10k of the training traces which consist of 700 samples targeting the first `Sbox`. Results are presented in Figure 4.10a. The attack is successful with a clear distinguishability for the correct key.

In order to identify which features of the traces has been exploited by the EVIL machine (with the correct key assumption) we plotted the evolution of the encoder's output in Figure 4.10b and colored each point of according to the value of the LSB. Indeed, we showed in Chapter 3 that the higher-order leakage of ASCAD traces is mostly related to the value of the LSB of $Z$. It appears that after 200 epochs (when the attack begins to work) the encoder actually learned to split the two classes related to the LSB illustrating the interpretability of the encoder's output.

To give an order of magnitude, running the full attack (with the 500 epochs) took approximately 5 minutes on a workstation embedding a Tesla V100.




(A) Evolution of the distinguisher $\mathcal{D}(k)$ versus epochs for all the key candidates.

(B) Evolution of the encoder's output: $\mathrm{E}(z), \forall z \in \mathcal{Z}$ colored according to the LSB of $z$.

FIGURE 4.10: Attack results on ASCAD with 10k traces.

## 4.5 Conclusion

This chapter presents a new unsupervised side-channel key recovery strategy denoted the EVIL Machine Attack (EMA). It is built on a GAN-like structure that aims at converging towards a representation of the leakage model of the device under a key assumption. Occam's razor's principle allows to derive from this tool an actual attack enabling the utilization of deep learning's potential while requiring very few knowledge on the leakage model. For example, a simple linear leakage assumption is enough as in a classical LRA. It requires only one network training where classical unsupervised deep learning strategies require as many trainings as key guesses (256 when targeting a key byte). Thus EMA reduces significantly the time complexity while being more flexible in the leakage model *a priori* than unsupervised DL based attack. This may lower the barrier for conducting such kind of attacks in practice. Simulations and real cases experiments suggest that the additional information brought by the multidimensional treatment of traces is beneficial and translates into more data-efficient attacks than the classical LRA. For example, it lower by more than 400% the number of required traces in the real case scenario targeting the openSLL implementation of the AES running on a SoC.

Eventually, the last part of the chapter is dedicated to an introduction of higher-order generalizations of the strategy, replacing the leakage model estimation with the joint moment estimation. It raises the question of how to conduct an optimal attack considering that the adversary is granted an estimation of the joint moments per class of the leakage. The following chapter precisely tackle this question offering a new unsupervised strategy working as a stand alone attack but which combines nicely with EMA to serve as a distinguisher in the higher-order case.

# Chapter 5

# Fit the Joint Moments

## How to Attack any Masking Scheme

*"No one can wear a mask for very long"*

Seneca the Younger

*One of the main countermeasure against SCA is to use secret sharing technique to mask the sensitive variables. Supervised attack can straightforwardly be extended to masked implementations even if the profiling phase may require more traces. However, defeating masking is less trivial when it comes to unsupervised attacks. While classical strategies such as CPA or LRA have been extended to masked implementations, we show in this chapter that these extensions only hold for Boolean and arithmetic schemes. Therefore, we propose a new unsupervised strategy, the Joint Moments Regression (JMR), able to defeat any masking schemes (multiplicative, affine, polynomial, inner product...), which are gaining popularity in real implementations. The starting point of JMR is an estimation of the joint moment per class. In this chapter this estimation is assumed to be computed in a classical way combining samples from traces. However as shown in the previous chapter the EVIL machine could be used to replace this joint moment estimation and benefits from all the deep learning's advantage. The main idea behind JMR is to directly regress the leakage model of the shares by fitting a system based on higher-order joint moments conditions. We show that this idea can be seen as part of a more general framework known as the Generalized Method of Moments (GMM). This offers mathematical foundations on which we rely to derive optimizations of JMR. Simulations results confirm the interest of JMR over state-of-the-art attacks, even in the case of Boolean and arithmetic masking. Eventually, we apply this strategy to real traces and provide, to the best of our knowledge, the first unsupervised attack on the protected AES implementation proposed by the ANSSI for SCA research, which embeds an affine masking and shuffling counter-measures.*

## 5.1 Introduction

### 5.1.1 Context

To prevent instantaneous leakage of the sensitive variables, a classical strategy is to protect implementations using masking techniques. It consists in splitting the internal state of the processing into multiple random shares following secret sharing ideas [BLA79]. SCA against masked implementations is still possible through the so called higher-order attacks which combine multiple leakage samples corresponding to each share. However, these attacks are harder to conduct since the impact of the noise is amplified exponentially with the masking order [PR13]. Among unsupervised attacks, the multivariate CPA described in [PRB09] has often proved to be an efficient strategy in practice. However, it relies on a Hamming weight leakage assumption (of the shares) that may not be correct especially when it comes to local EM measurements. Indeed, each bit of the intermediate variable can have very different leakage behavior and even sign inversions of their coefficients as shown in [CLH20]. To deal with such situations [DDP13] proposed a generalization of the LRA whose main strength is to offer flexibility on the *a priori* without constraining each bit to have the same impact on the leakage.

This method exploits information hidden in the covariance, *i.e.*, the second order joint moment of the distribution since the first order moments (the means) are leakage-free thanks to masking. However, we argue that this method is not generic enough because it is based on the assumption that the covariance per class could be expressed as a low algebraic function, assumption that only holds for Boolean and low order arithmetic masking as shown in this chapter. Indeed, the proposed attack fails even in theory (on synthetic traces with zero noise) when dealing with other masking schemes such as the multiplicative or affine ones. These masking schemes along with the polynomial and inner product masking are getting more and more studied recently and begin to be used in modern implementations. This trend may continue in the future since these schemes seem to offer better resistance against side-channel attacks [Bal+12]. Mutual information-based attacks have also been extended to masked implementations but have not either proven to be valid strategies for any kind of masking and their applicability is mainly related to the open questions, raised in Chapter 3, about the choice of the partitioning function. This leads us to the following observation:

*To the best of our knowledge, no generic unsupervised strategy able to defeat any kind of masking outside of the Hamming weight leakage assumption emerges from the state-of-the-art.*

We propose such a strategy in this chapter: the Joint Moment Regression (JMR).

The latter is built on the idea that the discriminating information, if it exists, is necessarily hidden in higher-order joint moments since lower-order leakages are prevented by masking (at least when not considering glitches from the physical implementation [MPG05]). Intuitively, joint moments encapsulate information about the corresponding distribution. The idea is to make a leakage assumption on each share (for example a linear leakage) and try to directly regress the leakage model of each share, using joint moments conditions, instead of trying to regress the joint moment itself as it is done in [DDP13]. This comes at the cost of the loss of linearity since the joint moment conditions involve a multiplication between the leakage parameters of the different shares which gives rise to a non-linear system of equations. However, we show that numerical optimization algorithms can be used to find an estimation of the solution that best fits the conditions. A measure of fitness is used as a distinguisher between key candidates. The joint moment conditions depend on the underlying masking scheme which allows to embed knowledge of the latter into the system and, therefore, makes the attack generic.

### 5.1.2 Contributions

- The first contribution of the chapter is to present the state-of-the-art on the stochastic higher-order attacks, especially focusing on the method proposed in [DDP13] to understand its strengths and limitations. This analysis can be found in section 5.2.
- As a second contribution, we introduce a new attack strategy: the Joint Moment Regression (JMR) in section 5.3. It is built to circumvent the issues found in the state-of-the-art and proposes a method which is agnostic to the underlying masking scheme.
- We then draw a parallel between the core of JMR and a more general framework: the Generalized Method of Moment (GMM) [Han82] which is a well-studied paradigm in statistics and economics. This allows to improve our attack in the case of biased masking schemes such as the multiplicative and affine ones. This analysis can be found in section 5.4.
- Finally, section 5.5.2 presents applications of JMR to real traces and provides at the same time, to the best of our knowledge, the first unsupervised attack on the secured AES implementation of the ANSSI, protected by an affine masking scheme. Attacks that do and do not exploit the lower-order leakage are both presented.

## 5.2 Related Work and Limitations

### 5.2.1 General Attack Framework

We recall the notation and attack framework used in this chapter. We consider that an adversary targets the manipulation of a sensitive variable $Z \in \mathcal{Z} = \mathbb{F}_2^n$, for a given $n \in \mathbb{N}$. This variable is supposed to functionally depends on a public variable $X \in \mathcal{X} = \mathbb{F}_2^m$, for a given $m \in \mathbb{N}$, and a secret key $k^* \in \mathcal{K} = \mathbb{F}_2^m$ through the relation: $Z = f(X, k^*)$ where $f : \mathcal{X} \times \mathcal{K} \to \mathcal{Z}$ is a known function depending on the underlying cryptographic algorithm. The adversary is supposed to own a set $\{(\ell_i, x_i), 1 \leq i \leq N\}$ of $N$ side channel traces labeled with the corresponding public value of $X$. Traces correspond to realizations of a leakage variable $L \in \mathcal{L}$ coming from a stochastic process $\mathcal{S}$, $Z \xrightarrow{\mathcal{S}} L$ (often separable into a deterministic and a noise part). The leakage variable $L$ is supposed to contain information about $Z$. The general idea of an unsupervised side-channel attack is to make a series of hypotheses $k_i$ on the key, and to use the dependency between $Z$ and $L$ to build a distinguisher $\mathcal{D} : \mathcal{K} \to \mathbb{R}$ to rank the different key candidates. One of these distinguishers, the LRA, is described in the next section.

### 5.2.2 Linear Regression Analysis

The following procedure recalls the steps required to perform an LRA such as suggested in [LPR13]. Traces are assumed to feature one sample. In a real-life scenario, the same procedure would be repeated for each sample and the final distinguisher keeps the best value along all samples according to a chosen policy (often being the minimum/maximum value of the distinguisher).

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{\ell_i, x_i = x\}$.
2. **Averaging.** Compute the average trace for each class $\bar{L} = (\bar{\ell}_x)_{x \in \mathcal{X}}$ with
$$\bar{\ell}_x = \frac{1}{|\mathcal{L}_x|} \sum_{\ell \in \mathcal{L}_x} \ell$$
3. **Basis choice.** Choose a basis of functions $(b_i)_{1 \leq i \leq r}$ such that $b_i : \mathcal{Z} \to \mathbb{R}$.
4. **Making hypotheses.** For $k \in \mathcal{K}$ compute the hypotheses matrix:
$$\mathcal{H}_k = \Big(b_i \circ f(x, k)\Big)_{\substack{x \in \mathcal{X}, \\ 1 \leq i \leq r}}$$
5. **Linear regression.** For $k \in \mathcal{K}$ find the parameter vector $\theta_k = (\theta_{k,1}, \ldots, \theta_{k,r})^T$ minimizing the euclidean norm of the error vector:
$$\theta_k = \underset{\theta}{\operatorname{argmin}} \, ||\mathcal{H}_k \cdot \theta - \bar{L}||_2$$

6. **Ranking.** Rank the keys according to their distinguisher value (from low to high)[1]:

$$\mathcal{D}(k) = ||\mathcal{H}_k \cdot \theta_k - \bar{L}||_2$$

Since step 5 corresponds to a linear regression it has a closed-form solution:

$$\theta_k = ({\mathcal{H}_k}^T \cdot \mathcal{H}_k)^{-1} \cdot {\mathcal{H}_k}^T \cdot \bar{L}$$

However, to highlight similarities with JMR later in the chapter, we decided to keep the generic formulation of the optimization problem.

The choice of the basis is important since it should be large enough for the leakage to be representable as a linear combination with the $b_i \circ f$ functions when $k = k^*$ but small enough so that it is not the case for wrong hypotheses. The adversary uses his *a priori* on the leakage model, often related to physical assumptions, to choose the basis.

A common example is to assume that each bit of the sensitive variable contributes to the leakage independently from the others. If this assumption holds there exists $\alpha = (\alpha_0, \ldots, \alpha_n)$ such that $\ell_i = \alpha_0 + \sum \alpha_j \cdot bit_j(z_i) + \epsilon$ with $bit_j$ denoting the projection on the $j^{th}$ bit and $\epsilon$ being sampled from a noise distribution. In such a case, the basis would be $\{1, bit_1, \ldots, bit_n\}$ and $\theta_{k^*}$ should be close to $\alpha$.

Another example is to assume that the leakage is depending on the Hamming Weight (HW) of the sensitive variable so that $\ell_i = \alpha_1 \mathrm{HW}(z_i) + \alpha_0 + \epsilon$. The basis is then reduced to $\{1, \mathrm{HW}\}$ and the attack corresponds to the classical CPA.

### 5.2.3 Masking

To prevent instantaneous leakages and mitigate the first-order attacks presented above, one of the most widely used countermeasures is masking [Cha+99]. The idea is to split each sensitive intermediate value $Z$, into $d$ shares: $(Z_i)_{1 \leq i \leq d}$. The $d-1$ shares $Z_2, ..., Z_d$ are randomly chosen and the last one, $Z_1$ is processed such that:

$$Z_1 = Z * Z_2 * \cdots * Z_d \tag{5.1}$$

for a group operation $*$ of $\mathcal{Z}$. This has the effect of complexifying the stochastic process $\mathcal{S}$ generating $L$ from $Z$, rendering it no longer separable into a deterministic and a noise part. Assuming the masks are uniformly distributed, the knowledge of $d-1$ shares does not tell anything about $Z$ (this is why such masking is said to be of order $d-1$). Therefore, any sound SCA strategy has to combine leakage samples from the $d$ shares to perform an attack (which corresponds to at least $d$ samples if the leakages are disjoint). Such attacks are called $d^{th}$ order attacks. One of them, the second-order LRA is presented in the next section.

[1] Sometimes the coefficient of determination $R^2$ is used instead but the ranking is strictly equivalent except that one ranks from high to low values of the distinguisher.

| | Group operation | Masked Variable ($Z_1$) | Uniform | Reference |
|---|---|---|---|---|
| Boolean | $\oplus$ | $Z \oplus Z_2 \oplus \cdots \oplus Z_d$ | Yes | [Cha+99] |
| Arithmetic | $+ \ mod \ 2^n$ | $Z + Z_2 + \cdots + Z_d \ [2^n]$ | Yes | [CG00] |
| Multiplicative | $\otimes$ | $Z \otimes Z_2 \otimes \cdots \otimes Z_d$ | No | [GPQ11] |
| Affine | $\oplus, \otimes$ | $Z \otimes Z_2 \oplus Z_3$ | No | [Fum+11] |

TABLE 5.1: Masking schemes studied in this work

The uniform assumption is sometimes not strictly realized in practice depending on the masking scheme being used. Four of the most common masking schemes that will be studied in this chapter are listed in Table 5.1. The $\oplus$ and $\otimes$ respectively stand for the addition and the multiplication operation in $\mathbb{F}_2^n$. Since the multiplication by 0 is not invertible the "multiplicative shares" have to be chosen in $\mathbb{F}_2^n \setminus \{0\}$. As $Z$ itself can take the value 0, the multiplicative and affine schemes are then slightly biased, and therefore, do not guarantee in theory, SCA resilience to all the $(d-1)^{th}$ and lower order attacks. Such attacks will be discussed in section 5.4.

### 5.2.4 Second-Order LRA

This section describes the generalization of the LRA introduced in [DDP13] which aims at defeating a first-order masked implementation ($d = 2$). Traces are considered to be composed of 2 samples: $L = (L_1, L_2)$ where $L_1$ and $L_2$ represent respectively the leakage of the first and second share. In a real-life scenario, the attack would be repeated with all the combinations of two samples from the raw traces. To perform a second-order LRA the adversary is supposed to own a set of $N$ traces $\{(\ell_1^i, \ell_2^i), 1 \leq i \leq N\}$. The idea is to replace the estimated mean per class by the estimated covariance per class in the classical LRA which naturally combines information from the two samples. Indeed the covariance $Y = \mathrm{cov}(L_1, L_2)$ involves the product of the centered variable $L_1 - \mu_1$ and $L_2 - \mu_2$, with $(\mu_1, \mu_2) = \mathbb{E}[L]$, which has been shown to be a good combining function for second-order SCA [PRB09]. The steps to perform a second-order LRA are depicted hereafter.

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{(\ell_1^i, \ell_2^i), x_i = x\}$.

2. **Estimating covariances.** Compute the estimated covariance for each class $\bar{Y} = (\bar{y}_x)_{x \in \mathcal{X}}$ with
$$\bar{y}_x = \frac{1}{|\mathcal{L}_x|} \sum_{\ell \in \mathcal{L}_x} (\ell_1 - \bar{\mu}_1)(\ell_2 - \bar{\mu}_2)$$
where $(\bar{\mu}_1, \bar{\mu}_2)$ stands for the estimated mean of $L$.

3. **Basis choice.** Choose a basis of functions $(b_i)_{1 \leq i \leq r}$ such that $b_i : \mathcal{Z} \to \mathbb{R}$.

4. **Making hypotheses.** For $k \in \mathcal{K}$ compute the hypotheses matrix:
$$\mathcal{H}_k = \Big( b_i \circ f(x, k) \Big)_{\substack{x \in \mathcal{X}, \\ 1 \leq i \leq r}}$$

5. **Linear regression.** For $k \in \mathcal{K}$ find the parameter vector $\theta_k = (\theta_{k,1}, \ldots, \theta_{k,r})^T$ minimizing the euclidean norm of the error vector:

$$\theta_k = \underset{\theta}{\operatorname{argmin}} ||\mathcal{H}_k \cdot \theta - \bar{Y}||_2$$

6. **Ranking.** Rank the keys according to their distinguisher value (from low to high):

$$\mathcal{D}(k) = ||\mathcal{H}_k \cdot \theta_k - \bar{Y}||_2$$

The attack may seem very similar to a first-order LRA except that it is performed on the covariance instead of the mean (the change happens in step 2). However, the choice of the basis is much more delicate. The link between the adversary *a priori* and a basis leading to a successful attack is not trivial anymore. Indeed, the hypotheses matrix is constructed using the unmasked variable $Z^{(k)} = f(X, k)$ while the leakage *a priori* concerns the shares. The choice of the basis proposed in [DDP13] is based on an assumption that is recalled hereafter.

Let us define the set of functions $(\varphi_k)_{k \in \mathcal{K}} : \mathcal{Z} = \mathbb{F}_2^n \to \mathbb{R}$ such that:

$$\varphi_k(z) = \operatorname{cov}(L_1, L_2 \mid Z^{(k)} = z) \tag{5.2}$$

Since all the Boolean functions in $\mathbb{F}_2^n$ can be represented by a multivariate polynomial in $\mathbb{R}[z_1, \ldots, z_n]/(z_1^2 - z_1, \ldots, z_n^2 - z_n)$ (*i.e.* the degree of every $z_i$ in every monomial is at most 1) [Car07], there exists, for any $k$, a unique set of coefficients $(\alpha_{k,u})_{u \in \mathbb{F}_2^n}$ such that:

$$\varphi_k(z) = \sum_{u=(u_1,\ldots,u_n) \in \mathbb{F}_2^n} \alpha_{k,u} \cdot z^u \tag{5.3}$$

where each term $z^u$ denotes the monomial (function) $z \to z_1^{u_1} z_2^{u_2} \ldots z_n^{u_n}$ with $z_i^{u_i} \in \mathbb{F}_2$. Let $deg(\varphi_k)$ stands for the degree of the polynomial representing $\varphi_k$. The assumption on which the attack from [DDP13] relies is the following:

**Assumption 1.** $\forall k \neq k^*, deg(\varphi_{k^*}) < deg(\varphi_k)$.

The intuition behind this assumption is that since $\varphi_k = \varphi_{k^*} \circ f_k \circ f_{k^*}^{-1}$ (where $f_k = f(\cdot, k)$), $\varphi_k$ is expected to have a high degree (close to $n$) if $k \neq k^*$, due to cryptographic properties of $f$ which often embeds highly non-linear S-boxes to prevent algebraic attacks. Note that this reasoning only holds if $\varphi_{k^*}$ itself has a low degree which is implicitly assumed in [DDP13]. This point will be discussed later.

If Assumption 1 holds, the basis: $(b_i)_i = \{z^u, u \in \mathbb{F}_2^n, \mathrm{HW}(u) \leq deg(\varphi_{k^*})\}$ is a valid basis for the second-order LRA. Indeed, it spans all the functions of degree less or equal $deg(\varphi_{k^*})$. Therefore there exists a decomposition of $\varphi_{k^*}$ in this basis while it is not the case for other $\varphi_k$, by hypothesis, which guarantees the success of the attack (provided that the number of traces allows for a fair approximation of the covariances per class).

### 5.2.5 Limitations

The first observation is that even if Assumption 1 holds, the attack may fail in practice if $\deg(\varphi_{k^*})$ is not low enough. Indeed, the cardinal of the basis, and therefore the number of parameters to estimate, increases quickly with $deg(\varphi_{k^*})$ offering a big capacity to the statistical model to fit the data whatever the considered value of $k$. If the noise is not negligible, this often means that the wrong hypotheses can reach similar scores to the correct one which reduces the distinguishability and therefore the effectiveness of the attack. For example, with $n = 8$ and $deg(\varphi_{k^*}) \in \{1, 2, 3\}$ the cardinal of the basis is respectively equal to 9, 37 and 93. In practice authors of [DDP13] run their attack with the following basis: $(b_i)_i = \{z^u, u \in \mathbb{F}_2^n, \mathrm{HW}(u) \leq d_{max}\}$ where $d_{max} \in \{1, 2, 3\}$. Choosing $d_{max} = 3$ never led to the best attack even in cases where $deg(\varphi_{k^*})$ was strictly greater than 2 (due to the high model capacity and lack of distinguishability).

Then, one may ask if Assumption 1 holds at all. Since $\varphi_{k^*}(z) = \mathrm{cov}(L_1, L_2 | Z^{(k^*)} = z)$ it is obviously related to the nature of the leakage $L_1$ and $L_2$. These variables can be assumed to be separable into a deterministic and a noise part with respect to the shares:

$$L_i = l_i(Z_i) + \epsilon_i \tag{5.4}$$

with $l_i : \mathcal{Z} \to \mathbb{R}$ representing the leakage of share $i$ and $\epsilon_i$ being an independent random noise variable. By bilinearity of the covariance and independence of $\epsilon_i$:

$$\begin{aligned} \varphi_{k^*}(z) &= \mathrm{cov}\big(l_1(Z_1), l_2(Z_2) \mid Z^{(k^*)} = z\big) \\ &= \mathrm{cov}\big(l_1(z * Z_2), l_2(Z_2)\big) \end{aligned} \tag{5.5}$$

since $Z_1 = Z^{(k^*)} * Z_2$. Both $l_1$ and $l_2$ can be assumed of low degree (through a physical *a priori* on the leakage). For example, it is realistic to assume that both shares follow a linear leakage. But we argue that this is not enough to guarantee Assumption 1 and that it is still depending on the underlying masking scheme, especially on the nature of the $*$ operation.

Then, a natural question arises: why does the attack presented in [DDP13] work? We argue that it is related to the studied masking schemes in their paper. Indeed, the latter one focuses on the Boolean and arithmetic masking schemes which are both exceptions as far as Assumption 1 is concerned. This claim is justified by the two following propositions.

**Proposition 5.** *(Boolean masking) Let $* = \oplus$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degree 1. Let $\varphi_{Bool}(z) = \mathrm{cov}\big(l_1(z \oplus Z_2), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Bool}) \leq 1 \tag{5.6}$$

Proof can be found in Appendix D.

**Proposition 6.** *(Arithmetic masking) Let* $* = + \, mod \, 2^n$. *Let* $l_1 : \mathcal{Z} \to \mathbb{R}$ *and* $l_2 : \mathcal{Z} \to \mathbb{R}$ *be two leakage functions of degree 1. Let* $\varphi_{Arith}(z) = \mathrm{cov}\big(l_1(z + Z_2 \, [2^n]), l_2(Z_2)\big)$. *Then,*

$$deg(\varphi_{Arith}) \leq 2 \tag{5.7}$$

Proof can be found in Appendix D.

These two propositions explain the success of the attacks presented in [DDP13]. However, we could not find equivalent propositions for other masking schemes, suggesting that Boolean and arithmetic masking are, in fact, exceptions. This will be empirically confirmed in subsection 5.3.4 where it is shown that even without noise, the higher-order LRA fails against multiplicative or affine masking with a linear leakage of the shares. Therefore, to the best of our knowledge, there is no strategy in the literature able to defeat a generic masking scheme in an unsupervised context, with a simple linear leakage assumption of the shares. We introduce such a strategy in the next section.

## 5.3 Joint Moments Regression

We first introduce the concept of Joint Moment (JM) which generalizes to any masking order the idea of the covariance, found in the previous section.

### 5.3.1 Joint Moments

Moments of probability distributions are quantitative measures related to the shape of the distribution. The moment of order $d$, denoted $\mu_d$, of the variable $X$ is defined as:

$$\mu_X^{(d)} = \mathbb{E}[X^d] \tag{5.8}$$

For second and higher orders, the centered moments $\check{\mu}_d$ of order $d$ are often used instead and are defined as:

$$\check{\mu}_X^{(d)} = \mathbb{E}[(X - \mu_X^{(1)})^d] \tag{5.9}$$

Joint moments are the generalization of moments to multivariate variables. Let $X = (X_1, \ldots, X_n) \in \mathbb{R}^n$ be a multivariate random variable. Let $u = (u_1, \ldots, u_k) \in \mathbb{N}^n$ be a vector of positive integers such that $\sum u_i = d$. The JM of order $d$ with respect to vector $u$, denoted $jm_u$, is defined as:

$$jm_X^{(u)} = \mathbb{E}\left[\prod_{i=1}^{n} X_i^{u_i}\right] \tag{5.10}$$

Centered JM are also defined as:

$$\check{jm}_X^{(u)} = \mathbb{E}\left[\prod_{i=1}^{n} (X_i - \mu_{X_i}^{(i)})^{u_i}\right] \tag{5.11}$$

One important property of JM (and of simple moments) is that, for distributions defined on a compact set of $\mathbb{R}^n$, the distribution is fully defined by the list (maybe infinite) of all its JM. This is also true for the centered JM provided that the first order JM are also given.

The effect of a $d$-order masking is that no information related to the sensitive variable can be found in the $d-1$ and lower JM. That is why the second-order LRA performed a regression on the second-order centered JM with $u = (1,1)$ which happens to be the covariance. Indeed it is the lowest order JM bringing information on the sensitive variable. Information could also be found in higher-order JM but they are harder to estimate. Indeed, more terms are involved in the product and the noise in each one of them is amplified through the multiplication. One typically wants to take the JM with the lowest standard error (the standard deviation of its estimator). This also explains why centered JM are preferred: as shown in [PRB09], they have a lower standard error than their uncentered counterpart.

### 5.3.2 Attack Description

Let an adversary own a set of $N$ traces $\{\ell^i, 1 \leq i \leq N\}$ of a $d$ order masked implementation. Traces are considered to be composed of $d$ samples: $\ell^i = \{(\ell^i_1, \ldots, \ell^i_d)$. In a real-life scenario, the attack would be repeated on combinations of $d$ samples from the raw traces depending on the attacker a priori on the points of interest. To defeat this implementation a naive solution would be to extend the attack proposed in [DDP13] using centered JM instead of covariance but as stated in section 5.2.5: there is no obvious link between the physical *a priori*, which happens to be on the shares, and the basis that has to be chosen and applied to the unmasked sensitive variable.

That is why we propose a new strategy where the adversary chooses $d$ basis, one for each share (in practice they will often be the same basis), and directly regresses the leakage of each share using information from the estimated $d$ order centered JM. The steps of what we call the Joint Moment Regression (JMR) are depicted hereafter.

**JMR Procedure**

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{(\ell^i_1, \ldots, \ell^i_d), x_i = x\}$.
2. **Estimating JM.** Compute the estimated centered $d$ order joint moments matrix $J\bar{M}$. Each row represents the estimation for one class:

$$\overline{JM} = \begin{pmatrix} \frac{1}{|\mathcal{L}_0|} \sum_{\ell \in \mathcal{L}_0} \prod_{j=1}^{d} (\ell_j - \bar{\mu}_j) \\ \vdots \\ \frac{1}{|\mathcal{L}_{2^m-1}|} \sum_{\ell \in \mathcal{L}_{2^m-1}} \prod_{j=1}^{d} (\ell_j - \bar{\mu}_j) \end{pmatrix}$$

where $(\bar{\mu}_1, \ldots, \bar{\mu}_d)$ stands for the estimated mean of $L$.

3. **Basis choice.** For $j \in [1, d]$, choose a basis of functions $(b_i^{(j)})_{1 \leq i \leq r}$ such that $b_i^{(j)} : \mathcal{Z} \to \mathbb{R}$. Intuitively, if $l_j$ corresponds to the leakage of share $j$, the adversary wants to choose a basis such that $l_j(z_j) = \sum_{i=1}^{r} \theta_{j,i} \cdot b_i^{(j)}(z_j) + \epsilon_j$ for some coefficient $\theta_j \in \mathbb{R}^r$, with $\epsilon_j$ representing an independent random noise variable.

4. **Making hypotheses.** Let $\tilde{l}_j(z_j)$ stand for the leakage prediction of share $j$ according to the chosen basis:

$$\tilde{l}_j(z_j) = \sum_{i=1}^{r} \theta_{j,i} \cdot b_i^{(j)}(z_j) \tag{5.12}$$

For $k \in \mathcal{K}$, define the theoretical JM vector $JM_k(\theta)$ with respect to $\theta \in \mathbb{R}^{d \times r}$, that traduces the leakage assumption of step 3 into $|\mathcal{X}| = 2^m$ JM per class expressions:

$$JM_k(\theta) = \begin{pmatrix} a_0 \sum\limits_{(z_1,\ldots,z_d) \in \mathcal{A}_0} \prod\limits_{j=1}^{d} (\tilde{l}_j(z_j) - \mu_{\theta_j}) \\ \vdots \\ a_{2^m-1} \sum\limits_{(z_1,\ldots,z_d) \in \mathcal{A}_{2^m-1}} \prod\limits_{j=1}^{d} (\tilde{l}_j(z_j) - \mu_{\theta_j}) \end{pmatrix}$$

with $\mathcal{A}_x = \{(z_1, \ldots, z_d) | Z = f(x, k)\}$ and $a_x = \frac{1}{|\mathcal{A}_x|}$. Here, $\mu_{\theta_j}$ stands for the theoretical mean of the leakage of share $j$ under the assumption of $\theta_j$:

$$\mu_{\theta_j} = \mathbb{E}_{Z_j}\left[\sum_{i=1}^{r} \theta_{j,i} \cdot b_i^{(1)}(Z_j)\right]$$

5. **Non-linear regression.** For $k \in \mathcal{K}$, find through numerical optimization techniques (see subsection 5.3.3), the parameter vector $\theta^{(k)} \in \mathbb{R}^d \times \mathbb{R}^r$ minimizing the euclidean norm of the error vector:

$$\theta^{(k)} = \underset{\theta}{\operatorname{argmin}} \, ||JM_k(\theta) - \overline{JM}||_2$$

6. **Ranking.** Rank the keys according to their distinguisher value (from low to high):

$$\mathcal{D}(k) = ||JM_k(\theta^{(k)}) - \overline{JM}||_2$$

### 5.3.3 Attack Soundness

The general attack structure of JMR is very similar to the LRA and second-order LRA. The main difference with the latter one is that the assumption is done on the

leakage of the shares and is therefore directly related to the physical *a priori*. These assumptions are then combined to build a parameterized system of unknown $\theta \in \mathbb{R}^{d \times r}$:

$$JM_k(\theta) - \overline{JM} = 0 \tag{5.13}$$

where each line represents a condition on the JM knowing that $X = x$. Note that by the independence assumption, the noise terms $\epsilon_j$ are canceled from the theoretical equations of the JM per class, listed in the $JM_k(\theta)$ vectors. The goal is then to find the solution $\theta^{(k)}$ that fits the most the system and to use a measure of fitness as distinguisher. Note that the knowledge of the underlying masking scheme is embedded in the system through the $\mathcal{A}_x$ sets which describe the possible values $(z_1, \ldots, z_d)$ of the shares given the value of $Z$. This is what ensures the genericity of JMR regarding the masking scheme.

When the number of traces $N$ tends towards infinity, the estimated JM per class $\overline{JM}$ tends towards the true JM per class. If the leakage assumptions are correct there exists $\theta^{(k^*)} \in \mathbb{R}^{d \times r}$ such that $JM_{k^*}(\theta^{(k^*)})$ is equal to the true JM per class. Therefore:

$$\lim_{N \to \infty} \mathcal{D}(k^*) = 0 \tag{5.14}$$

while it is unlikely to be the case for $k \neq k^*$ due to cryptographic property of $f$, which assures the soundness of the attack.

However, this multi-shares assumption comes at the cost of linearity. Indeed, even if all the shares are assumed to leak linearly, the system that JMR regresses is not linear anymore: it is of degree $d$. Therefore there is no closed-form solution and one has to use numerical optimization tools to find an approximation of the solution. Numerical optimization is a research field in itself and is out of the scope of this paper. There exist multiple ready-to-use implementations in different programming languages, which is enough for our concern. Note that since the system is of degree $d$, the uniqueness of the solution of step 5 is not guaranteed. This is not a problem for the attack: as long as one solution can be found and that equation 5.14 holds only for the correct key hypothesis, the attack will succeed for a sufficient number of traces.

### 5.3.4 Simulation Experiments

This section provides simulation experiments to assess the feasibility of JMR in practice against the masking schemes presented in table 5.1. Its efficiency is compared with state-of-the-art attacks at second and third order.

#### Implementation

We implemented the core of the JMR attack using the `least_squares` function from the python `scipy.optimize` package [Vir+20]. It solves a non-linear least-squares fitting problem using the Levenberg-Marquardt (LM) algorithm [Lev44; Mar63] which

is itself based on the Gauss-Newton algorithm and the method of gradient descent. The attack time or complexity is mostly constant regarding the number of traces because the latter does not affect the number of parameters nor equations in the system. The only part that scales with the number of traces is the estimation of the JM per class which is just a product and a sum and that can be handled with `numpy` [Har+20] array manipulations.

Since the least-squares problems related to the different key hypotheses are independent, the implementation is highly parallelizable. We exploited this using a 48 cores Xeon Platinium 8168 processor which speeded up the attack by a significant factor since the implementation of the `least_squares` function is not parallelized in itself. Other implementation optimization could be explored such as using the fast GPU version of the LM algorithm proposed in [Prz+17] but this is not in the scope of this paper. To give an order of magnitude, with our setup, running the full JMR procedure as described in subsection 5.3.2 for a $d-$tuple of time samples, requires around 10 and 15 seconds for respectively a second and third-order attack (assuming one trace for each possible values of the shares: $2^{16}$ and $2^{24}$ respectively).

#### Generating Datasets

To assess the JMR method and to compare it with state-of-the-art attacks, synthetic trace datasets with linear leakage of the shares have been generated for first and second-order masking ($d \in \{2,3\}$). Boolean, arithmetic, multiplicative and affine (only with $d = 3$) schemes are used to mask the classical sensitive variable of an AES: $Z = \texttt{Sbox}[k^* \oplus P]$ ($k^*$ and P are both supposed to be 8 bits long). To be able to average the results of 100 different attacks, performed with 100 different linear leakage models, we have generated a matrix of random coefficients $C_i$ for each share ($1 \leq i \leq d$):

$$C_i = \left(\alpha_{a,b}\right)_{\substack{0\leq a\leq 99 \\ 0\leq b\leq 8}} \tag{5.15}$$

where all the $\alpha_{a,b}$ are uniformly drawn from $[-1,1]$. Each row represents a different linear leakage model.

To avoid any kind of estimation error (the error coming from sampling), each dataset contains one trace for each of the possible values of the shares $(z_1,\ldots,z_d) \in \mathcal{Z}^d$ (for multiplicative and affine schemes the multiplicative shares can not be 0 so we take them from $[\![1,255]\!]$ instead). The trace $\ell^{(a)}_{(z_1,\ldots,z_d)}$ corresponding to the $d$-tuple $(z_1,\ldots,z_d)$ is generated by concatenating the leakage of each shares (represented by the $a^{th}$ row of the $C_i$ matrices) as follows:

$$\ell^{(a)}_{(z_1,\ldots,z_d)} = \left[l_1^{(a)}(z_1),\cdots,l_d^{(a)}(z_d)\right] \tag{5.16}$$

**Algorithm 4:** Generate Traces

**Input:** $k^*$, The correct key byte
**Input:** $a$, representing a row in the matrices $C_i$
**Input:** $d$, the masking order
**Input:** $\star$, a group operation with / the associated division
**Input:** $\sigma$, the value of the noise
**Output:** $L$, *a* $(2^{8*d}, d)$ *array*
**Output:** $P$, *a* $(2^{8*d})$ *array*
$L \leftarrow$ empty list
$P \leftarrow$ empty list
**for** $(z, z_2, \ldots, z_d) \in \mathcal{Z}^d$ **do**
  $z_1 \leftarrow z \star \cdots \star z_d$
  $l \leftarrow \ell^{(a)}_{(z_1,\ldots,z_d)}$ (*Equation* 5.16)
  $p \leftarrow \texttt{Sbox}^{-1}[z] \oplus k^*$
  Append $l$ to $L$
  Append $p$ to $P$
**end**
$R \leftarrow$ Draw a $(2^{8*d}, d)$ array from $\mathcal{N}(0, \sigma^2)$
$L \leftarrow L + R$
**return** L, P

with

$$l_i^{(a)}(z_i) = C_i[i, 0] + \sum_{b=1}^{8} C_i[a, b] \cdot z_i[b] + \epsilon_i(\sigma) \tag{5.17}$$

where $z_i[b]$ corresponds to the $b^{th}$ bit of $z_i$ and $\epsilon_i$ is drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$. The exact procedure that generates the traces considering the the $a^{th}$ leakage model is depicted in algorithm 4.

### Results

Second-order attacks results are presented in Figure 3.1. Each point represents the average rank of $k^*$ over the 100 datasets for a given value of $\sigma$. We recall that we are using exhaustive datasets, therefore, a failed attack for a given value of $\sigma$ does not mean that the attack is impossible but rather that the adversary would need more traces than one per possible value of the shares. We compare JMR with

- A higher-order CPA, denoted HO-CPA, computed with a Hamming weight prediction model and using the JM of order $d$ as combining function which happens to be the same as the centered product described in [PRB09].
- Higher order LRA, denoted HO-LRA-$d_{max}$ where $d_{max}$ is the assumed degree of $\varphi_{k^*}$ as defined in Equation 5.5. Therefore the basis used in HO-LRA-$d_{max}$ is $(b_i)_i = \{z^u, u \in \mathbb{F}_2^n, \mathrm{HW}(u) \leq d_{max}\}$. The combining function is also the JM of order $d$ which for $d = 3$ is a straightforward extension of the second order attack described in [DDP13].

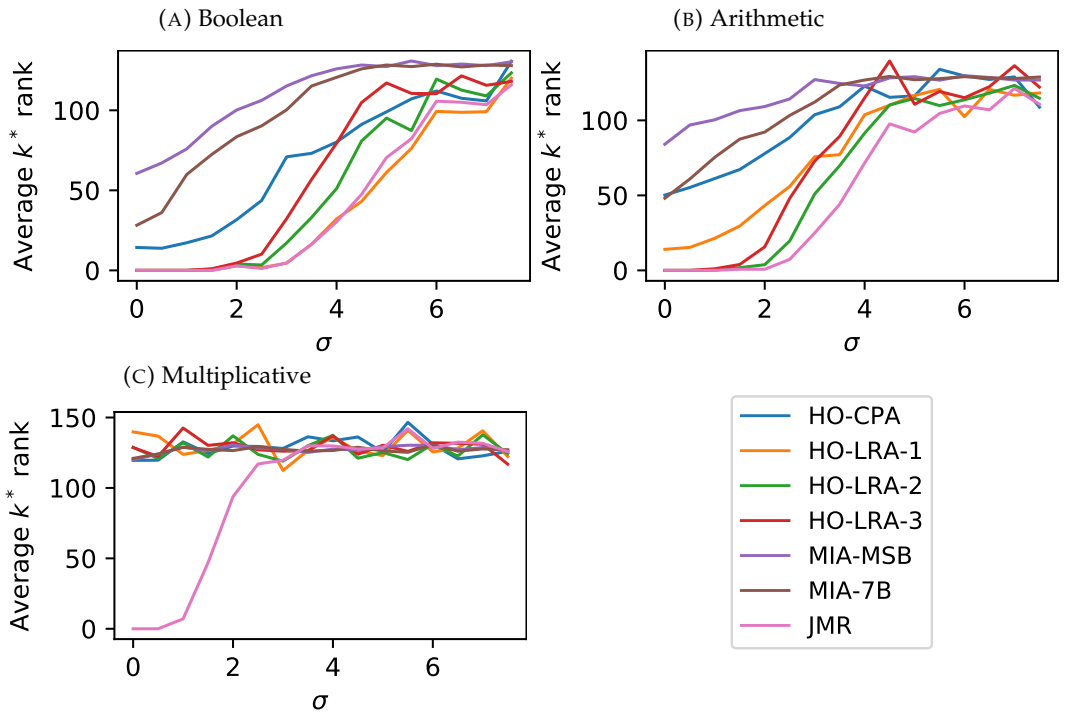



FIGURE 5.1: Guessing entropies versus standard deviation of the noise for the considered second-order attacks after the processing of A) $2^{16}$, B) $2^{16}$, C) $2^8 \times 255$ traces.

- Mutual Information Analysis, denoted MIA-$f$, where the distinguisher used is $MI(f(Z_k), L)$. MIA requires the use of a non-injective function $f$ to create distinguishability for the correct hypothesis. Since the leakage model is unknown we used very generic models: $f = MSB$ and $f = 7B$, where $MSB$ stands for the most significant bit of $Z_k$ and $7B$ stands for the 7 most significant bits of $Z_k$. The MI has been estimated using the histogram method described in [PR09].

(a) For the Boolean case, JMR and HO-LRA-1 performs approximately the same which is not surprising since, by Proposition 5, Assumption 1 holds for HO-LRA1. It also holds for HO-LRA-2/3 but HO-LRA-1 perfectly explains the data with fewer parameters, and thus, performs better. One can notice that even without noise the HO-CPA is not converging towards 0 which confirms that it relies on the Hamming weight leakage assumption. Also, MIA strategies do not perform well which is not surprising since the underlying leakage model is unknown and it is, therefore, hard to select a good non-injective function.

(b) For the arithmetic scheme, JMR outperforms all the other attacks even, HO-LRA-2 in which Assumption 1 holds by Proposition 6. Again this is explained by the fact that JMR only needs $(9 \times 2)$ parameters to predict the data while HO-LRA-2 needs 37 parameters. Even without noise, the data can not be perfectly explained in an HO-CPA or HO-LRA-1 model since their curves do not converge towards 0.

(c) For the multiplicative scheme, as predicted, none of the state-of-the-art attacks perform better than random even without noise which confirms that Assumption 1 does not hold at all for such masking scheme. JMR is the only sound attack in this case.

Results for third-order attacks are presented in Figure 3.2. In this case, HO-LRA-$d_{max}$ represents the generalization of the second-order LRA replacing the covariance by the third-order joint moment. Conclusions are the same than for the second-order attacks. Among the considered attack strategies, one can observe that, as for the multiplicative case, JMR is the only sound option to attack affine masking under a linear leakage of the shares.

**About the Biased Schemes**

Both multiplicative and affine schemes are slightly biased which can induce lower-order leakage. We argue that such leakage has not been exploited in this section since the estimated JM were computed with the leakage of all the shares (thus, the variance of the estimation result from $d$ multiplications of noisy leakages). To confirm this statement, we repeated the previous experiments for the biased schemes removing $Z = 0$ from the possible values, thus, simulating non-biased schemes. The results being essentially the same than those presented in Figures 3.1c, 5.2c and 5.2d so we do not plot them. Since the multiplicative and affine masking do not seem to have special algebraic properties like the Boolean and arithmetic scheme as shown in propositions 5 and 6, we argue that these results could be extended to any other masking scheme. Indeed, the real added value of JMR is its ability to encode the scheme knowledge in the system's equation making it generic and able to work even for non-biased schemes with a high algebraic degree[2] where other attacks would not.

However, in the specific case of biased schemes, lower-order leakage could be exploited with simpler attacks such as a classical CPA with a zero-valued based power model. One could also perform more advanced attacks taking advantage of leakages at multiple orders at the same time. All these attacks are discussed in the next section where we introduce the generalized method of moment paradigm.

[2] Formally, we refer to the degree of the function $f$ representing the joint moments per class $f(z) = JM(l_1(Z_1), \ldots, l_d(Z_d) \mid Z = z$ according to the degree of the leakage function $l_i$.

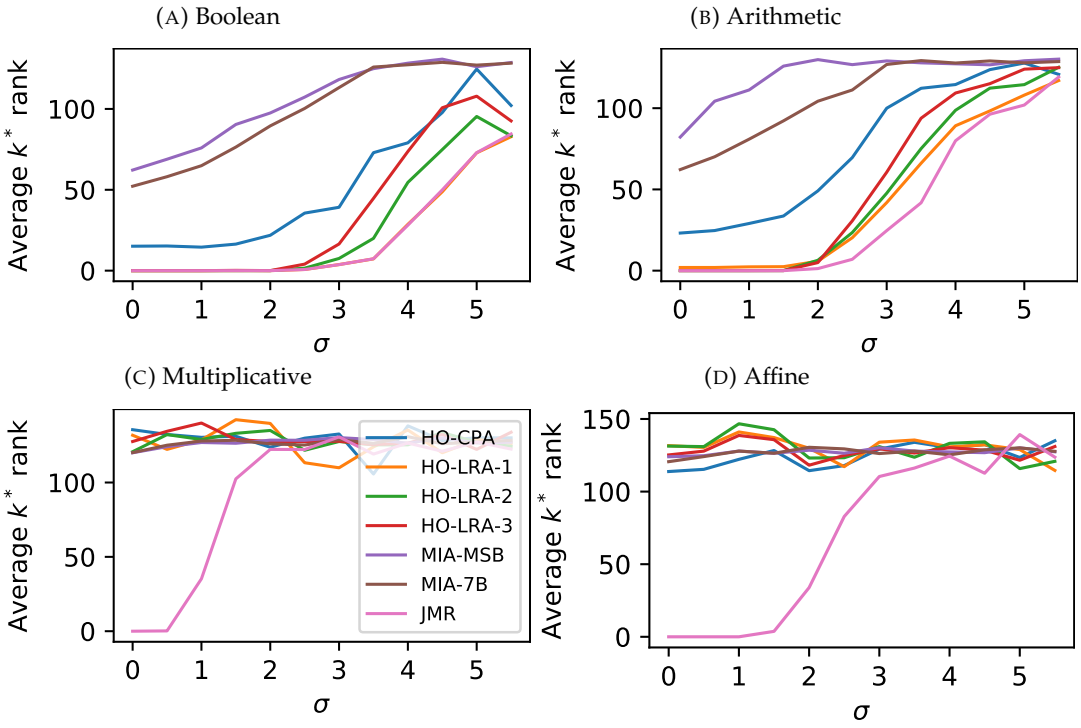



FIGURE 5.2: Guessing entropies versus standard deviation of the noise for the considered third-order attacks after the processing of A) $2^{24}$, B) $2^{24}$, C) $2^8 \times 255^2$, D) $2^{16} \times 255$ traces.

## 5.4 Generalized Method of Moments Paradigm

Looking from a broader perspective, it appears that the core of the JMR attack can be seen as part of a more general framework known as the Generalized Method of Moments (GMM) [Han82]. This method comes from the field of statistics and economy and its main purpose is to estimate parameters in a statistical model. Embracing this paradigm requires to gain a level of abstraction but it allows to use the powerful mathematical foundations behind it. In particular, it will tell us how to optimally combine information from different orders, which is useful when the masking scheme is biased.

### 5.4.1 Background on GMM

Let suppose that the available data consists of $N$ observations $(L_i)_{1 \leq i \leq N}$ of a random variable $L \in \mathbb{R}^n$. This data is assumed to come from a stochastic process defined up to an unknown parameter vector $\theta \in \mathbb{R}^p$. The goal is to find the true value $\theta_0$ of this parameter or at least a reasonably close estimate.

In order to apply GMM the data must come from a weakly stationary ergodic stochastic process (independent and identically distributed (iid) variables are a special case of these conditions). Then one needs to have $c$ "moment conditions" defined as a function $g(\ell, \theta) : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^c$ such that:

$$\mathbb{E}[g(L, \theta_0)] = 0 \tag{5.18}$$

The idea is then to replace the theoretical expectation with its empirical analog:

$$m(\theta) = \frac{1}{N}\sum_{i=1}^{N} g(\ell_i, \theta) \tag{5.19}$$

and to minimize the norm of $m(\theta)$ with respect to $\theta$. The properties of the GMM estimator depend on the chosen norm and therefore the theory considers the entire family of norms defined up to a positive-definite weighting matrix $W \in M_c(\mathbb{R})$:

$$||m(\theta)||_W = \sqrt{m(\theta)^T W m(\theta)} \tag{5.20}$$

The GMM estimator is then defined as:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} ||m(\theta)||_W \tag{5.21}$$

The way of solving this optimization problem is not specified in the GMM theory. It is left to the numerical optimization field.

The purpose of $W$ is to weigh the different conditions. Choosing $W = \mathrm{Id}_c$ leads to consider the classical euclidean norm and is equivalent to considering that all conditions should weigh the same. The intuition behind the fact that one may prefer another norm is that some conditions may be less informative, redundant, or more volatile in their empirical estimation. One typically wants to use the norm minimizing the asymptotic variance of the resulting estimator. This problem has a closed-form solution with the following theorem:

**Theorem 6.** *(Hansen 1982) Let $\hat{\theta}_N$ be the random variable representing the output of the GMM estimator with N data observations. Let also define $\Omega$ as the covariance matrix of the conditions function g evaluated at $\theta_0$: $\Omega = \text{cov-mat}\big(g(L, \theta_0)\big)$. Then,*

$$\underset{W}{\operatorname{argmin}} \lim_{N\to\infty} \operatorname{var}(\hat{\theta}_N) = \Omega^{-1} \tag{5.22}$$

In the particular case where conditions are independent the matrix $\Omega^{-1}$ is diagonal and choosing $W = \Omega^{-1}$ simply means that the moments' condition should be weighted inversely proportionally to their underlying variance. This is in line with the intuition that conditions with high variance are less informative.

### 5.4.2 Parallel with the JMR Attack

This section exhibits the similarities between the GMM and JMR. The core of the JMR attack relies on an estimation of the true parameters $\theta_0 \in \Theta = \mathbb{R}^d \times \mathbb{R}^r$ of a chosen statistical model (encoded in the choice of the basis) in order to explain the leakage of each share. Let $L = (L_1, \ldots, L_d)$ represents the observed leakage variable and $L_\theta$ the predicted leakage variable under the assumption of $\theta$ so that, under the assumption that the chosen model is correct, $L = L_{\theta_0}$.

Since the moment conditions in JMR depend on the value of another public variable $X$, let define, for each key hypothesis $k$, a condition function $g_k : \mathbb{R}^d \times \mathcal{X} \times \Theta \to \mathbb{R}^{|\mathcal{X}|}$ as:

$$g_k(\ell, x, \theta) = e_x \cdot \left( j\check{m}^{(\mathbf{1}_d)}_{L_\theta | Z=f(x,k)} - \prod_{i=1}^{d} (\ell_i - \bar{\mu}_i) \right) \tag{5.23}$$

where $e_x = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^{|\mathcal{X}|}$ stands for a vector of 0 with one 1 at position[3] $x$, $\mathbf{1}_d$ stands for a vector of $d$ ones : $\mathbf{1}_d = (1, \ldots, 1)$ and $j\check{m}^{(u)}_L$ is defined as in Equation 5.11. This definition of $g_k$ may seem very artificial but it is designed so that Equation 5.18 holds for the correct hypothesis $k = k^*$ (under the assumption that $L = L_{\theta_0}$):

$$\begin{aligned} \mathbb{E}[g_{k^*}(L, X, \theta_0)] &= \frac{1}{|\mathcal{X}|} \left( j\check{m}^{(\mathbf{1}_d)}_{L_{\theta_0} | Z=f(x,k^*)} - j\check{m}^{(\mathbf{1}_d)}_{L|X=x} \right)_{x \in \mathcal{X}} \\ &= 0 \end{aligned} \tag{5.24}$$

Therefore applying GMM with $g_{k^*}$ as condition function is sound while it is not for wrong key hypotheses. In fact this property is the one exploited by JMR since step 1 to 5 of JMR are equivalent to apply $|\mathcal{K}|$ GMM estimations, one for each of the $g_k$ condition functions, with $W = \mathrm{Id}_{|\mathcal{X}|}$.

### 5.4.3 Improving JMR Using GMM Theory

This section describes two ways of improving JMR using the GMM theory. The first one is generic and the second one focuses on the unbalanced masking schemes.

#### Using the Optimal Weighting Matrix

Since the GMM theory recommends to use $\Omega^{-1}$ as weighting matrix, one could ask if using the identity matrix was optimal. Indeed, the adversary typically wants to minimize the variance of the GMM estimator for the correct key $k = k^*$. Therefore it would be natural to replace the identity matrix with $\Omega^{-1}$ where $\Omega = \text{cov-mat}\big(g_{k^*}(L, X, \theta_0)\big)$. The problem is that $\Omega$ is hard to estimate with data since $\theta_0$ is unknown. The solution to this problem is usually to apply the so-called two-step estimator where an estimation of $\theta_0$ is first computed with JMR with a sub-optimal weighting matrix (for example the identity) which allows estimating $\Omega$ and eventually apply GMM with the latter estimation as weighting matrix. However, in our case, $\Omega$ does not depend on $\theta_0$ which makes the process easier. Indeed, the variance of the components of $g_{k^*}$ (and therefore the covariance matrix) only comes from the right term of Equation 5.23 which does not depend on $\theta$. Therefore the equation of $\Omega$ can be re-written as:

$$\Omega = \text{cov-mat}\left[ e_X \left( \prod_{i=1}^{d} (L_i - \bar{\mu}_i) \right) \right] \tag{5.25}$$

[3] Here $x \in \mathcal{X} = \mathbb{F}_2^m$ is seen as an element of $\mathbb{Z}/2^m\mathbb{Z}$.

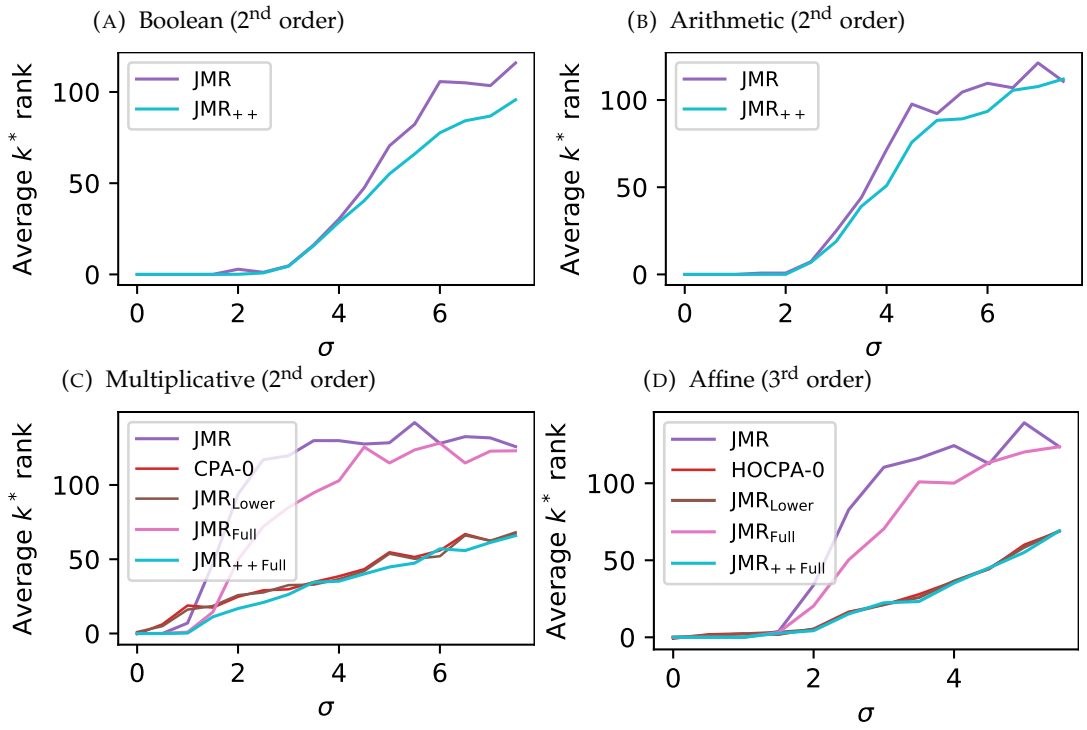



FIGURE 5.3: Guessing entropies for the improved JMR attacks, using the GMM theory, and for (HO)CPA-0 after the processing of A) $2^{16}$, B) $2^{16}$, C) $2^8 \times 255$, D) $2^{16} \times 255$ traces.

In addition, since for a fixed $x$, only one component of $g_{k^*}(\ell, x, \theta)$ is non-zero, $\Omega$ is diagonal. Then, one can estimate the diagonal terms of $\Omega$ using the observed data and then apply GMM. We denote by $\text{JMR}_{++}$ the JMR attack with $W = \bar{\Omega}^{-1}$ where $\bar{\Omega}$ is an estimation of the optimal weighting matrix.

To confirm the soundness of this approach, we performed the same experiments as those described in subsection 5.3.4 to compare JMR and $\text{JMR}_{++}$. Figures 3.3a and 3.3b show the results for the second-order Boolean and arithmetic masking and, according to the theory, $\text{JMR}_{++}$ performs a little better than JMR. It can be noticed that in the case of Boolean masking $\text{JMR}_{++}$ also outperforms HO-LRA-1, which has approximately the same performance as JMR, despite having more parameters to estimate.

### The Case of Biased Schemes

Some masking schemes, such as the multiplicative or the affine one, violate the assumption of shares uniformity. Therefore the resilience to $(d-1)^{th}$-order attack is not guaranteed anymore. For example, $Z = 0$ implies $Z_1 = 0$ in a multiplicative scheme inducing a first-order leakage. As well, when $Z = 0$, the affine scheme becomes a Boolean scheme of order 2 inducing second-order leakages. Since lower order JM are informative in these cases, a first idea to exploit this weakness is to apply JMR but at a lower-order. This means that the considered conditions concern only the first-order moments for a multiplicative scheme and the second-order JM

for an affine scheme. Such an attack is denoted $\text{JMR}_{\text{Lower}}$. Since this would only exploit the difference between the class $Z = 0$ and $Z \neq 0$ this attack would be very close to a CPA computed with a zero-valued model considering only two classes: $Z = 0$ and $Z \neq 0$, denoted CPA-0 (or HOCPA-0 in the affine case) afterward.

Figures 5.3c and 5.3d confirm this intuition by showing that both CPA-0 and $\text{JMR}_{\text{Lower}}$ behave very similarly and have better results than JMR for high noise values but worse results for low noise values. Indeed, since the main advantage of masking is to amplify the impact of the noise exponentially with the order of the mask [PR13] or more accurately, with the order of the attack required to defeat it. For low values of $\sigma$ the JM conditions used in $\text{JMR}_{\text{Lower}}$ are less informative than the one used by JMR (they only exploit a difference between the class $Z = 0$ and the other classes) but the impact of the noise is amplified by a lower order which explains the better results of $\text{JMR}_{\text{Lower}}$ for high $\sigma$.

A natural challenge is to design an attack benefiting from the best of both worlds: JMR and $\text{JMR}_{\text{Lower}}$. To this aim, we propose to use the flexibility of the GMM paradigm to develop an attack with conditions from both informative orders at the same time. This corresponds to building a system with 512 conditions instead of 256 when attacking a key byte. In this case, the weighting matrix is very important since each half of the system concerns conditions with very different variances (estimating joint moments is exponentially hard with the order). To highlight this fact we denote by $\text{JMR}_{\text{Full}}$ and $\text{JMR}_{++\text{Full}}$ the version of JMR with both order conditions respectively with $W = \text{Id}_{512}$ and $W = \bar{\Omega}^{-1}$.

Results are presented in figures 5.3c and 5.3d. As expected, $\text{JMR}_{\text{Full}}$ outperforms JMR but is impacted by the variance of the $d$-order conditions and therefore performs worse than $\text{JMR}_{\text{Lower}}$ for high values of $\sigma$. However, $\text{JMR}_{++\text{Full}}$ benefits from the advantage of exploiting the $d$-order conditions for low values of $\sigma$ but still converges towards $\text{JMR}_{\text{Lower}}$ for high noise values thanks to the well-chosen weighting of these conditions. Indeed, it is proven in [Han82] that adding more moments conditions can only improve the performance of the GMM estimator (by lowering its variance) when using the optimal weighting matrix $\Omega^{-1}$.

We highlight the fact that for the multiplicative scheme, there is an interest in using $\text{JMR}_{++\text{Full}}$ over CPA-0 since for example it would give a successful attack at $\sigma = 1$ where CPA-0 would rank the correct key at the $20^{th}$ position which is not enumerable considering the full 16 bytes key. However, for the affine case, the curves look very similar and we argue that the overhead in time complexity of using $\text{JMR}_{++\text{Full}}$ (or $\text{JMR}_{\text{Lower}}$) over CPA-0 is not worth it.

## 5.5 Experiments on Real Traces

To assess the performance of JMR on real traces, we decided to attack two open source protected AES implementations. The first one is protected by a first-order

Boolean masking scheme (ASCAD) [Ben+18]. The second one embeds an affine scheme and a shuffling countermeasure (ASCADv2) [Ben+19]).

### 5.5.1 Attack on a First-Order Boolean Masked AES (ASCAD)

As a first experiment, we performed the different stochastic attacks discussed in this paper on the public dataset of ASCAD. It is a common set of side-channel traces, introduced for research purposes on deep learning-based side-channel attacks. The targeted implementation is a software AES, protected with a first-order Boolean masking, running on an 8-bit ATMega8515 board.

We performed guessing entropies for the different attacks, using the training dataset containing 50k traces. We extracted from the dataset, the two Point-of-Interests (PoI) corresponding to the highest signal-to-noise ratio, one for each share. This step requires the knowledge of the shares and would not be feasible by a non-profiled adversary. In a real scenario, a visual analysis of the trace combined with knowledge on the implementation can be used to perform a PoI selection to reduce the number of sample combinations to be tested. Our goal here is to assess the security supposing that the adversary is able to apply the methodology on the best sample combination.

#### Results

Results are depicted in Figure 5.4. We observe similar outcomes than in the first-order Boolean simulations. As expected the results of $\text{JMR}_{++}$ and HO-LRA-1 are very close since it is a Boolean masking (and thus, Assumption 1 holds). Similarly to Figure 3.3a, the slight advantage of $\text{JMR}_{++}$ may be explained by the use of the optimal weighting matrix. One may notice that the HO-CPA performs better than in the simulations. It outperforms all the other attacks for low numbers of traces even though attacks with an average correct key rank higher than 25 does not allow for a successful enumeration in a reasonable time. The better performance of HO-CPA can be explained by the fact that the leakage model of the ASCAD traces is much closer to a Hamming weight leakage model than those used in the simulated experiments. In such cases, regression-based attacks benefits less from their genericity. As the execution time has a low dependency to the number of traces, running $\text{JMR}_{++}$ took approximately 10 seconds as in the simulations.

### 5.5.2 Attack of an open source Hardened AES implementation (ASCADv2)

As a second experiment, we decided to attack the second protected AES implementation proposed by the *Agence Nationale de la Sécurité des Systèmes d'Information* (ANSSI) [Ben+19]. They published a library implementing an AES-128 on an ARM Cortex-M4 architecture using state-of-the-art counter-measures. Indeed, this implementation uses an affine masking as well as random shuffling of independent

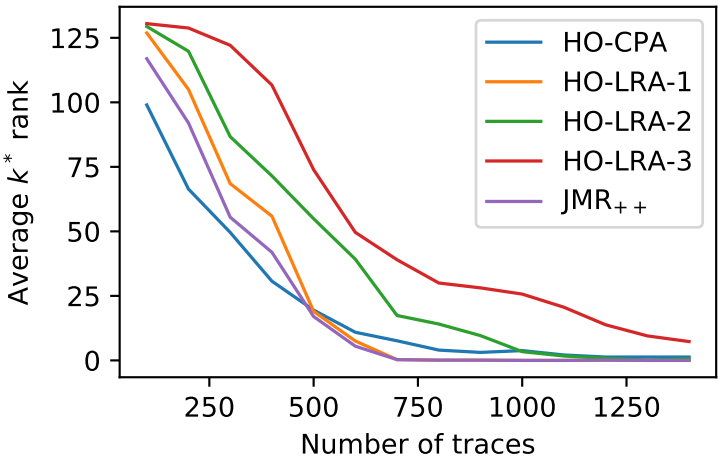



FIGURE 5.4: Comparison of different attacks' guessing entropies on ASCAD

operations [VC+12]. It is accompanied by a publicly available dataset called ASCADv2 providing 800,000 traces acquired on an STM32F303 microcontroller running this protected AES. A detailed leakage analysis of this dataset has been published in [MS21]. Following their terminology we tried to attack the unmasked variable $Z = Sbox[k^* \oplus P]$ using the leakage of the three shares:

$$\begin{aligned} Z_1 &= Z \otimes r_{\text{mul}} \oplus r_{\text{out}} \\ Z_2 &= r_{\text{mul}} \\ Z_3 &= r_{\text{out}} \end{aligned} \tag{5.26}$$

Unfortunately, the number of traces turned out to be too low to analyze the unsupervised attacks discussed in this paper. Thus, we reproduced a similar experimental setup, described in the next section, in order to collect significantly more traces.

#### Acquisition Setup

Our setup has the following features:

- The acquisitions have been performed on a NUCLEO-F303RE board, which embeds the same STM32F303 micro-controller as used in ASCADv2.
- The device is clocked at 8MHz, while ASCADv2 device is clocked a 4 MHz. This allows faster acquisitions without altering the execution behavior (e.g., introducing FLASH wait cycles). Being in an evaluation setup, we had the labels of the shares and validated that it did not affect the signal-to-noise ratio of these intermediate variables.
- We measured the magnetic field produced by the circuit with a Langer H-field probe (RF-U 5-2). This differs from ASCADv2 setup, which measures the current of the device through a ChipWhisperer [OC14]. However, we observed

better signal to noise ratios on the EM field. The probe covers a large portion of the CPU and no specific tuning of the probe placement was performed.

- The scope was configured at 3.125 GS/s and acquired a window of $8\mu s$, which represents 25,000 time samples.
- The masked AES implementation was taken "as-is" from the SecAESSTM32 repository [Ben+19]. We only made the following changes to the assembly code:
  - a GPIO is raised in the `Load_random` function, which manipulates $r_{\text{mul}}$ and $r_{\text{out}}$.
  - a GPIO is raised in the first round of the AES, just after the `Xor_Word` operation.

To further speed up the acquisitions, we do not transfer the plaintext and masking inputs through the serial port. Indeed, this represents 54 bytes ($16 + 19 \times 2$) per encryption, which quickly becomes a bottleneck for acquisitions. Instead, the device runs a PCG32 Pseudo-Random Number Generator (PRNG) [O'N14] to generate those data on the fly. This PRNG is re-seeded randomly (by sending 8 bytes on the serial port) every 250 acquisitions. This allows to regenerate (from the stored seeds) the plaintexts and random masks offline, to label the dataset.

For each encryption, the scope triggers twice and acquires 50,000 samples. The final dataset contains 100M traces and took 14 days to acquire. In summary, we used the same AES implementation and micro-controller as in the ASCADv2 setup. We only made some changes in the instrumentation and measurement chain to reduce the number of traces needed and improve the speed of acquisition.

#### Simulating an Unshuffled Version

The implementation uses random permutation of the 16 `Sboxes` applications. However, using the same idea as developed in technical analysis of the ANSSI repository [Ben+19], one can simulate (through the knowledge of the key and the permutation $Sh$ being used) an attack on an unshuffled version even if the acquired traces are shuffled. Instead of targeting the first byte $Z = Sbox[k^*[0] \oplus P[0]]$ one may target:

$$Z = Sbox[k^*[Sh^{-1}(0)] \oplus P[Sh^{-1}(0)]] \tag{5.27}$$

where $Sh^{-1}(0)$ denotes the index of the byte that is computed first through the permutation $Sh$. Then such an attack would uses $Z^{(\bar{k})} = Sbox[\bar{k} \oplus k^*[Sh^{-1}(0)] \oplus P[Sh^{-1}(0)]]$ as hypothesis intermediate variable, the attack being successful if the best hypothesis is 0.

**Results**

In a similar way to the first experiment from subsection 5.5.1, we extracted from the dataset described in section 5.5.2, the three Point-of-Interests (PoI) corresponding to the highest signal-to-noise ratio, one for each share. We performed the attacks on both the shuffled and unshuffled versions. The attacks on the shuffled version only use the leakage of the first `Sbox` computation. Shuffling adds a lot of noise since even for the correct key hypothesis the predicted value of $Z$ is only correct once out of 16 in average.

Results are presented in Figure 3.4. Each point represents the mean ranking of the correct key over 100 attacks performed with the corresponding number of traces. For each attack, traces are randomly drawn among the 100M dataset. Both $\text{JMR}_{++\text{Full}}$ and $\text{JMR}_{++}$ converge towards a guessing entropy of 0 which provides by the same token, the first unsupervised attack on the secured ANSSI's AES implementation.

**Using the scheme bias.** Not surprisingly, $\text{JMR}_{++\text{Full}}$ and HOCPA-0, which exploits the bias in the masking scheme, gives the best results. These attacks require 30k[4] and 15M traces to converge towards 0 for the unshuffled and shuffled version respectively. This confirms that for high noise value, a lower-order leakage induces attacks with at least one order of magnitude smaller data complexity. Thus, it confirms that even though $d$ shares are used to mask the sensitive value, a biased $d$-order masking should not be considered of order $d$ as far as security is concerned.

**Not using the scheme bias.** When this lower-order leakage is not considered in the attack, $\text{JMR}_{++}$ outperforms the other state-of-the-art attacks and is the only attack able to converge toward a guessing entropy of 0 with the considered number of traces. As in the simulations, the amount of time required to run this attack is approximately 15 seconds.

For biased masking schemes, there is no interest to perform this attack over CPA-0. However, we argue that this result is interesting since it shows how JMR would perform on a generic (with a high algebraic degree) unbiased second-order masking schemes.

### 5.5.3 Discussion

Results obtained on the real traces collected on AES implementations (proposed by the ANSSI) protected with boolean and affine masking are in line with the simulation results. It confirms that JMR gives a sound methodology, able to work with flexible leakage model assumptions (linear, quadratic...), which is applicable to any

[4] It should be noted that even if some of the presented attacks require less than 800k traces, they have not been successful on the original ASCADv2 dataset. We have confirmed that our traces have a better SNR on the leakage of each of the shares which could explain this difference.

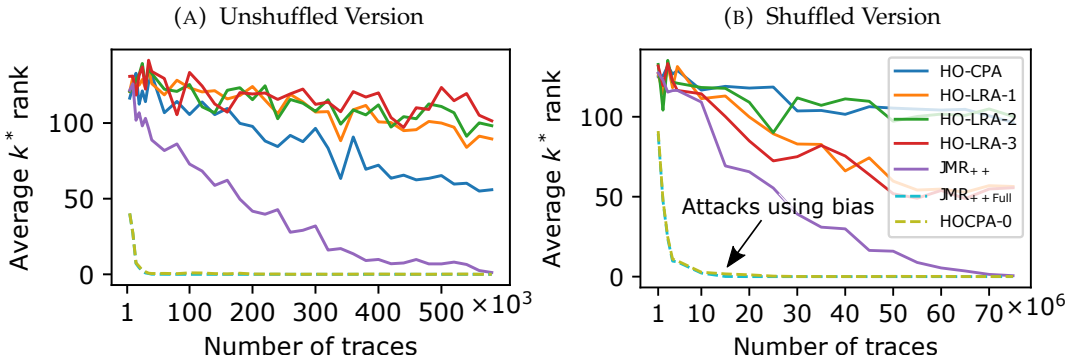



FIGURE 5.5: Comparison of different attacks' guessing entropies on the secured ANSSI's AES

masking scheme, even newly invented ones. Such strategy widens the state-of-the-art[5].

## 5.6 Conclusion

This chapter introduced a new unsupervised strategy, JMR, which embeds the masking structure within it, allowing it to defeat arbitrary masking schemes. It is based on a non-linear system regression which allows to derive the leakage model of each share by carefully exploiting higher-order joint moments conditions. JMR outperforms state-of-the-art attacks which are limited to Boolean and arithmetic masking, especially when the Hamming weight leakage assumption does not hold. We reduced the core of JMR into a more general framework: the generalized method of moments and derived optimizations of JMR from it. Experiments performed on synthetic data confirmed the effectiveness of the proposed attack, especially against multiplicative and affine masking schemes. Eventually, this new strategy has been confirmed on real traces, allowing a fully unsupervised attack of the ANSSI's protected AES implementation which embeds an affine masking and shuffling countermeasures.

The JMR method described in this chapter is not highly multi-dimensional in the sense that it only exploits $d$ times sample when applied on a $d^{th}$-order masking. It is well known that sensitive variables can leak several times in a single trace. Section 4.4.1 suggests that the EVIL machine produce an estimation of the joint moment per class under a key hypothesis. It is therefore possible in theory to bridge the two strategies to mount an unsupervised attack, able to work with a flexible *a priori*, agnostic to the masking scheme and which benefits from the advantage of deep learning techniques. Such a strategy would deserve a full analysis in a dedicated work which constitutes an interesting and stimulating line of research for future works.

[5] One may notice that the other attacks perform better than in the simulated experiments. We explain this by the fact that in this case, the leakage model is fixed and may be closer to a Hamming weight model.

# Chapter 6

# General Conclusion

This thesis investigated new unsupervised methods for side-channel analysis based on mutual information estimated in a high dimensional way. The starting point is that the mutual information between the full traces and the secret is a key quantity for SCA since it provides an absolute leakage quantification. Such a quantity has not really been studied before due to its computational intractability in high dimensional spaces: traces usually contains too many time samples for classical MI estimators to stay sound. However recent works from the pure machine learning community proposed a new MI estimator (MINE), based on deep learning techniques, that is supposed to scale with high dimensional variables.

The first contribution of this thesis, from Chapter 2, is to provide an analysis of MINE in an SCA context. We showed that with the correct representation of the input variable MINE is effectively able to estimate the MI between sensitive variables and full (or a significant portion of) traces. It is therefore able to automatically combines samples making the tool sound for masked implementations or for low signal cases where multiple sources have to be accumulated to detect a significant leakage. MINE appears to be a generic tool allowing to have an objective leakage evaluation from traces. As a result, it could be used by designers/evaluators as a sound leakage metric during their development/evaluation process.

As a second contribution, Chapter 3 presents NEMIA which is an extension of this new tool to the context of unsupervised attacks, aiming at recovering a cryptographic key. It revisits the classical mutual information analysis, in a multidimensional paradigm. It provides rigorous proofs whose goal are to derive the optimal MI-based attacks that deal with high-dimensional traces. NEMIA is the first unsupervised attack able to benefit at the same time from the framework of information theory and the potential of deep learning (multidimensional treatment of traces, reduces the need for preprocessing, adaptable architectures...). Being able to combine information from multiple sources, NEMIA outperforms classical strategies and may be worth considering especially in low information/high noise scenarios, where all (or most) of the available information contained in the trace needs to be exploited to conduct a successful attack.

From a practical point of view, the two main drawbacks of NEMIA are: its computational complexity, requiring as many network trainings as there are key candidates, and its requirement for a strong leakage model *a priori*. Chapter 4 proposes, as a third contribution, a new attack strategy, the EVIL Machine Attack (EMA), which is built to overcome these two problems. The EVIL machine is a network architecture, inspired by generative adversarial networks, whose aim is to produce a representation of the leakage model of the device. It is derived as an unsupervised attack combining techniques from the stochastic attacks which allows for a more flexible *a priori*. In addition, EMA requires only one network training, which may be a game changer regarding unsupervised deep learning-based attacks. Simulations and real case experiments confirmed its practicable applicability showing significant improvement compared to classical stochastic attacks.

EMA applied in the context of masking raised questions about higher-order generalizations of stochastic attacks. The last contribution, provided by Chapter 5, is an analysis of such generalizations along with a new proposal based on non-linear regressions of the joint moments of the leakage distribution (JMR). Its main advantage is to be agnostic to the masking scheme. We showed that state-of-the-art attack exploits specificities of the Boolean and arithmetic scheme and do not provide a generic strategy. JMR is based on an estimation of the joint moment per class. We showed that EMA applied on masked traces produces a representation of such joint moment per class while benefiting from the deep learning advantages. Therefore, EMA and JMR can be combined in an unsupervised attack strategy able to use the potential of deep learning with a flexible leakage model *a priori* and whatever the type of masking scheme. We provided a proof of concept of such a combination. However, such a combination deserves a more in-depth characterization and may constitute an interesting and challenging path for future research.

One of the main objectives of this thesis was to provide theoretical bases as well as new unsupervised attack propositions which satisfied certain properties (aka. multidimensional treatment of the traces, deep-learning potential, flexibility on the leakage model *a priori*, genericity regarding the masking scheme...). Throughout this thesis, neural networks have been seen mostly as a black box that accomplishes the task encoded in their loss function. We have kept the network architectures as simple as possible and did not dive into fine parameters tuning or analysis of the different architectures related to the latest deep-learning progress. A more practical study, focused on the DL side, would be valuable and may improve the real case performances. Indeed such techniques help the networks to converge better and faster towards their objectives, reducing the so called optimization error in the learning process.

# Appendix A

# Proofs for chapter 3

## A.1 Proofs of Lemma 1

**Lemma 1.** *Let $f$: $\mathcal{Z} \to \mathbb{R}^n$ be any function. For any leakage model $\varphi$: $\mathcal{Z} \to \mathbb{R}^n$ there exists a decomposition of $f$ into $f = f_2 \circ f_1$, with $f_1 : \mathcal{Z} \to \mathbb{N}$, $f_2 : \mathbb{N} \to \mathbb{R}^n$, satisfying the two following properties:*

1) $\exists\, f_3 : \mathrm{Im}\, f_1 \to \mathbb{R}^n$ *such that* $f_3 \circ f_1 = \varphi$

2) $\forall z \in \mathcal{Z}, f_2|_{f_1\left(\varphi^{-1}(\{\varphi(z)\})\right)}$ *is bijective of reciprocal* $f_2^{-1}|_{f_2 \circ f_1\left(\varphi^{-1}(\{\varphi(z)\})\right)}$

*Proof.* Let us create a partition of $\mathcal{Z} = \sqcup_{i=1}^{n} P_i$ where two elements $z_1, z_2 \in \mathcal{Z}$ are in the same $P_i$ if and only if:

- $\varphi(z_1) = \varphi(z_2)$
- $f(z_1) = f(z_2)$

Then, one may define $f_1$ as $f_1(z) = i, \forall z \in P_i$. Since $f_1$ only collides for $z$ that already collides through $\varphi$, there exists $f_3$ such that $f_3 \circ f_1 = \varphi$. As $f$ is constant on $P_i$, let us denote by $v_i$ its output on elements of $P_i$. Then $f_2$ can be defined as $f_2(i) = v_i$ so that $f_2 \circ f_1 = f$. Now let us prove 2). Let $z \in \mathcal{Z}$ and $a, b \in f_1(\varphi^{-1}(\{\varphi(z)\}))$ such that $f_2(a) = f_2(b)$. There exists $z_a$ and $z_b$ such that $a = f_1(z_a)$ and $b = f_1(z_b)$ with $\varphi(z_a) = \varphi(z_b) = \varphi(z)$. So:

- $\varphi(z_a) = \varphi(z_b)$
- $f_2(f_1(z_a)) = f_2(f_1(z_b)) \iff f(z_a) = f(z_b)$

which means that $z_a$ and $z_b$ are in the same $P_i$ and thus collides through $f_1$. So $a = b$ which proves that $f_2|_{f_1(\varphi^{-1}(\{\varphi(z)\}))}$ is injective. Then, considering its set of destination being its image, one can say that this function is bijective with reciprocal function: $f_2^{-1}|_{f_2 \circ f_1(\varphi^{-1}(\{\varphi(z)\}))}$. □

## A.2 Proof of Corollary 1

**Definition 1.** *A function $f$ is said wider- than $g$ if there exists another function $h$ such that: $h \circ f = g$.*

**Corollary 1.** *Let $L$ be defined as in Equation 3.43. Then, for any function $\bar{h}$ wider than $HW$, $\mathcal{S}_{HW} \geq \mathcal{S}_{\bar{h}}$.*

*Proof.* There exists $h$ such that $h \circ \bar{h} = HW$. So:

$$\begin{aligned} \mathcal{S}_{HW} &= \mathcal{I}\big(HW(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(HW(Z_k), L\big)\big] \\ &= \mathcal{I}\big(h \circ \bar{h}(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(h \circ \bar{h}(Z_k), L\big)\big] \end{aligned} \tag{A.1}$$

Since removing $h$ in the second term can only increase the information:

$$\mathcal{S}_{HW} \geq \mathcal{I}\big(h \circ \bar{h}(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(\bar{h}(Z_k), L\big)\big] \tag{A.2}$$

By Theorem 5, $HW$ maximizes over $g$ the quantity: $\mathcal{I}\big(g(Z_{k^*}), L\big)$, so removing $h$ in the first term cannot increase the information:

$$\begin{aligned} &\mathcal{S}_{HW} \geq \mathcal{I}\big(\bar{h}(Z_{k^*}), L\big) - \max_{k \neq k^*} \big[\mathcal{I}\big(\bar{h}(Z_k), L\big)\big] \\ &\mathcal{S}_{HW} \geq \mathcal{S}_{\bar{h}} \end{aligned} \tag{A.3}$$

□

## A.3 Complementary material on the entropy

**Lemma 2.** *Let $A$ and $B$ be a two discrete random variables. Let $f\colon \mathcal{A} \to \mathbb{R}^n$ be any function. Then:*

$$\mathcal{H}\big(f(A) \mid B\big) \leq \mathcal{H}(A \mid B) \tag{A.4}$$

*Proof.* The data processing inequality [BR12] ensures that applying $f$ to any variables can not increase its mutual information with another variable so:

$$\begin{aligned} \mathcal{I}\big(f(A), f(A) \mid B\big) &\leq \mathcal{I}(A, A \mid B) \\ \mathcal{H}\big(f(A) \mid B\big) &\leq \mathcal{H}(A \mid B) \end{aligned} \tag{A.5}$$

□

**Lemma 3.** *Let $A$ and $B$ be a two discrete random variables. Let $f\colon \mathcal{A} \to \mathbb{R}^n$ be any function. Then:*

$$\mathcal{H}\big(A \mid f(B)\big) \geq \mathcal{H}(A \mid B) \tag{A.6}$$

*Proof.* Again, using the data processing inequality [BR12]:

$$\begin{aligned} \mathcal{I}\big(A, f(B)\big) &\leq \mathcal{I}(A, B) \\ \mathcal{H}(A) - \mathcal{H}\big(A \mid f(B)\big) &\leq \mathcal{H}(A) - \mathcal{H}(A \mid B) \\ \mathcal{H}\big(A \mid f(B)\big) &\geq \mathcal{H}(A \mid B) \end{aligned} \tag{A.7}$$

□

## A.4 Proof of Theorem 5 at Order $n$

**Theorem 7.** *Let $L$ represent the leakage of an intermediate variable $Z_{k^*}$ protected by a boolean masking of order $n$. Let all shares follow any bijection $b_0, \dots, b_n$ of a Hamming weight leakage such that:*

$$L = [b_0\big(HW(Z_k^* \oplus M_1 \oplus \cdots \oplus M_n)\big), b_1\big(HW(M_1)\big), \dots, b_n\big(HW(M_n)\big)] \tag{A.8}$$

$$\textit{Then, } \mathcal{I}\big(HW(Z_{k^*}), L\big) = \mathcal{I}\big(Z_{k^*}, L\big).$$

*Proof.* Since bijective transformations do not impact mutual information, one can consider without loss of generality that:

$$L = L = [HW(Z_k^* \oplus M_1 \oplus \cdots \oplus M_n), HW(M_1), \dots, HW(M_n)] \tag{A.9}$$

Now let us evaluate $\mathcal{I}\big(f(Z_{k^*}), L\big)$:

$$\mathcal{I}\big(f(Z_{k^*}), L\big) = \sum_{\bar{f} \in f(\mathcal{Z})} \sum_{l \in \mathcal{L}} P(\bar{f}, l) \cdot log\left(\frac{P(\bar{f}, l)}{P(\bar{f}) \cdot P(l)}\right) \tag{A.10}$$

One can split the first sum by summing on $z$ instead of $\bar{f}$:

$$\begin{aligned} \mathcal{I}\big(f(Z_{k^*}), L\big) &= \sum_{z \in \mathcal{Z}} \sum_{l \in \mathcal{L}} P(z, l) \cdot log\left(\frac{P(l \mid f(z))}{P(l)}\right) \\ &= \sum_{z \in \mathcal{Z}} \sum_{l \in \mathcal{L}} P(z) \cdot P(l \mid z) \cdot log\left(\frac{P(l \mid f(z))}{P(l)}\right) \end{aligned} \tag{A.11}$$

Since the identity function is bijective and maximizes this quantity, it would be enough to show that $P(l \mid HW(z)) = P(l \mid z)$.

Let define the following propositions $\mathcal{P}_n$ for $n \geq 1$:

$$\mathcal{P}_n : \quad \begin{array}{l} \forall\, z, m_1, \dots, m_n \in \mathcal{Z}, \text{let} \\ l = [HW(z \oplus m_1 \oplus \cdots \oplus m_n), HW(m_1), \dots, HW(m_n)] \\ P(l \mid HW(z)) = P(l \mid z) \end{array} \tag{A.12}$$

Let us prove $\mathcal{P}_n$ for any $n$ by induction. $\mathcal{P}_1$ is true by Th.2. Let suppose that $\mathcal{P}_n$ is true for a given $n$. Now, $\forall z, m_1, \dots, m_{n+1} \in \mathcal{Z}$ let us define $l$:

$$l = [HW(z \oplus m_1 \oplus \cdots \oplus m_{n+1}), HW(m_1), \dots, HW(m_{n+1})] \tag{A.13}$$

Now by independence of the shares:

$$P(l \mid z) = \prod_{i}^{n+1} \big[P(HW(m_i)\big] \cdot P\big(HW(\overbrace{z \oplus m_1 \oplus \cdots \oplus m_{n-1}}^{\bar{z}} \oplus \overbrace{m_n \oplus m_{n+1}}^{\bar{m}}) \mid \overbrace{z, HW(m_1), \ldots HW(m_{n-1})}^{K_1}, \overbrace{HW(m_n), HW(m_{n+1})}^{K_2}\big) \tag{A.14}$$

let us evaluate the right term by summing over all the possible value of $HW(\bar{m})$:

$$P\big(HW(\bar{z} \oplus \bar{m}) \mid K_1, K_2\big) = \sum_{HW(\bar{m})} P\big(HW(\bar{m}) \mid K_2\big) \cdot \overbrace{P\big(HW(\bar{z} \oplus \bar{m}) \mid K_1, K_2, HW(\bar{m})\big)}^{A} \tag{A.15}$$

One can remove $K_2$ in $A$ since $K_2$ does not add any information when $HW(\bar{m})$ is known:

$$\begin{aligned} A &= P\big(HW(\bar{z} \oplus \bar{m}) \mid K_1, HW(\bar{m})\big) \\ &= P\big(HW(z \oplus \cdots \oplus m_{n-1} \oplus \bar{m}) \mid z, HW(m_1), \ldots, HW(m_{n-1},), HW(\bar{m})\big) \\ &= P\big(HW(z \oplus \cdots \oplus m_{n-1} \oplus \bar{m}) \mid HW(z), HW(m_1), \ldots, HW(m_{n-1},), HW(\bar{m})\big) \end{aligned} \tag{A.16}$$

by induction property $\mathcal{P}_n$. Plug-in this into eq. A.15 and then in eq. A.14 gives:

$$P(l \mid z) = P(l \mid HW(z)) \tag{A.17}$$

which concludes the proof. □

# Appendix B

# Networks Architecture for chapter 3

Figure B.1 and Figure B.2 show the network architectures used for the experiments performed respectfully with MINE and classifiers (supervised and DDLA). For fairness, we tried to keep the two architectures as close as possible. The optimizer used in both cases is Adam [KB14] with default parameters. The loss function used for the classifiers is the categorical cross-entropy. Note that when using convolutional layers with MINE, the convolutional layers should only be applied to the trace variable and not to $f(Z_k)$ which would not make sense.

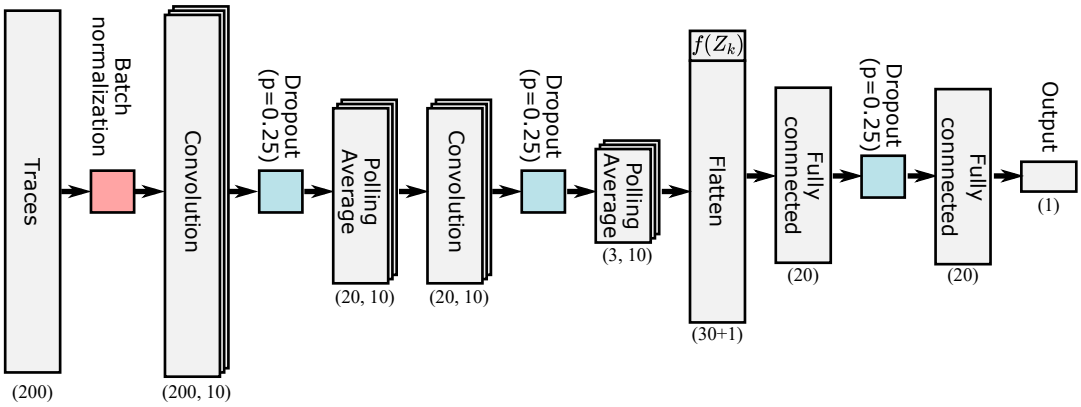



FIGURE B.1: Network architecture for MINE

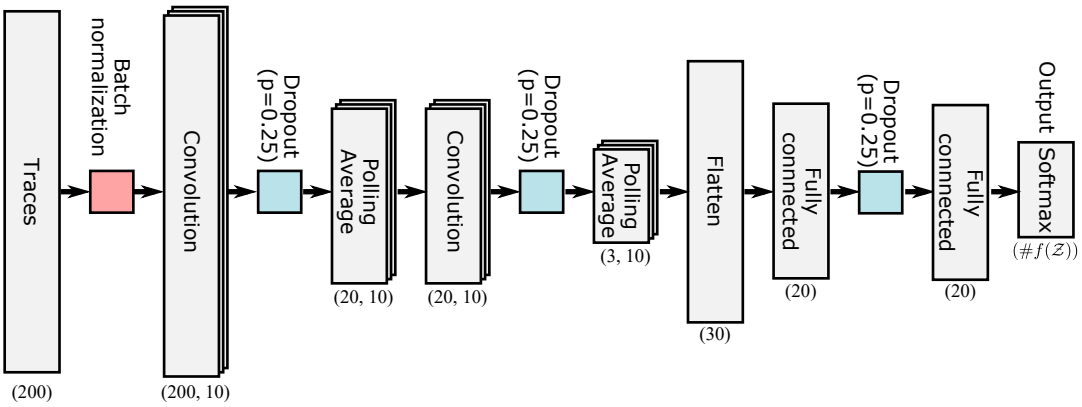



FIGURE B.2: Network architecture for the classifiers (Supervised and DDLA)

# Appendix C

# Networks Architecture for chapter 4

Figure C.1 and Figure C.2 show the network architectures used for all the the experiments of this paper. The optimizer used is Adam [KB14] with default parameters. The convolutional and fully connected layers use the exponential linear unit (ELU) as activation function.

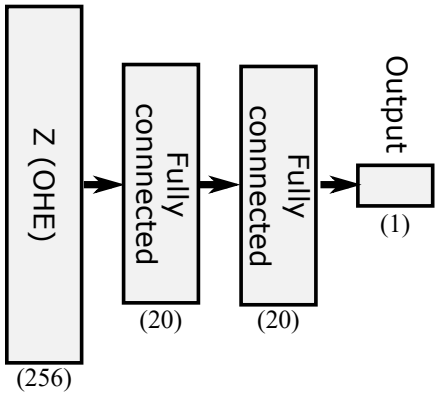



FIGURE C.1: Architecture of the encoder E

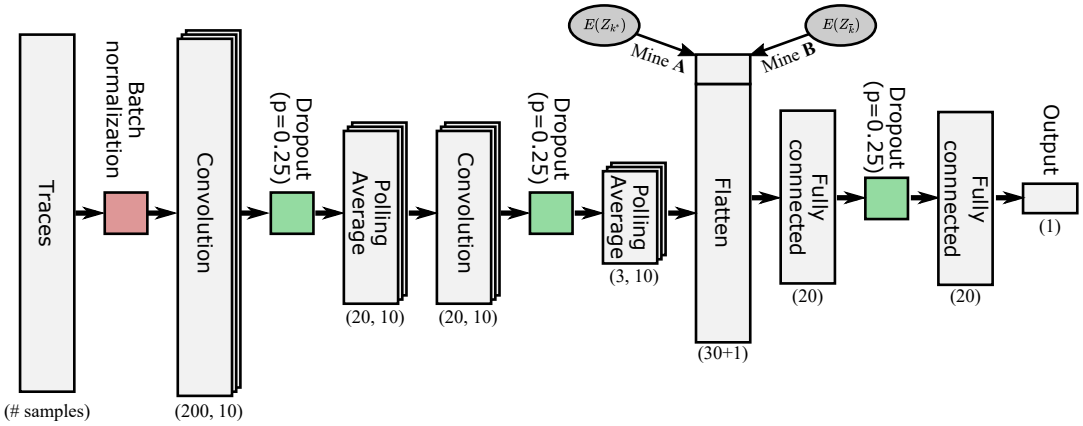



FIGURE C.2: Architecture of MINE **A** and **B**

# Appendix D

# Proofs for chapter 5

## D.1 Proof of Proposition 5

**Proposition 5.** *(Boolean masking) Let $* = \oplus$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degree 1. Let $\varphi_{Bool}(z) = \mathrm{cov}\big(l_1(z \oplus Z_2), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Bool}) \leq 1 \tag{D.1}$$

*Proof.* Since both $l_1$ and $l_2$ are of degree 1, there exist two unique sets of coefficients $(\alpha_i^{(1)})_{0 \leq i \leq n} \in \mathbb{R}$ and $(\alpha_i^{(2)})_{1 \leq i \leq n} \in \mathbb{R}$ such that:

$$l_j(z) = \alpha_0^{(j)} + \sum_{i=1}^{n} \alpha_i^{(j)} \cdot z[i] \tag{D.2}$$

where $z[i]$ stands for the $i^{th}$ bit of $z$. Since the covariance involves a centered product, one can suppose without loss of generality that $\alpha_0^{(j)} = 0$ (we removed $\alpha_0^{(j)}$ for readability reasons but it does not change anything to the proof). Injecting Equation D.2 into the expression of $\varphi_{Bool}$:

$$\begin{aligned} \varphi_{Bool}(z) &= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z \oplus z_2)[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) \\ &= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z[i] \oplus z_2[i]) - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) \end{aligned} \tag{D.3}$$

Using the identity : $z[i] \oplus z_2[i] = z[i] + z_2[i] - 2 \cdot (z[i] \wedge z_2[i])$ where $\wedge$ stands for the Boolean AND:

$$\begin{aligned}
\varphi_{Bool}(z) &= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \sum_{i=1}^{n} \big( \alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} \sum_{z_2 \in \mathcal{Z}} \big( \alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=0}} \big( \alpha_i^{(1)} \cdot (z[i] + z_2[i]) - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) + \\
&\quad \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big( \alpha_i^{(1)} \cdot (-z[i] + z_2[i]) - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} z[i] \cdot \Big[ \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big( \alpha_i^{(1)} \cdot z_2[i] - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) - \\
&\qquad \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big( \alpha_i^{(1)} \cdot z_2[i] - \mu_1 \big) \cdot \big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \big) \Big]
\end{aligned} \tag{D.4}$$

which is of degree at most 1 since the $z[i]$ terms are not mixed. □

## D.2 Proof of Proposition 6

**Proposition 6.** *(Arithmetic masking) Let $* = + \bmod 2^n$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degree 1. Let $\varphi_{Arith}(z) = \mathrm{cov}\big(l_1(z + Z_2\ [2^n]), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Arith}) \leq 2 \tag{D.5}$$

*Proof.* We give a proof by induction. Let define the property $\mathcal{P}_n$:

$\mathcal{P}_n$ : For any $l_1$ and $l_2$ of degree 1, $deg(\varphi_n) \leq 2$, where for $z \in \mathcal{Z} = \mathbb{F}_2^n$:

$$\varphi_n(z) = \mathrm{cov}\big(l_1(z + Z_2\ [2^n]), l_2(Z_2)\big)$$

**Initialisation.** The case $n = 1$ is trivial since $deg(\varphi_{Arith})$ is at most 1 in this case.

**Induction.** Let suppose that $\mathcal{P}_n$ holds. We are going to prove that $\mathcal{P}_{n+1}$ also holds. Since both $l_1$ and $l_2$ are of degree 1, there exists two unique sets of coefficients $(\alpha_i^{(1)})_{0 \leq i \leq n+1} \in \mathbb{R}$ and $(\alpha_i^{(2)})_{0 \leq i \leq n+1} \in \mathbb{R}$ such that:

$$l_j(z) = \alpha_0^{(j)} + \sum_{i=1}^{n+1} \alpha_i^{(j)} \cdot z[i] \tag{D.6}$$

where $z[i]$ stands for the $i^{th}$ bit of $z$. Since the covariance involves a centered product, one can suppose without loss of generality that $\alpha_0^{(j)} = 0$ (we removed $\alpha_0^{(j)}$ for readability reasons but it does not change anything to the proof). Injecting this into the expression of $\varphi_{n+1}$ one has:

$$\varphi_{n+1}(z) = \sum_{z_2=0}^{2^{n+1}-1} \Big( \sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2\ [2^{n+1}])[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) \tag{D.7}$$

for $i \in [\![1, n+1]\!]$, the following identity holds: $(z + z_2\ [2^{n+1}])[i] = (z + z_2)[i]$. Indeed, the modulo corresponds to either doing nothing or subtracting $2^{n+1}$ when $z + z_2 \geq 2^{n+1}$. Then:

$$\begin{aligned}
\varphi_{n+1}(z) &= \sum_{z_2=0}^{2^{n+1}-1} \Big( \sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) \\
&= \sum_{z_2=0}^{2^{n}-1} \Big( \sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) + \\
&\quad \sum_{z_2=2^n}^{2^{n+1}-1} \Big( \sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) \\
&= \sum_{z_2=0}^{2^{n}-1} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2)[i] + \alpha_{n+1}^{(1)} \cdot (z + z_2)[n+1] - \mu_1 \Big) \cdot \\
&\quad \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i]\ + \alpha_{n+1}^{(2)} \cdot z_2[n+1] - \mu_2 \Big) + \\
&\quad \sum_{z_2=2^n}^{2^{n+1}-1} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2)[i] + \alpha_{n+1}^{(1)} \cdot (z + z_2)[n+1] - \mu_1 \Big) \cdot \\
&\quad \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] + \alpha_{n+1}^{(2)} \cdot z_2[n+1] - \mu_2 \Big)
\end{aligned} \tag{D.8}$$

Again, one can add a $[2^n]$ in the $(z + z_2)[i]$ terms since it does not change anything for $i \in [\![1, n]\!]$. Then:

$$\begin{aligned}
\varphi_{n+1}(z) &= \sum_{z_2=0}^{2^{n}-1} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2\ [2^n])[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i]\ - \mu_2 \Big) + \\
&\quad \sum_{z_2=2^n}^{2^{n+1}-1} \Big( \sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2\ [2^n])[i] - \mu_1 \Big) \cdot \Big( \sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2 \Big) + \\
&\quad \sum_{z_2=0}^{2^{n+1}-1} \big( \alpha_{n+1}^{(1)} \cdot (z + z_2)[n+1] \big) \cdot \big( \alpha_{n+1}^{(2)} \cdot z_2[n+1] \big)
\end{aligned} \tag{D.9}$$

The second line of Equation D.9 can be re-indexed summing from 0 to $2^n - 1$. Then, by $\mathcal{P}_n$, the first two line of Equation D.9 are of degree at most 2. So let us focus on

the last term denoted $A$ and prove that it is also of degree at most 2:

$$\begin{aligned} A &= \sum_{z_2=0}^{2^{n+1}-1} \left(\alpha_{n+1}^{(1)} \cdot (z+z_2)[n+1]\right) \cdot \left(\alpha_{n+1}^{(2)} \cdot z_2[n+1]\right) \\ &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot \sum_{z_2=2^n}^{2^{n+1}-1} (z+z_2)[n+1] \end{aligned} \tag{D.10}$$

since $z_2[n+1] = 0$ implies that all the term in the sum are equal to 0.
One can notice that the latter sum has two expression depending on the $(n+1)^{th}$ bit of $z$:

$$\sum_{z_2=2^n}^{2^{n+1}-1} (z+z_2)[n+1] = \begin{cases} 2^n - z & \text{if } z[n+1] = 0 \\ z - 2^n & \text{if } z[n+1] = 1 \end{cases} \tag{D.11}$$

Therefore:

$$\begin{aligned} A &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot (z - 2^n) \cdot (2 \cdot z[n+1] - 1) \\ &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot (\sum_{k=1}^{n+1} 2^{k-1} \cdot z[k] - 2^n) \cdot (2 \cdot z[n+1] - 1) \end{aligned} \tag{D.12}$$

which is of degree at most 2 since developing the latter sum involves product of at most 2 bits of $z$ together.

Injecting this into Equation D.9 show that $deg(\varphi_{n+1}) \leq 2$ and therefore that $\mathcal{P}_{n+1}$ holds. This concludes the induction and therefore the proof of Proposition 6.

For the interested reader, we give as a bonus the coefficients of $\varphi_{Arith}$ in terms of $\alpha_i^{(j)}$:

$$\varphi_{Arith} = \alpha_0 + \sum_{i=1}^{n} \alpha_i \cdot z[i] + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \alpha_{i,j} \cdot z[i]z[j] \tag{D.13}$$

With:

$$\begin{aligned} \alpha_0 &= \frac{1}{4} \cdot \sum_{k=1}^{n} \alpha_k^{(1)} \alpha_k^{(2)} \\ \alpha_i &= -\sum_{k=1}^{i} \frac{\alpha_k^{(1)} \alpha_k^{(2)}}{2^{i-k}}, \text{ for } 1 \leq i \leq n \\ \alpha_{i,j} &= \frac{\alpha_i^{(1)} \alpha_i^{(2)}}{2^{j-i}}, \text{ for } 1 \leq i < j \leq n \end{aligned} \tag{D.14}$$

□

# Bibliography


[al.15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[Bal+12] Josep Balasch et al. "Theory and Practice of a Leakage Resilient Masking Scheme". In: *ASIACRYPT*. Vol. 7658. Springer, 2012, pp. 758–775. DOI: 10.1007/978-3-642-34961-4_45. URL: https://www.iacr.org/archive/asiacrypt2012/76580746/76580746.pdf.

[Bat+11] Lejla Batina et al. "Mutual Information Analysis: A Comprehensive Study". In: *J. Cryptol.* 24.2 (Apr. 2011), 269–291. ISSN: 0933-2790. DOI: 10.1007/s00145-010-9084-8. URL: https://doi.org/10.1007/s00145-010-9084-8.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. "Correlation Power Analysis with a Leakage Model". In: *Cryptographic Hardware and Embedded Systems - CHES 2004*. Ed. by Marc Joye and Jean-Jacques Quisquater. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[Bel+18a] Mohamed Ishmael Belghazi et al. *MINE: Mutual Information Neural Estimation*. 2018. arXiv: 1801.04062 [cs.LG].

[Bel+18b] Nicolas Belleville et al. "Automated Software Protection for the Masses Against Side-Channel Attacks". In: *ACM Transactions on Architecture and Code Optimization* 15.4 (Dec. 2018), 47:1–47:27. DOI: 10.1145/3281662. URL: https://hal.sorbonne-universite.fr/hal-01927625.

[Ben+18] Ryad Benadjila et al. "Study of deep learning techniques for side-channel analysis and introduction to ASCAD database". In: *ANSSI, France & CEA, LETI, MINATEC Campus, France.* (2018).

[Ben+19] Ryad Benadjila et al. *Hardened library for aes-128 encryption/decryption on arm cortex m4 achitecture*. https://github.com/ANSSI-FR/SecAESSTM32. 2019.

[Bha+14] Shivam Bhasin et al. "NICV: Normalized inter-class variance for detection of side-channel leakage". In: *2014 International Symposium on Electromagnetic Compatibility, Tokyo*. 2014, pp. 310–313.

[BLA79] G. R. BLAKLEY. "Safeguarding cryptographic keys". In: *1979 International Workshop on Managing Requirements Knowledge (MARK)*. 1979, pp. 313–318. DOI: 10.1109/MARK.1979.8817296.

[BR12] Normand J. Beaudry and Renato Renner. *An intuitive proof of the data processing inequality*. 2012. arXiv: 1107.0740 [quant-ph].

[Bro+19] Olivier Bronchain et al. *Leakage Certification Revisited: Bounding Model Errors in Side-Channel Security Evaluations*. Cryptology ePrint Archive, Report 2019/132. https://eprint.iacr.org/2019/132. 2019.

[BS19] Olivier Bronchain and François-Xavier Standaert. *Side-Channel Countermeasures' Dissection and the Limits of Closed Source Security Evaluations*. Cryptology ePrint Archive. https://eprint.iacr.org/2019/1008. 2019.

[BT11] Billy Bob Brumley and Nicola Tuveri. "Remote Timing Attacks Are Still Practical". In: *Computer Security – ESORICS 2011*. Ed. by Vijay Atluri and Claudia Diaz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 355–371.

[Car07] Claude Carlet. "Boolean Functions for Cryptography and Error Correcting Codes". In: (Nov. 2007).

[CDP17] Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures". In: *Cryptographic Hardware and Embedded Systems – CHES 2017*. Ed. by Wieland Fischer and Naofumi Homma. Cham: Springer International Publishing, 2017, pp. 45–68.

[CG00] Jean-Sébastien Coron and Louis Goubin. "On Boolean and Arithmetic Masking against Differential Power Analysis". In: *Cryptographic Hardware and Embedded Systems — CHES 2000*. Ed. by Çetin K. Koç and Christof Paar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 231–237. ISBN: 978-3-540-44499-2.

[Cha+19] Chung Chan et al. *Neural Entropic Estimation: A faster path to mutual information estimation*. 2019. arXiv: 1905.12957 [cs.IT].

[Cha+99] Suresh Chari et al. "Towards Sound Approaches to Counteract Power-Analysis Attacks". In: *Advances in Cryptology — CRYPTO' 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 398–412. ISBN: 978-3-540-48405-9.

[CK09] Jean-Sébastien Coron and Ilya Kizhvatov. "An Efficient Method for Random Delay Generation in Embedded Software". In: *Cryptographic Hardware and Embedded Systems - CHES 2009*. Ed. by Christophe Clavier and Kris Gaj. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 156–170. ISBN: 978-3-642-04138-9.

[CK10] Jean-Sébastien Coron and Ilya Kizhvatov. "Analysis and Improvement of the Random Delay Countermeasure of CHES 2009". In: *Cryptographic Hardware and Embedded Systems, CHES 2010*. Ed. by Stefan Mangard and

François-Xavier Standaert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 95–109. ISBN: 978-3-642-15031-9.

[CL20] Kwanghee Choi and Siyeong Lee. *Regularized Mutual Information Neural Estimation*. 2020. arXiv: 2011.07932 [cs.LG].

[CLH20] Valence Cristiani, Maxime Lecomte, and Thomas Hiscock. "A Bit-Level Approach to Side Channel Based Disassembling". In: *Smart Card Research and Advanced Applications*. Ed. by Sonia Belaïd and Tim Güneysu. Cham: Springer International Publishing, 2020, pp. 143–158. ISBN: 978-3-030-42068-0.

[Cri+22] Valence Cristiani et al. *Fit The Joint Moments - How to Attack any Masking Schemes*. Cryptology ePrint Archive, Paper 2022/927. https://eprint.iacr.org/2022/927. 2022. URL: https://eprint.iacr.org/2022/927.

[CRR02] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. "Template attacks". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. 2002.

[DDP13] Guillaume Dabosville, Julien Doget, and Emmanuel Prouff. "A New Second-Order Side Channel Attack Based on Linear Regression". In: *IEEE Transactions on Computers* 62.8 (2013), pp. 1629–1640. DOI: 10.1109/TC.2012.112.

[Dog+12] Julien Doget et al. "Univariate side channel attacks and leakage modeling". In: *Journal of Cryptographic Engineering* 1 (Apr. 2012), pp. 123–144. DOI: 10.1007/s13389-011-0010-2.

[EG12a] M. Elaabid and Sylvain Guilley. "Portability of templates". In: *Journal of Cryptographic Engineering* 2 (May 2012). DOI: 10.1007/s13389-012-0030-6.

[EG12b] M. Abdelaziz Elaabid and Sylvain Guilley. "Portability of templates". In: *Journal of Cryptographic Engineering* 2 (2012), pp. 63–74.

[EPW10] Thomas Eisenbarth, Christof Paar, and Björn Weghenkel. "Building a side channel based disassembler". In: *Transactions on computational science*. 2010.

[Fum+11] Guillaume Fumaroli et al. "Affine Masking against Higher-Order Side Channel Analysis". In: *Selected Areas in Cryptography*. Ed. by Alex Biryukov, Guang Gong, and Douglas R. Stinson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 262–280.

[Gie+08] Benedikt Gierlichs et al. "Mutual Information Analysis". In: *Cryptographic Hardware and Embedded Systems – CHES 2008*. Ed. by Elisabeth Oswald and Pankaj Rohatgi. Springer Berlin Heidelberg, 2008.

[GLRP06] Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. "Templates vs. Stochastic Methods". In: *Cryptographic Hardware and Embedded Systems - CHES 2006*. Ed. by Louis Goubin and Mitsuru Matsui. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 15–29. ISBN: 978-3-540-46561-4.

[Goo+14] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. DOI: 10.48550/ARXIV.1406.2661. URL: https://arxiv.org/abs/1406.2661.

[GP08] Martin Goldack and Ing Christof Paar. "Side-channel based reverse engineering for microcontrollers". In: *Master's thesis, Ruhr-Universität Bochum, Germany* (2008).

[GPQ11] Laurie Genelle, Emmanuel Prouff, and Michaël Quisquater. "Thwarting Higher-Order Side Channel Analysis with Additive and Multiplicative Maskings". In: *Cryptographic Hardware and Embedded Systems – CHES 2011*. Ed. by Bart Preneel and Tsuyoshi Takagi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 240–255.

[Han82] Lars Peter Hansen. "Large Sample Properties of Generalized Method of Moments Estimators". In: *Econometrica* 50.4 (1982), pp. 1029–1054. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912775 (visited on 04/10/2022).

[Har+20] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[HSW89] K. Hornik, M. Stinchcombe, and H. White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366.

[IPS02] J. Irwin, D. Page, and N.P. Smart. "Instruction stream mutation for non-deterministic processors". In: *Proceedings IEEE International Conference on Application- Specific Systems, Architectures, and Processors*. 2002, pp. 286–295. DOI: 10.1109/ASAP.2002.1030727.

[KB14] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[Kel60] Henry J Kelley. "Gradient theory of optimal flight paths". In: *Ars Journal* 30.10 (1960), pp. 947–954.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. "Differential power analysis". In: *Annual International Cryptology Conference*. 1999.

[Koc96] Paul C Kocher. "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems". In: *Annual International Cryptology Conference*. 1996.

[KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. "Estimating mutual information". In: *Physical Review* (2004). URL: http://dx.doi.org/10.1103/PhysRevE.69.066138.

[Lev44] Kenneth Levenberg. "A METHOD FOR THE SOLUTION OF CERTAIN NON-LINEAR PROBLEMS IN LEAST SQUARES". In: *Quarterly of Applied Mathematics* 2.2 (1944), pp. 164–168. ISSN: 0033569X, 15524485. URL: http://www.jstor.org/stable/43633451 (visited on 04/06/2022).

[Lin+19] Xiao Lin et al. *Data-Efficient Mutual Information Neural Estimator*. 2019. arXiv: 1905.03319 [cs.LG].

[LPR13] Victor Lomné, Emmanuel Prouff, and Thomas Roche. "Behind the Scene of Side Channel Attacks". In: *Advances in Cryptology - ASIACRYPT 2013*. Ed. by Kazue Sako and Palash Sarkar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 506–525. ISBN: 978-3-642-42033-7.

[Man04] Stefan Mangard. "Hardware Countermeasures against DPA – A Statistical Analysis of Their Effectiveness". In: *Topics in Cryptology – CT-RSA 2004*. Ed. by Tatsuaki Okamoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 222–235. ISBN: 978-3-540-24660-2.

[Mar63] Donald W. Marquardt. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters". In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441. ISSN: 03684245. URL: http://www.jstor.org/stable/2098941 (visited on 04/06/2022).

[MDP19] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. "A Comprehensive Study of Deep Learning for Side-Channel Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2020 (2019).

[Mes00] Thomas S. Messerges. "Using Second-Order Power Analysis to Attack DPA Resistant Software". In: *Cryptographic Hardware and Embedded Systems — CHES 2000*. Ed. by Çetin K. Koç and Christof Paar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 238–251. ISBN: 978-3-540-44499-2.

[MMS01] David May, Henk L. Muller, and Nigel P. Smart. "Non-deterministic Processors". In: *Information Security and Privacy*. Ed. by Vijay Varadharajan and Yi Mu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 115–129. ISBN: 978-3-540-47719-8.

[Moo+02] S. Moore et al. "Improving smart card security using self-timed circuits". In: *Proceedings Eighth International Symposium on Asynchronous Circuits and Systems*. 2002, pp. 211–218. DOI: 10.1109/ASYNC.2002.1000311.

[Moo+04] Simon Moore et al. "Balanced Self-Checking Asynchronous Logic for Smart Card Applications". In: *Microprocessors and Microsystems* 27 (Aug. 2004), pp. 421–430. DOI: 10.1016/S0141-9331(03)00092-9.

[MOP10] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. 1st. Springer Publishing Company, Incorporated, 2010. ISBN: 1441940391.

[Mor+08] Amir Moradi et al. "Information Leakage of Flip-Flops in DPA-Resistant Logic Styles." In: *IACR Cryptology ePrint Archive* 2008 (Jan. 2008), p. 188.

[MPG05] Stefan Mangard, Thomas Popp, and Berndt M. Gammel. "Side-Channel Leakage of Masked CMOS Gates". In: *Topics in Cryptology – CT-RSA 2005*. Ed. by Alfred Menezes. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 351–365. ISBN: 978-3-540-30574-3.

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. "Breaking Cryptographic Implementations Using Deep Learning Techniques". In: *Security, Privacy, and Applied Cryptography Engineering*. Ed. by Claude Carlet, M. Anwar Hasan, and Vishal Saraswat. Cham: Springer International Publishing, 2016, pp. 3–26. ISBN: 978-3-319-49445-6.

[MS21] Loïc Masure and Rémi Strullu. *Side Channel Analysis against the ANSSI's protected AES implementation on ARM*. Cryptology ePrint Archive, Report 2021/592. 2021.

[MSQ08] François Macé, François-Xavier Standaert, and Jean-Jacques Quisquater. "Information Theoretic Evaluation of Side-Channel Resistant Logic Styles". In: vol. 2008. Jan. 2008, p. 5.

[MWM21] Thorben Moos, Felix Wegener, and Amir Moradi. "DL-LA: Deep Learning Leakage Assessment: A modern roadmap for SCA evaluations". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2021.3 (2021), 552–598. DOI: 10.46586/tches.v2021.i3.552-598. URL: https://tches.iacr.org/index.php/TCHES/article/view/8986.

[Nag+07] Sei Nagashima et al. "DPA Using Phase-Based Waveform Matching against Random-Delay Countermeasure". In: *2007 IEEE International Symposium on Circuits and Systems*. 2007, pp. 1807–1810. DOI: 10.1109/ISCAS.2007.378024.

[OC14] Colin O'Flynn and Zhizhang (David) Chen. "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research". In: *Constructive Side-Channel Analysis and Secure Design*. Ed. by Emmanuel Prouff. Cham: Springer International Publishing, 2014, pp. 243–260. ISBN: 978-3-319-10175-0.

[OM06] Elisabeth Oswald and Stefan Mangard. "Template Attacks on Masking—Resistance Is Futile". In: *Topics in Cryptology – CT-RSA 2007*. Ed. by Masayuki Abe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 243–256. ISBN: 978-3-540-69328-4.

[O'N14] Melissa E. O'Neill. *PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation*. Tech. rep. HMC-CS-2014-0905. Claremont, CA: Harvey Mudd College, Sept. 2014.

[Per05] Colin Percival. "Cache Missing for Fun and Profit". In: *In Proc. of BSDCan 2005*. 2005.

[PR09] Emmanuel Prouff and Matthieu Rivain. "Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis." In: Jan. 2009, pp. 499–518.

[PR13] Emmanuel Prouff and Matthieu Rivain. "Masking against Side-Channel Attacks: A Formal Security Proof". In: *Advances in Cryptology – EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 142–159. ISBN: 978-3-642-38348-9.

[PRB09] E. Prouff, M. Rivain, and R. Bevan. "Statistical Analysis of Second Order Differential Power Analysis". In: *IEEE Transactions on Computers* 58.6 (2009), pp. 799–811. DOI: 10.1109/TC.2009.15.

[Prz+17] Adrian Przybylski et al. "Gpufit: An open-source toolkit for GPU-accelerated curve fitting". In: (2017). DOI: 10.1101/174110.

[PSG16] Romain Poussier, François-Xavier Standaert, and Vincent Grosso. "Simple Key Enumeration (and Rank Estimation) Using Histograms: An Integrated Approach". In: *Cryptographic Hardware and Embedded Systems – CHES 2016*. Ed. by Benedikt Gierlichs and Axel Y. Poschmann. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 61–81. ISBN: 978-3-662-53140-2.

[QS01] Jean-Jacques Quisquater and David Samyde. "Electromagnetic analysis: Measures and counter-measures for smart cards". In: 2001.

[RGV14a] Oscar Reparaz, Benedikt Gierlichs, and Ingrid Verbauwhede. "A Note on the Use of Margins to Compare Distinguishers". In: *Constructive Side-Channel Analysis and Secure Design*. Ed. by Emmanuel Prouff. Cham: Springer International Publishing, 2014, pp. 1–8.

[RGV14b] Oscar Reparaz, Benedikt Gierlichs, and Ingrid Verbauwhede. "Generic DPA Attacks: Curse or Blessing?" In: *Constructive Side-Channel Analysis and Secure Design*. Ed. by Emmanuel Prouff. Cham: Springer International Publishing, 2014, pp. 98–111. ISBN: 978-3-319-10175-0.

[RO05] Christian Rechberger and Elisabeth Oswald. "Practical Template Attacks". In: *Information Security Applications*. Ed. by Chae Hoon Lim and Moti Yung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 440–456. ISBN: 978-3-540-31815-6.

[San+18] Shibani Santurkar et al. *How Does Batch Normalization Help Optimization?* 2018. arXiv: 1805.11604 [stat.ML].

[Sch+14] Alexander Schaub et al. "Attacking Suggest Boxes in Web Applications Over HTTPS Using Side-Channel Stochastic Algorithms". In: vol. 8924. Aug. 2014, pp. 116–130. DOI: 10.1007/978-3-319-17127-2_8.

[Sha48] Claude E. Shannon. "A mathematical theory of communication." In: *Bell Syst. Tech. J.* 27.3 (1948), pp. 379–423.

[SM15] Tobias Schneider and Amir Moradi. "Leakage assessment methodology". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. 2015.

[Sri+14] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958.

[Ste14] Greg Ver Steeg. *Non-parametric Entropy Estimation Toolbox*. 2014. URL: https://github.com/gregversteeg/NPEET.

[Str+15] Daehyun Strobel et al. "Scandalee: a side-channel-based disassembler using local electromagnetic emanations". In: *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition*. 2015.

[The03] The OpenSSL Project. "OpenSSL: The Open Source toolkit for SSL/TLS". www.openssl.org. 2003.

[Thi+18] Hugues Thiebeauld et al. "SCATTER: A New Dimension in Side-Channel". In: *Constructive Side-Channel Analysis and Secure Design*. Ed. by Junfeng Fan and Benedikt Gierlichs. Cham: Springer International Publishing, 2018, pp. 135–152. ISBN: 978-3-319-89641-0.

[Tim19] Benjamin Timon. "Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 (2019), pp. 107–131. DOI: 10.13154/tches.v2019.i2.107-131. URL: https://tches.iacr.org/index.php/TCHES/article/view/7387.

[VC+12] Nicolas Veyrat-Charvillon et al. "Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note". In: *Advances in Cryptology – ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 740–757. ISBN: 978-3-642-34961-4.

[VC+13] Nicolas Veyrat-Charvillon et al. "An Optimal Key Enumeration Algorithm and Its Application to Side-Channel Attacks". In: *Selected Areas in Cryptography*. Ed. by Lars R. Knudsen and Huapeng Wu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 390–406. ISBN: 978-3-642-35999-6.

[VCS09] Nicolas Veyrat-Charvillon and François-Xavier Standaert. "Mutual Information Analysis: How, When and Why?" In: *Cryptographic Hardware and Embedded Systems - CHES 2009*. Ed. by Christophe Clavier and Kris Gaj. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 429–443. ISBN: 978-3-642-04138-9.

[VEB10] Nguyen Xuan Vinh, Julien Epps, and James Bailey. "Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance". In: *Journal of Machine Learning Research* 11.95 (2010), pp. 2837–2854. URL: http://jmlr.org/papers/v11/vinh10a.html.

[Vir+20] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[WO11] Carolyn Whitnall and Elisabeth Oswald. "A Comprehensive Evaluation of Mutual Information Analysis Using a Fair Evaluation Framework". In: *Advances in Cryptology – CRYPTO 2011*. Ed. by Phillip Rogaway. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 316–334.

[WOS14] Carolyn Whitnall, Elisabeth Oswald, and François-Xavier Standaert. "The Myth of Generic DPA...and the Magic of Learning". In: *Topics in Cryptology – CT-RSA 2014*. Ed. by Josh Benaloh. Cham: Springer International Publishing, 2014, pp. 183–205. ISBN: 978-3-319-04852-9.

[ZLG21] Chi Zhang, Xiangjun Lu, and Dawu Gu. "Binary Classification-Based Side-Channel Analysis". In: *2021 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*. 2021, pp. 1–6. DOI: 10.1109/AsianHOST53231.2021.9699563.

# *Abstract*

Side-Channel Analysis (SCA) is defined as the process of gaining information on a device holding a secret through its physical leakage such as power consumption or Electromagnetic (EM) emanations. Whatever the utilized strategy, the amount of information one could gain from a side-channel data, called a trace, is always bounded by the Mutual Information (MI) between the secret and the trace. This makes it, all punning aside, a key quantity for leakage evaluation. Unfortunately, traces are usually of too high dimension for classical statistical estimators to stay sound when computing the MI over full traces. However, recent works from the machine learning community have shown that it is possible to evaluate the MI in high dimensional space thanks to newest deep learning techniques. This thesis explores how this new estimator impacts the side-channel domain.

The first part is dedicated to an analysis of the Mutual Information Neural Estimation (MINE) technique in a side-channel context which aim is to derive the best way of using such tool in practice. It shows that the intrinsic multi-dimensional aspect of the technique is highly valuable for SCA since there are often multiple leakage sources in side-channel traces. The method is derived as a generic leakage assessment tool that can be used whatever the type of data, device or implementation.

Knowing how much information is contained in the traces is different from knowing how to exploit it optimally to recover a secret such as a cryptographic key, especially in an unsupervised context when no profiling of the target is allowed. Therefore, the second part of this thesis presents a new mathematical framework, designed to bridge classical Mutual Information Attacks (MIA) and the multidimensional aspect of neural-based estimators. This allows to derive, to the best of our knowledge, the first unsupervised attack able to benefit from both the power of deep learning techniques and the valuable theoretical properties of MI. In practice this attack suffers from two drawbacks : the time complexity, since it requires as many network trainings as there are key hypotheses (often 256), and a strong a priori on the leakage model of the target device.

The third part of the thesis makes use of the previously introduced mathematical framework to build a deep learning architecture able to recover by itself such a leakage model. It allows to derive a new unsupervised attack, the EVIL Machine Attack, with only one network training solving the two precedent issues at the same time.

The analysis of the EVIL machine in the context of masked implementations gave rise to questions about stochastic attacks and their generalization to higher-order versions. The last part of this thesis is dedicated to an analysis followed by a new unsupervised attack proposition, the Joint Moment Regression, which is agnostic to the underlying masking scheme as opposed to state-of-the-art techniques.