Toward Optimal Deep-Learning Based Side-Channel Attacks: Probability Concentration Inequality Loss and Its Usage

Akira Ito[1], Rei Ueno[1] and Naofumi Homma[1]

Miyagi, Japan, ito@riec.tohoku.ac.jp, rei.ueno.a8@tohoku.ac.jp, homma@riec.tohoku.ac.jp

Abstract. In this paper, we present solutions to some open problems for constructing efficient deep learning-based side-channel attacks (DL-SCAs) through a theoretical analysis. There are two major open problems in DL-SCAs: (i) the effect of the difference in secret key values used for profiling and attack phases is unclear, and (ii) the optimality of the negative log-likelihood (NLL) loss function used in the conventional learning method is unknown. These two problems have hindered the accurate performance evaluation and optimization of DL-SCAs. To address the problem (i), we clarified the strict conditions under which the use of different correct keys in profiling and attack phases affects the performance of DL-SCA. For the problem (ii), we then analyzed the relationship between the NLL loss and direct performance metrics of DL-SCAs (i.e., success rate (SR)/guessing entropy (GE)) and proved that the minimum NLL loss is sufficient but not necessary to achieve the optimal distinguisher of DL-SCA. This explains why DL-SCA succeeds even when the NLL loss is large and motivated us to design a new loss function. Based on the above analysis result, we also propose a new loss function called the probability concentration inequality (PCI) loss function. We derive the PCI loss as an upper bound of GE and a lower bound of the SR using a probability concentration inequality. Minimizing the PCI loss during training can directly optimize the GE and SR of the subsequent attack phase. In this paper, we describe the characteristics of PCI loss and NLL loss and introduce a new learning method that takes full advantage of the characteristics. We also analytically investigate the difference between the PCI loss and ranking loss reported in a previous work for a similar purpose and explain the advantage of PCI loss over the ranking loss. Finally, we validate the analysis and demonstrate the effectiveness of the proposed DL-SCA using the PCI loss through experimental attacks on public datasets.

Keywords: Side-channel analysis · Deep learning · Optimal distinguisher

1 Introduction

1.1 Background

Deep-learning-based side-channel attacks (DL-SCAs) on cryptographic modules have increasingly emerged in recent years [MHM14, CDP17, ZBHV19, HHGG20, RWPP21]. A side-channel attack (SCA) extracts secret information by analyzing side-channel information (e.g., power consumption and electromagnetic radiation) leaked from cryptographic modules during the execution of cryptographic operations. DL-SCA is a type of profiling-based SCA consisting of a profiling phase and an attack phase. In the profiling phase, an attacker obtains side-channel traces from profiling device(s), which would have the same (or similar) leakage characteristics as the target device, in order to extract a model representing the target's characteristics using DL. In the attack phase, the attacker uses the trained model to efficiently estimate the secret key from the side-channel information of the target device. Compared with conventional profiling attacks such as template attacks, DL-SCA has the advantage of not requiring any specific knowledge about the detailed implementation or assumptions of the side-channel characteristics of the target device. There is a high demand for the accurate assessment of DL-SCA threats because of the increasing number of cryptographic devices that can be profiled, owing to the deployment of Internet-of-Things applications.

There are two major open problems in DL-SCA: (i) the effect of secret key value(s) used during the profiling phase is unknown, and (ii) the optimality of the loss function used during training, such as NLL loss, is unclear. These major problems make it difficult to optimize and assess the attack performance of DL-SCA.

Regarding Problem (i), DL-SCA has been evaluated in two ways: the secret key value is fixed or variable during profiling; this is because several public datasets such as ASCAD [BPS+20], AES_HD [PHJ+19] and AES_RD [CK09] contain so-called "fixed-key database" and "variable-key database" (Surprisingly, for some fixed-key databases, the secret key values during profiling and attack phases are identical). However, the effect of the difference between the fixed (or identical) and variable keys on the attack performance has not been investigated analytically. In other words, it is unknown why and when we can or cannot fix the secret key during the profiling phase. In addition, this problem obscures the validity of the conventional loss functions presented for DL-SCA, namely the cross-entropy ratio (CER) loss [ZZN+20] and ranking loss [ZBD+20], because they are formulated using the secret key used in the profiling phase. Therefore, it is necessary to clarify the strict conditions under which the secret key value used in the profiling does not affect the attack performance in order to evaluate and optimize the performance of DL-SCA with proper use of the known datasets and loss functions.

Concerning Problem (ii), some previous studies have already emphasized the difficulty of evaluating the performance of the key estimation in the attack phase by the common metrics in DL, such as accuracy and negative log-likelihood (NLL) [PHJ+19, ZZN+20, RZC+21]. As an extreme example, in [PHJ+19], Picek et al. showed that a DL-SCA could be successful even when the accuracy was 0%. Therefore, instead of the common metrics, the guessing entropy (GE) and success rate (SR) are generally used as performance metrics in SCAs; however, the analytical reason for the incomplete evaluation of DL-SCA remains unclear. Measure et al. proved that NLL loss gives a lower bound of the mutual information between the side-channel information and a target intermediate value [MDP19]. Accordingly, the NLL loss gives an approximate lower bound of the SR upper bound because this mutual information is known to give an upper bound of the SR [dCGRP19], which is the first (but weak) relationship between NLL loss and SR/GE. Recently, Zaid et al. argued that a model trained with ranking loss provides an upper bound on the perceived information estimated by NLL loss and a lower bound of mutual information [ZBD+20]. The authors mentioned that this is because the ranking loss maximized the SR during the attack phase, and the probability distribution maximizing the SR provided mutual information. This argument is intuitive but it is not analytically or theoretically rigorous.[1] Consequently, their explanation does not validate the statement that NLL loss is not the best method for evaluating DL-SCA. Thus, it is still unclear

[1]For example, Eq. (18) in [ZBD+20] contains some errors, although this equation is the basis of Section 4.2, which insists on the superiority of ranking loss over NLL loss. Specifically, $\log_2(\max_\theta \Pr[Z = z \mid \boldsymbol{t}_i, \theta])$ in the equation always becomes zero because $\max_\theta \Pr[Z = z \mid \boldsymbol{t}_i, \theta] = 1$ necessarily holds. It is worth noting that $\Pr[Z = z \mid \boldsymbol{t}_i, \theta]$ is not a likelihood function, but the estimated conditional probability of an intermediate value given a trace $\boldsymbol{t}_i$ by a model. Thus, Eq. (18) indicates that $H(\boldsymbol{Z}) = MI(\boldsymbol{Z}; \boldsymbol{T})$, which implies that the mutual information is always equal to the entropy of the intermediate value. This, generally, does not hold.

why the NLL loss is incomplete for optimizing the performance of DL-SCA (i.e., obtaining the optimal distinguisher). This problem also leads to difficulty in designing a more appropriate loss function for DL-SCAs.

1.2 Our contributions

This paper first provides solutions to problems (i) and (ii) from the viewpoint of probability-theoretic analysis. We therefore propose a new loss function and learning method that improves the performance of DL-SCAs beyond conventional losses. Our contributions are as follows.

- As a solution to problem (i), we clarify the condition for which the performance of DL-SCAs in the attack phase is independent of the secret key value in the profiling phase. We refer to it as the *key-independence* condition, which is derived by analyzing the formal definition of GE. Under the key-independence condition, we can demonstrate the utility of an arbitrary secret key value including a fixed key value for profiling without any effect on the subsequent attack. We examine whether typical AES software and hardware implementations satisfy the key-independence condition.

- To solve problem (ii), we theoretically analyze the relationship between the optimal distinguisher and the cross-entropy (CE) loss function because the NLL loss is an approximation of the CE loss. Then, as a solution, we present a proof that the minimum CE/NLL loss is sufficient but not necessary to provide an optimal distinguisher in DL-SCA. More precisely, our proof reveals that the model parameters that (exactly) minimize the CE/NLL loss provide an optimal distinguisher; however, the probability distribution that gives the optimal distinguisher (i.e., yields the maximized SR) does not necessarily have the minimum CE/NLL loss. The core idea of the proof comprises a concrete method for deriving an infinite number of different probability distributions that provide the optimal SR and GE but have a larger CE/NLL. The proof suggests that a smaller (but non-minimum) value of the NLL loss does not necessarily make the SR and GE, respectively, larger and smaller in DL-SCA, which motivates us to propose a new loss function suitable for DL-SCA.

- We propose the PCI (probability concentration inequality) loss function for DL-SCA to overcome the NLL loss's insufficiency. The PCI loss is derived from an upper bound of GE and a lower bound of SR, which we present on the basis of a probability concentration inequality. The PCI loss minimization directly maximizes the attack performance of the DL-SCA. This study also investigates the relationship between the PCI loss and a conventional loss function, the ranking loss. We prove that the PCI loss is an upper bound of the ranking loss minimized by adjusting the parametrizing coefficient. As it is unknown how to derive the coefficient for the practical use and evaluation of ranking loss, the PCI loss has an advantage in that it has no such coefficient to be derived.

- We propose a new model learning method that takes full advantage of both PCI loss and NLL loss. The proposed learning method consists of two parts: training with NLL loss and PCI loss. The first part trains a model with NLL loss to extract information about the intermediate values of the correct key from the side-channel traces. Subsequently, the second part trains another model with PCI loss to determine its output probability from the information extracted by the model trained in the first part. In this study, the output distribution of the second model was trained to directly optimize SR and GE, rather than NLL. Through experimental evaluations via public datasets, we demonstrate that the proposed method achieves a successful key recovery with fewer traces compared with using the ranking loss and NLL loss alone.

1.3 Paper organization

The remainder of this paper is organized as follows: Section 2 describes the notation used throughout this study and an overview of DL-SCA. Section 3 presents our probability-theoretical analysis to solve two open problems in DL-SCAs. Section 4 proposes a new loss function named PCI loss for DL-SCAs. Section 5 proposes a new learning method that utilizes both PCI and NLL losses for a more efficient DL-SCA. Section 6 presents the experimental attacks used to validate the proposed method. Finally, Section 7 concludes this paper.

2 Preliminaries

2.1 Notation

A calligraphic letter (e.g., $\mathcal{X}$) represents a set, and an uppercase variable (e.g., X) represents a random variable over the corresponding set (e.g., $\mathcal{X}$); a lowercase variable (e.g., x) is an element of the corresponding set; Pr denotes a probability measure, and the probability density and mass functions are denoted by the symbols p, q, and r; q corresponds to the true distribution (i.e., the true probability measure Pr); p denotes the probability distribution represented by a neural network; and r represents any distribution. For example, the true probability mass function of discrete random variables X, Y is $q_{X,Y}(x, y) = \Pr(X = x \wedge Y = y)$. We may omit the subscripted random variables if the random variables of the probability distribution are obvious. For instance, we may simply write $q(x, y)$ instead of $q_{X,Y}(x, y)$. The expectation is denoted by $\mathbb{E}$, and if we need to specify the target random variables to take the expectation, we write the random variables as the subscript. For example, $\mathbb{E}f(X, Y) = \mathbb{E}_{X,Y} f(X, Y)$ for a real-valued function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. The conditional probability distribution is denoted by $q_{X|Y}(x \mid y) = q(x \mid y)$, and $\mathbb{E}\left[f(X, Y) \mid Y = y\right]$ denotes the expected value using the conditional probability distribution $q(x \mid y)$.

In the sequel, we describe the definitions of the symbols used in this study. We represent side-channel traces as a multidimensional real vector $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^{n_t}$, where $n_t \in \mathbb{N}$ is the number of sample points of a side-channel trace. $\boldsymbol{X}$ denotes a random variable of the side-channel trace. This study mainly targets DL-SCAs on block ciphers, such as AES; therefore, the goal of DL-SCAs in this study is to estimate the correct key value $k^* \in \mathcal{K}$ by extracting from the side-channel trace $\boldsymbol{x}$; the intermediate value is denoted as $g(k, m) \in \mathbb{N}$, where g is a selection function, $k \in \mathcal{K} = \{0, 1\}^{n_k}$ is a key and $m \in \mathcal{M} = \{0, 1\}^{n_m}$ is public information such as plaintexts and ciphertexts. Here, n_k is the bit length of the partial key, and n_m is the bit length of the plaintext and ciphertext.[2] We represent the relationship between the side-channel trace and the correct key by the Markov chain $(K^*, M) \to g(K^*, M) \to \boldsymbol{X}$, where K^* is the random variable of the correct key. We also have $\boldsymbol{X} \to g(K^*, M) \to (K^*, M)$ because the Markov chain is also valid in the opposite direction [CT06]. This relationship indicates that the side-channel trace can be used to estimate the correct key. In this study, we assume that M and K^* have uniform distributions. We introduce an intermediate random variable $Z^* = g(K^*, M)$ to simplify the notation. If we need to specify the correct key of the intermediate value, we write the correct key value k^* as $Z^{(k^*)} = g(k^*, M)$. Note that the probability of $Z^{(k^*)}$ corresponds to the conditional probability of Z^* (i.e., $q_{Z^{(k^*)}}(z) = q_{Z^* \mid K^*}(z \mid k^*)$). In

[2]We often need two-byte ciphertexts c, c' for DL-SCAs on hardware implementations such as AES. In this case, $m = (c, c') \in \mathcal{M} = \{(c, c') \mid c, c' \in \{0, 1\}^{n_m}\}$, where $n_m = 8$.

SCAs, we need to consider a key candidate k in addition to the correct key k^*, which is actually used for cryptographic operations. Let K be a random variable of k. In some previous studies, it was assumed that the intermediate value $Z^{(k)}$ of an incorrect key $k \neq k^*$ is ideally independent of the side-channel trace because the key candidate k is not used for cryptographic operations [TPR13]; however, this is practically not true because the intermediate value of the correct key $Z^{(k^*)}$ and that of the incorrect key k are not independent owing to the characteristics of the S-box and the selection function g [FDLZ15, ISUH21]. We therefore assume that the Markov chain $\boldsymbol{X} \to Z^{(k^*)} \to Z^{(k)}$ holds, which causes ghost peaks in differential power analyses (DPAs) [BCO04].

2.2 Overview of DL-SCA

The DL-SCA consists of a profiling phase and an attack phase. In the profiling phase, we trained a model estimate the secret key. Let $\mathcal{S}_p = \{ (\boldsymbol{X}_i, Z_i) \mid 1 \leq i \leq n_p \}$ be a training dataset used in the profiling phase. Here, $\boldsymbol{X}_i$ denotes the side-channel trace (i.e., power consumption or electromagnetic radiation) of the i-th observation, and Z_i denotes the corresponding intermediate value. In addition, $|\mathcal{S}_p| = n_p$ is the number of traces used in the profiling phase. We assume that $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_{n_p}$ and $Z_1, Z_2, \ldots, Z_{n_p}$ are independent and identically distributed random variables, respectively. Let θ denote the model parameters of the NN (i.e., the weights). The goal of the profiling phase is to estimate the optimal model parameter $\hat{\theta}$ using the training dataset $\mathcal{S}_p$. This optimal parameter is usually given as the solution to the minimization of the cross-entropy (CE) function, defined as

$$H(q, p) = \mathbb{E}_{Z,\boldsymbol{X}}[-\log p(Z \mid \boldsymbol{X}; \theta)] = -\int \sum_z q_{Z,\boldsymbol{X}}(z, \boldsymbol{x}) \log p(z \mid \boldsymbol{x}; \theta)\, d\boldsymbol{x}, \tag{1}$$

where Z and $\boldsymbol{X}$ are the random variables of a label z and trace $\boldsymbol{x}$, respectively, [MDP19], and p is the conditional probability distribution represented by the NN with the parameter θ. In the profiling phase, the resulting $\hat{\theta}$ may vary depending on whether we choose the key value randomly (i.e., $Z = Z^* = g(K^*, M)$) or fix the key value k^* (i.e., $Z = Z^{(k^*)} = g(k^*, M)$). In this study, we investigated the effect of key selection in the profiling phase in Section 3.1.

The CE $H(q, p)$ in Eq. (1) takes the minimum value if and only if $p = q$ [Bis06, GBC16]. Therefore, we can obtain a model that approximates the true distribution q if we determine the optimal parameter $\hat{\theta}$ that minimizes $H(q, p)$; however, we cannot calculate Eq. (1) because it contains the integral and summation of the unknown probability distribution q. We therefore generally approximate $H(q, p)$ using the training data $\mathcal{S}_p$ as follows:

$$H(q, p) \approx L_{\mathrm{NLL}}(\theta) = \frac{1}{n_p} \sum_{i=1}^{n_p} -\log p(Z_i \mid \boldsymbol{X}_i; \theta). \tag{2}$$

The approximated CE in Eq. (2) is called NLL. The NLL converges in probability into CE $H(q, p)$ as $n_p \to \infty$.

In the attack phase, we then estimate the secret key k^* of the target device using the trained model parameter $\hat{\theta}$. Let $\mathcal{S}_a = \{ (\boldsymbol{X}_j, M_j) \mid 1 \leq j \leq n_a \}$ be a dataset used in the attack phase, where $|\mathcal{S}_a| = n_a$ is the number of traces, $\boldsymbol{X}_j$ is the side-channel trace at the j-th observation, and M_j is the corresponding plaintext and/or ciphertext. In the attack phase, we calculate the following NLL of each hypothetical key candidate $k \in \mathcal{K}$ using the intermediate value $Z_j^{(k)}$ calculated from M_j as

$$L_{\mathrm{NLL}}^{(k)}(\hat{\theta}) = \frac{1}{n_a} \sum_{j=1}^{n_a} -\log p(Z_j^{(k)} \mid \boldsymbol{X}_j; \hat{\theta}).$$

Thereafter, the key candidate with the smallest NLL value was estimated as the correct key. This is approximately equivalent to computing and comparing CE

$$H_k(q,p) = \mathbb{E}[-\log p(Z^{(k)} \mid \boldsymbol{X}; \hat{\theta})],$$

for a key candidate k.

To evaluate the performance of (DL-)SCA, the success rate (SR) and guess entropy (GE) are commonly used as quantitative metrics in the attack phase. The SR and GE with n_a traces in the attack phase are represented as

$$\begin{aligned} \mathrm{SR}_{n_a} &= \Pr(\mathrm{rank}(k^*, n_a) = 1), \\ \mathrm{GE}_{n_a} &= \mathbb{E}\mathrm{rank}(k^*, n_a), \end{aligned} \tag{3}$$

respectively [SMY09]. Here,

$$\mathrm{rank}(k^*, n_a) = 1 + \sum_{k \in \mathcal{K}_{\setminus k^*}} \mathbb{1}_{\left\{L_{\mathrm{NLL}}^{(k^*)}(\hat{\theta}) \geq L_{\mathrm{NLL}}^{(k)}(\hat{\theta})\right\}} = 1 + \sum_{k \in \mathcal{K}_{\setminus k^*}} \mathbb{1}_{\left\{\frac{1}{n_a}\sum_{j=1}^{n_a} \Delta_j^{(k)}(\hat{\theta}) \geq 0\right\}}$$

denotes the rank of the correct key among all key candidates, where $\mathbb{1}_{\mathcal{A}}$ is an indicator function on a set $\mathcal{A}$. Here, $\Delta_j^{(k^*,k)}(\hat{\theta}) = -\log p(Z_j^{(k^*)} \mid \boldsymbol{X}_j; \hat{\theta}) + \log p(Z_j^{(k)} \mid \boldsymbol{X}_j; \hat{\theta})$ is the difference between the log probabilities of the correct key and another key. From the definition, SR is the probability that the rank of the correct key is one, and GE is the expected rank of the correct key.

3 Analyses and Solutions to Open Problems

As a theoretical basis of this study, this section provides solutions to the two problems of DL-SCAs. The problems considered herein are that (i) the effect of key value(s) in profiling is unknown and (ii) the optimality of the conventional loss function is unclear. As a solution to Problem (i), we formulate the condition for which we can fix the key value in the profiling phase. We can use any key value-dependent loss functions such as CER and ranking losses if the target implementation satisfies the condition; this is because the key value used in the profiling phase does not affect the DL-SCA performance. We describe whether or not typical AES implementations satisfy the condition. To solve Problem (ii), we prove that the minimum NLL loss, which is commonly used in conventional DL-SCAs, is not a necessary condition for obtaining the optimal distinguisher. This indicates that the NLL loss is not necessarily the best performance metric for DL-SCAs.

3.1 Effect of key value in profiling

3.1.1 Key Independence Condition

Our idea to solve Problem (i) is that the correct key value in the profiling phase would not affect the attack performance if there is no difference in the attack performances against the same two implementations with different secret keys. We derive it as a condition where the key-value does not affect the attack performance, and we refer to this as “key independence condition”. The key independence condition likely depends on the conditions on the implementation and selection function, but, as we mention later, it depends only on the selection function because of the Markov chain $(M, K^*) \to Z^* \to \boldsymbol{X}$. Here, we focus on GE in deriving the key independence conditions for the profiling phase.

Let k_1^* and k_2^* be arbitrary key values. Then, let GE_{n_a,k_1^*} and GE_{n_a,k_2^*} be GEs when the correct keys in the attack phase are k_1^* and k_2^*, respectively. Let $\Delta_j^{(k_1^*,k_1)} = -\log p(Z_j^{(k_1^*)} \mid \boldsymbol{X}_j; \theta) + \log p(Z_j^{(k_1)} \mid \boldsymbol{X}_j; \theta)$ and $\Delta_j^{(k_2^*,k_2)} = -\log p(Z_j^{(k_2^*)} \mid \boldsymbol{X}_j; \theta) + \log p(Z_j^{(k_2)} \mid \boldsymbol{X}_j; \theta)$

for other key candidates k_1 and k_2. In addition, let $\mathcal{K}_{\setminus k_1^*}$ and $\mathcal{K}_{\setminus k_2^*}$ denote the sets of all key values, except for k_1^* and k_2^*, respectively. It is worth noting that $\Delta_j^{(k^*,k)}$ is i.i.d because the labels $Z_j^{(k)}$, $Z_j^{(k^*)}$ and the traces $\boldsymbol{X}_j$ are i.i.d. Using GE_{n_a,k_1^*} and GE_{n_a,k_2^*}, the condition can be written as:

$$\mathrm{GE}_{n_a,k_1^*} = \mathrm{GE}_{n_a,k_2^*} \Leftrightarrow \sum_{k_1 \in \mathcal{K}_{\setminus k_1^*}} \Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a} \Delta_j^{(k_1^*,k_1)} \geq 0\right) = \sum_{k_2 \in \mathcal{K}_{\setminus k_2^*}} \Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a} \Delta_j^{(k_2^*,k_2)} \geq 0\right). \tag{4}$$

Equation (4) holds if, for any key k_1, there exists a key k_2 such that[3]

$$\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a} \Delta_j^{(k_1^*,k_1)} \geq 0\right) = \Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a} \Delta_j^{(k_2^*,k_2)} \geq 0\right). \tag{5}$$

Then, let $Y_{k_1^*,k_1}$ and $Y_{k_2^*,k_2}$ be $\sum_{j=1}^{n_a} \Delta_j^{(k_1^*,k_1)}$ and $\sum_{j=1}^{n_a} \Delta_j^{(k_2^*,k_2)}$, respectively. If $Y_{k_1^*,k_1}$ and $Y_{k_2^*,k_2}$ have the same probability distribution, then Eq. (5): This holds if the characteristic functions of these two random variables are identical. The characteristic function of $Y_{k_1^*,k_1}$ can be written as

$$\mathbb{E}\exp(itY_{k_1^*,k_1}) = \mathbb{E}\exp\left(it\sum_{j=1}^{n_a} \Delta_j^{(k_1^*,k_1)}\right) = \prod_{j=1}^{n_a} \mathbb{E}\exp\left(it\Delta^{(k_1^*,k_1)}\right),$$

where i is the imaginary unit, and t is any real number. Hereafter, we omit the subscript j of $\mathbb{E}\exp\left(it\Delta^{(k_1^*,k_1)}\right)$ because it does not depend on j. We also denote the characteristic function of $Y_{k_2^*,k_2}$ in a similar manner. Accordingly, the condition is equivalent to $\mathbb{E}\exp\left(it\Delta^{(k_1^*,k_1)}\right) = \mathbb{E}\exp\left(it\Delta^{(k_2^*,k_2)}\right)$. Using the Markov chain $\boldsymbol{X} \to Z^{(k^*)} \to Z^{(k)}$, we rewrite $\mathbb{E}\exp\left(it\Delta^{(k_1^*,k_1)}\right)$ as

$$\begin{aligned}\mathbb{E}\exp\left(it\Delta^{(k_1^*,k_1)}\right) &= \mathbb{E}_{Z^{(k_1^*)},\boldsymbol{X}}\mathbb{E}_{Z^{(k_1)}}\left[\exp\left(it\Delta^{(k_1^*,k_1)}\right)\Big| Z^{(k_1^*)}\right] \\ &= \sum_{z}\sum_{z'} q_{Z^{(k_1)}|Z^{(k_1^*)}}(z' \mid z) \int q_{Z^{(k_1^*)},\boldsymbol{X}}(z,\boldsymbol{x}) \exp\left(it\Delta^{(k_1^*,k_1)}\right) d\boldsymbol{x}.\end{aligned}$$

Because $\mathbb{E}\exp\left(it\Delta^{(k_2^*,k_2)}\right)$ can also be rewritten in the same manner, we obtain the necessary conditions as

$$\forall k_1^*, k_2^* \in \mathcal{K}, q_{Z^{(k_1^*)},\boldsymbol{X}} = q_{Z^{(k_2^*)},\boldsymbol{X}}, \tag{6}$$

$$\forall k_1^*, k_2^* \in \mathcal{K}, k_1 \in \mathcal{K}_{\setminus k_1^*}, \exists k_2 \in \mathcal{K}_{\setminus k_2^*}, q_{Z^{(k_1)}|Z^{(k_1^*)}} = q_{Z^{(k_2)}|Z^{(k_2^*)}}. \tag{7}$$

Then, we simplify the first equation (6), which can be rewritten as $q_{\boldsymbol{X}|Z^{(k_1^*)}}q_{Z^{(k_1^*)}} = q_{\boldsymbol{X}|Z^{(k_2^*)}}q_{Z^{(k_2^*)}}$. It should be noted that $q_{\boldsymbol{X}|Z^{(k_1^*)}} = q_{\boldsymbol{X}|Z^{(k_2^*)}}$ because of the Markov chain $(M, K^*) \to Z^* \to \boldsymbol{X}$, where $Z^* = g(M, K^*)$. In fact, we have

$$q_{\boldsymbol{X}|Z^{(k^*)}}(\boldsymbol{x} \mid z) = q_{\boldsymbol{X}|Z^*,K^*}(\boldsymbol{x} \mid z, k^*) = \sum_{m} \frac{q_{\boldsymbol{X}|Z^*}(\boldsymbol{x} \mid z) q_{Z^*,M,K^*}(z,m,k^*)}{q_{Z,K^*}(z,k^*)} = q_{\boldsymbol{X}|Z^*}(\boldsymbol{x} \mid z).$$

Thus, Eq. (6) becomes $q_{Z^{(k_1^*)}} = q_{Z^{(k_2^*)}}$. Combining Eqs. (6) and (7), we can define the following key-independence condition for which the attack performance is independent of the key value in profiling.

[3]This condition is not sufficient if there exist two different incorrect keys $k, k' \in \mathcal{K}_{\setminus k^*}$ and a correct key k^* such that $\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k^*,k)} \geq 0\right) = \Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k^*,k')} \geq 0\right)$. However, we ignore this here because it does not hold in the case of AES mainly targeted in the present and previous works.

Definition 1 (Key-independence condition)**.** We say that a target implementation satisfies the key-independence condition if, for any correct keys k_1^*, k_2^*, and any incorrect key k_1, there exists an incorrect key k_2 such that $\forall z, z^*, q_{Z^{(k_1)},Z^{(k_1^*)}}(z, z^*) = q_{Z^{(k_2)},Z^{(k_2^*)}}(z, z^*)$, regarding the selection function.

Remark 1. The key-independence condition implies that, for any correct keys k_1^* and k_2^*, $q_{Z^{(k_1^*)}} = q_{Z^{(k_1^*)}}$ holds. This is equivalent to the *equal images under different subkeys* (EIS) [SLP05] and *symmetric key* assumptions [FDLZ15]. In [MOS11], Mangard et al. claimed that the EIS assumption implies that SR does not change with the correct key value(s) in DPA; however, the EIS assumption is weaker than the key-independence condition, hence it is not sufficient to assume it for the purpose at hand.

3.1.2 Software implementation

This subsection shows that the AES software implementations satisfy the key-independence condition. We usually use the S-box output in the first round $Z^{(k)} = \mathrm{Sbox}(k \oplus M)$ as the selection function (i.e., the intermediate value). Given the true probability distribution q represented by the probability measure Pr, we rewrite the key-independence condition as

$$\begin{aligned}\forall z, z^*, &\Pr(\mathrm{Sbox}(k_1 \oplus M) = z, \mathrm{Sbox}(k_1^* \oplus M) = z^*) \\ &= \Pr(\mathrm{Sbox}(k_2 \oplus M) = z, \mathrm{Sbox}(k_2^* \oplus M) = z^*), \quad (8)\end{aligned}$$

where k_1^*, k_2^*, and k_1 are arbitrary key values. It is necessary for us to check whether there exists an incorrect key k_2 such that Eq. (8) holds. First, we focused on the right-hand side. Let M' be a random variable of plaintext following a uniform distribution such that $M = M' \oplus k_1^* \oplus k_2^*$. Substituting $M' \oplus k_1^* \oplus k_2^*$ into M and letting $k_2 = k_1 \oplus k_1^* \oplus k_2^*$, we rewrite the right-hand side of Eq. (8) as

$$\begin{aligned}&\Pr(\mathrm{Sbox}(k_2 \oplus M) = z, \mathrm{Sbox}(k_2^* \oplus M) = z^*) \\ &\qquad = \Pr(\mathrm{Sbox}(k_1 \oplus M') = z, \mathrm{Sbox}(k_1^* \oplus M') = z^*).\end{aligned}$$

This is equivalent to the left-hand side of Eq. (8); therefore, the AES software implementation satisfies the key-independence condition, which means that we can use any key value (including a fixed value) in the profiling phase. The proof also shows that a dataset using the key k_2^* during profiling can be converted to a dataset using the other key k_1^* by replacing the plaintext M and the incorrect key k_2 with $M = M' \oplus k_1^* \oplus k_2^*$ and $k_2 = k_1 \oplus k_1^* \oplus k_2^*$, respectively. Similarly, we can convert a dataset with random key values into a dataset using a fixed key value. This indicates that the key value used is not important in the profiling phase in the case of software implementation.

3.1.3 Hardware implementation

This subsection discusses whether round-based AES hardware implementations satisfy the key-independence condition. In this case, the selection function is usually determined from the register writing of the S-box output as $Z^{(k)} = \mathrm{Sbox}^{-1}(k \oplus C) \oplus C'$ (i.e., the Hamming distance between the ciphertext byte and the corresponding 10-th round input regarding ShiftRows), where C and C' are the random variables of ciphertexts. Depending on ShiftRows, $C = C'$ for the 1st, 5th, 9th, and 13th bytes, and $C \neq C'$ otherwise. The distribution of $Z^{(k)}$ changes depending on whether $C = C'$ or $C \neq C'$, as described below.

$\boldsymbol{C = C'}$**:** Fix $k \in \mathcal{K}$. $Z^{(k)} = \mathrm{Sbox}^{-1}(k \oplus C) \oplus C$ is not a bijection function of C, and the output of the selection function does not appear depending on k. Thus, $\forall z, q_{Z^{(k_1^*)}}(z) = q_{Z^{(k_2^*)}}(z)$ does not hold, which indicates that this case does not satisfy the key-independence condition.

$C \neq C'$: The key-independence condition holds if, for any sub-keys k_1^*, k_2^*, and k_1, there exists k_2 such that

$$\begin{aligned}\forall z, z^*, &\Pr(\mathrm{Sbox}^{-1}(k_1 \oplus C) \oplus C' = z, \mathrm{Sbox}^{-1}(k_1^* \oplus C) \oplus C' = z^*) \\ &= \Pr(\mathrm{Sbox}^{-1}(k_2 \oplus C) \oplus C' = z, \mathrm{Sbox}^{-1}(k_2^* \oplus C) \oplus C' = z^*). \quad (9)\end{aligned}$$

When we choose $k_2 = k_2^* \oplus k_1^* \oplus k_1$ and replace C with $C'' \oplus k_2^* \oplus k_1^*$, where C'' is a uniformly distributed random variable, we obtain

$$\begin{aligned}&\Pr(\mathrm{Sbox}^{-1}(k_2 \oplus C) \oplus C' = z, \mathrm{Sbox}^{-1}(k_2^* \oplus C) \oplus C' = z^*) \\ &= \Pr(\mathrm{Sbox}^{-1}(k_1 \oplus C'') \oplus C' = z, \mathrm{Sbox}^{-1}(k_1^* \oplus C'') \oplus C' = z^*).\end{aligned}$$

This is equivalent to the left-hand side of Eq. (9); therefore, this case satisfies the key-independence condition. From the above, it is not justified to fix the key value in the profiling phase when $C = C'$; however, when $C \neq C'$, the key-independence condition is satisfied and the key value is fixed. As in the software implementation, the dataset can be converted to a fixed key even if the key value is changed randomly in the profiling phase.

3.2 CE loss minimization and optimal distinguisher

This section provides an analytical solution as an explanation why the conventional NLL loss is an inadequate performance metric for evaluating DL-SCA in terms of SR and GE. Here, we analyze the CE loss function instead of the NLL loss because the NLL loss is an approximation of the CE loss, as mentioned in SubSection 2.2. First, we show that a probability distribution with minimum CE yields the optimal distinguisher. We then show that there is an infinite number of probability distributions that do not have the minimum CE but can provide the optimal distinguisher (i.e., yield the maximized SR). In other words, the CE can be large even for a probability distribution that provides the optimal distinguisher. For simplicity, assuming that the key-independence condition holds and the selection function $g(k, m)$ is given as a bijection on k when m is fixed, we write $q_{Z^*|\boldsymbol{X}}$ instead of $q_{Z^{(k^*)}|X}$ because the relationship between intermediate values and side-channel traces does not depend on the correct key value under the key-independence condition.

Probability distribution with the minimum CE gives the optimal distinguisher. We first introduce the definition of the optimal distinguisher as follows:

Definition 2 (Optimal distinguisher)**.** Let d denote the distinguisher, which is given as a set function from a dataset to a key value. We say that d is the optimal distinguisher if, for any dataset $\mathcal{S}_a = \{(\boldsymbol{X}_j, M_j) \mid 1 \leq j \leq n_a\}$, an attack using $d(\mathcal{S}_a)$ maximizes the SR.

We then provide Proposition 1, which indicates the *sufficient* condition for the optimality of the distinguisher.

Proposition 1. *Let $\mathcal{S}_a$ and $d(\mathcal{S}_a)$ be a dataset and a distinguisher for the attack phase, respectively. The true distribution $q_{Z^*|\boldsymbol{X}}$ gives the optimal distinguisher, as follows:*

$$d(\mathcal{S}_a) = \arg\max_k \sum_{j=1}^{n_a} \log q_{Z^*|\boldsymbol{X}}(Z_j^{(k)} \mid \boldsymbol{X}_j).$$

Proof. See Appendix A.1 for the proof [4]. □

[4]Some previous studies [MDP19, ZBD+20] are based on Proposition 1, although they used the proposition without proof.

Proposition 1 proves that the derivation of the true distribution, which necessarily has the minimum CE loss, yields the optimal distinguisher. In other words, the CE loss is the best loss function if we can obtain the true distribution $q_{Z^*|\boldsymbol{X}}$ through CE loss minimization. Note that, if the key-independence condition does not hold, we should train models to approximate the conditional probability distribution $q_{Z(k^*)|\boldsymbol{X}}$ of each key candidate k^* to get the optimal distinguisher (e.g., we should train 256 models for all possible key candidates), because the conditional probability $q_{Z(k^*)|\boldsymbol{X}}$ depends on the secret key value.

Optimal distinguisher does not necessarily minimize CE loss. In reality, we cannot usually obtain the true distribution during training because of errors known as *bias*, *variance*, and noise [HTF09]. The question that arises here is whether a smaller (but not a minimum) CE value indicates a better performance of DL-SCAs. The CE loss can be a good indicator if any probability distribution giving the optimal distinguisher *necessarily* has the minimum CE loss (i.e., the converse of Proposition 1 is true). However, we prove that this is not true for an explicit counterexample.

We introduce Lemma 1 to prove that the converse of Proposition 1 is false.

Lemma 1. *Let*

$$r'_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x}) = \frac{r_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x})^\beta}{\sum_z r_{Z^*|\boldsymbol{X}}(z \mid \boldsymbol{x})^\beta},$$

where $r_{Z^*|\boldsymbol{X}}$ is any probability distribution and β is a positive real number. SR_{n_a} *and* GE_{n_a} *are identical for any β in the SCA using r' as a distinguisher.*

Proof. See Appendix A.2 □

Lemma 1 guarantees that there are conversions from r to r' with the preserved SCA result. In particular, an infinite number of such conversions exist because β is any positive real number. The CE, however, is not invariant to β. Thus, we have Proposition 2.

Proposition 2. *Let $\mathcal{S}_a$ and $d(\mathcal{S}_a)$ be a dataset and an optimal distinguisher for the attack phase, respectively. The optimal distinguisher $d(\mathcal{S}_a)$ does not necessarily have a probability distribution with a minimum CE.*

Proof. Let $q_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x})$ be the true distribution that provides the optimal distinguisher. From Lemma 1, $q'_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x}) = q_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x})^\beta / \sum_z q_{Z^*|\boldsymbol{X}}(z \mid \boldsymbol{x})^\beta$ also provides the optimal distinguisher for any β. However, $H(q_{Z^*|\boldsymbol{X}}, q'_{Z^*|\boldsymbol{X}})$ generally does not reach the minimum value when $\beta \neq 1$. This completes the proof. □

Combining Propositions 1 and 2. We summarize the above argument on the optimal distinguisher in Theorem 1 below.

Theorem 1. *Let $q_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x})$ be the true distribution, and let $q'_{Z^*|\boldsymbol{X}}(z^{(k)} \mid \boldsymbol{x})$ be any probability distribution. It is sufficient but not necessary to provide the optimal distinguisher that $H(q, q')$ is the minimum value.*

Proof. It is obvious from Propositions 1 and 2. □

From Theorem 1, we can see that the (non-minimum) value of the CE (NLL) loss does not necessarily correspond to the attack performance of DL-SCA. The theorem also explains why a large CE loss does not necessarily mean that the DL-SCA would fail. In fact, by adapting the parameter β, we can create a probability distribution q' that is far from the true distribution q with regard to the CE loss. Consequently, Theorem 1 suggests that CE loss is not necessarily the best metric for evaluating the attack performance of

DL-SCA. This problem occurs because the CE loss function is not designed from the DL-SCA metrics, namely, SR and GE. This motivated us to propose a new loss function based on the SR and GE as described in the next section.

4 Probability Concentration Inequality Loss Function

This section derives the probability concentration inequality (PCI) loss function for DL-SCA based on the above analyses and solutions and explains its intuitive meaning. In addition, we describe the relationship between the PCI loss and the ranking loss introduced in [ZBD+20]. We assume that the key-independence condition is satisfied throughout this section because we should fix the correct key for deriving the PCI loss.

4.1 Derivation of PCI loss

The PCI loss is a differentiable function derived as a lower bound of SR. We first (i) introduce an upper bound of GE, then (ii) derive a lower bound of SR from it using Lemma 2, and finally (iii) provide the definition of the PCI loss by using the derived lower bound of SR.

(i) Upper bound of GE. Let k^* be a correct key. Let

$$\Delta^{(k)}(\hat{\theta}) = -\log p(Z^{(k^*)} \mid \boldsymbol{X}; \hat{\theta}) + \log p(Z^{(k)} \mid \boldsymbol{X}; \hat{\theta}) \tag{10}$$

be the difference between the negative log-probabilities of the correct and incorrect keys k. For simplicity, we omit $\hat{\theta}$ as $\Delta^{(k)}$. Note that the difference $\Delta^{(k)}$ is a random variable. Suppose that $\mathbb{E}|\Delta^{(k)}| < \infty$ and $\mathbb{E}[\Delta^{(k)} - \mathbb{E}\Delta^{(k)}]^2 < \infty$. From this assumption, we can define the mean $\mathbb{E}\Delta^{(k)}$ and variance $\mathbb{E}[\Delta^{(k)} - \mathbb{E}\Delta^{(k)}]^2$. Similarly, suppose that there exists a constant positive real number $R_k \in (0, \infty)$ such that $|\Delta^{(k)} - \mathbb{E}\Delta^{(k)}| < R_k$ with probability 1. If $\mathbb{E}\Delta^{(k)} < 0$ for all incorrect keys, the law of large numbers [Ver18] yields $\mathrm{GE}_{n_a} \xrightarrow{P} 1$ as $n_a \to \infty$, where $\xrightarrow{P}$ is the convergence in probability. This indicates that, in this case (i.e., $\mathbb{E}\Delta^{(k)} < 0$), DL-SCA should be successful if we can use an infinite number of traces; however, the number of traces available in the attack phase is finite in practice, and the GE gradually approaches a value of one as the number of traces increases. This approach speed (i.e., upper bound of GE for a given number of traces) has been analyzed using the central limit theorem in previous studies [Riv09, DZFL14, LPR+14, ZDF20]; however, these methods provide only asymptotic results. In contrast, this study uses Bennett's inequality[5] to obtain a meaningfully tighter bound, which is a concentration inequality that holds for any number of traces. According to Bennett's inequality, we introduce Theorem 2.

Theorem 2 (Upper bound of GE)**.** *Let $\mathcal{K}_{\setminus k^*}$ be the set of all the incorrect keys. Let $\Delta_1^{(k)}, \Delta_2^{(k)}, \ldots, \Delta_j^{(k)}, \ldots, \Delta_{n_a}^{(k)}$ be independent copies of $\Delta^{(k)}$ defined by Eq. (10). We assume that $|\Delta^{(k)} - \mathbb{E}\Delta^{(k)}| < R_k$ almost surely for every k, where $R_k \in (0, \infty)$. Let μ_k and σ_k^2 be the mean and variance of $\Delta^{(k)}$, respectively. The GE with n_a traces is upper-bounded as*

$$\mathrm{GE}_{n_a} \leq 1 + |\mathcal{K}'| + \sum_{k \in \mathcal{K}'} \exp\left(-n_a c_k\right),$$

where $\mathcal{K}' = \{ k \mid \mu_k \geq 0, k \in \mathcal{K}_{\setminus k^*} \}$, $c_k = \frac{\sigma_k^2}{R_k^2} h\left(\frac{-R_k \mu_k}{\sigma_k^2}\right)$, and $h(u) = (1+u)\log(1+u) - u$.

[5]See Appendix C.

Proof. First, for $\mu_k \geq 0$, the law of large numbers yields that $\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)} \xrightarrow{P} \mu_k \geq 0$ as $n_a \to \infty$. Thus, we have

$$\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)} \geq 0\right) \leq 1. \tag{11}$$

Then, for $\mu_k < 0$, applying Bennett's inequality to $\Delta_j^{(k)}$, we obtain

$$\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}(\Delta_j^{(k)} - \mu_k) \geq t\right) \leq \exp\left(-n_a\frac{\sigma_k^2}{R_k^2}h\left(\frac{R_k t}{\sigma_k^2}\right)\right),$$

for any real positive number t. If we substitute $t = -\mu_k$ and replace $c_k = \frac{\sigma_k^2}{R_k^2}h\left(\frac{-R_k\mu_k}{\sigma_k^2}\right)$, we have

$$\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)} \geq 0\right) \leq \exp\left(-n_a c_k\right). \tag{12}$$

From these inequalities, (11) and (12), and the definition of GE in Eq. (3), we can deduce the upper bound of GE_{n_a} as

$$\mathrm{GE}_{n_a} = 1 + \sum_{k\in\mathcal{K}_{\setminus k^*}}\Pr\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)} \geq 0\right) \leq 1 + |\mathcal{K}'| + \sum_{k\in\mathcal{K}_{\setminus k^*}}\exp\left(-n_a c_k\right).$$

□

To use this theorem in practice, we need to calculate R_k, σ_k^2, and μ_k; the calculation method is described in Section 5.2.

(ii) Lower bound of SR. In contrast to the upper bound of the GE, the lower bound of the SR cannot be obtained by a direct evaluation via probability inequalities. This is because SR is defined by

$$\mathrm{SR}_{n_a} = \Pr(\mathrm{rank}(k^*, n_a) = 1) = \Pr\left(\bigwedge_{k\in\mathcal{K}_{\setminus k^*}}\left(\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)} < 0\right)\right),$$

cannot be simplified because $\Delta_j^{(k)}$ for different keys is not independent [TPR13,LPR$^+$14]. Therefore, instead of directly evaluating the SR, we focus on the inequality relation between SR and GE, introduced as Lemma 2.

Lemma 2. *For* SR_{n_a} *and* GE_{n_a} *with* n_a *traces, we have*

$$2 - \mathrm{GE}_{n_a} \leq \mathrm{SR}_{n_a} \leq \frac{|\mathcal{K}| - \mathrm{GE}_{n_a}}{|\mathcal{K}| - 1}.$$

Proof. From Markov's inequality, [6] and the definitions of SR and GE in Eq. (3), we derive the upper bound as $\mathrm{SR}_{n_a} = \Pr\left(|\mathcal{K}| - \mathrm{rank}(k^*, n_a) \geq |\mathcal{K}| - 1\right) \leq (|\mathcal{K}| - \mathrm{GE}_{n_a})/(|\mathcal{K}| - 1)$. For the lower bound, we rewrite $\mathrm{SR}_{n_a} = \Pr(1 \geq \mathrm{rank}(k^*, n_a)) = 1 - \Pr(\mathrm{rank}(k^*, n_a) \geq 2)$. Markov's inequality yields $\Pr(\mathrm{rank}(k^*, n_a) - 1 \geq 1) \leq \mathbb{E}\mathrm{rank}(k^*, n_a) - 1 = \mathrm{GE}_{n_a} - 1$. Thus, $2 - \mathrm{GE}_{n_a} \leq \mathrm{SR}_{n_a}$ holds. □

[6] Markov's inequality is one of the most famous inequalities; that is, for any positive random variable Y and for any positive number t, we have $\Pr(Y \geq t) \leq \mathbb{E}Y/t$.

Consequently, we have Theorem 3 from Lemma 2 and Theorem 2.

Theorem 3 (Lower bound of SR)**.** *We define $\mathcal{K}'$ and c_k as in Theorem 2. The SR with n_a traces is lower bounded as*

$$1 - |\mathcal{K}'| - \sum_{k \in \mathcal{K}'} \exp\left(-n_a c_k\right) \leq \mathrm{SR}_{n_a}.$$

Proof. It is obvious from Theorem 2 and Lemma 2. □

(iii) Definition of PCI loss. We define the PCI loss as a loss function closely (and analytically) related to the attack performance (i.e., SR and GE).

Definition 3 (PCI loss)**.** We define PCI loss by

$$L_{\mathrm{PCI}}(\hat{\theta}) = \frac{1}{n_a} \log\left(|\mathcal{K}'| + \sum_{k \in \mathcal{K}'} \exp\left(-n_a c_k\right)\right).$$

From Theorems 3 and 2, we have that $\mathrm{GE}_{n_a} \leq 1 + \exp(n_a L_{\mathrm{PCI}})$ and $\mathrm{SR}_{n_a} \geq 1 - \exp(n_a L_{\mathrm{PCI}})$.

4.2 Intuitive meaning of PCI loss

This subsection provides the intuitive meaning of the PCI loss and its minimization. For simplicity, suppose that $\mu_k < 0$ holds for any $k \in K_{\setminus k^*}$; that is,

$$L_{\mathrm{PCI}}(\hat{\theta}) = \frac{1}{n_a} \log\left(\sum_{k \in \mathcal{K}_{\setminus k^*}} \exp\left(-n_a \frac{\sigma_k^2}{R_k^2} h\left(\frac{-R_k \mu_k}{\sigma_k^2}\right)\right)\right). \tag{13}$$

For the interpretation, we adopt the inequality $h(u) \geq u^2/(2 + 2u/3)$ to rewrite Eq. (13) into the following form corresponding to Bernstein's inequality [Ver18]:

$$L_{\mathrm{PCI}}(\hat{\theta}) \leq \frac{1}{n_a} \log\left(\sum_{k \in \mathcal{K}_{\setminus k^*}} \exp\left(\frac{-n_a \mu_k^2/2}{\sigma_k^2 - R_k \mu_k/3}\right)\right). \tag{14}$$

Because the PCI loss is designed based on the SR and GE bounds, a smaller PCI loss can make the SR and GE larger and smaller, respectively. This leads to a more efficient DL-SCA when the argument of the exponential in Eq. (14) is smaller. More concretely, the attack is easier if (i) the number of traces n_a is larger, (ii) the absolute mean $|\mu_k|$ is larger, and (iii) the variance σ_k^2 is smaller. In addition, minimizing the PCI loss trains a model such that the NLL difference between the correct and other keys with a small absolute mean $|\mu_k|$ and large variance σ_k^2 increases when n_a is large. To demonstrate the intuitive meaning of PCI loss, we introduce Proposition 3.

Proposition 3. *Let μ_k and σ_k^2 be the mean and variance of $\Delta^{(k)}$, respectively. In addition, let $\mathcal{K}$ be the set of keys. Suppose that $\mu_k < 0$ holds for any incorrect key k. The PCI loss $L_{\mathrm{PCI}}(\hat{\theta})$ parameterized by $\hat{\theta}$ is bounded as follows:*

$$-\min_{k \neq k^*} c_k < L_{\mathrm{PCI}}(\hat{\theta}) \leq -\min_{k \neq k^*} c_k + \frac{1}{n_a} \log(|\mathcal{K}| - 1). \tag{15}$$

Proof. Let $m = \max_{k \neq k^*} -n_a c_k = -\min_{k \neq k^*} n_a c_k$. We have

$$m = \log \exp(m) < \log \sum_{k \neq k^*} \exp(-n_a c_k) \leq \log \sum_{k \neq k^*} \exp(m) = m + \log(|\mathcal{K}| - 1).$$

Substituting $-\min_{k \neq k^*} n_a c_k$ for m, we obtain Eq. (15). □

From Proposition 3, we confirm that $\min_{k \neq k^*} c_k$ characterizes the PCI loss because $L_{\mathrm{PCI}} \to -\min_{k \neq k^*} c_k$ as $n_a \to \infty$. This indicates that the convergence rates of SR and GE are determined by $\min_{k \neq k^*} c_k$ when the number of traces is sufficiently large. A similar result was reported in a previous study on SR and GE approximation using the central limit theorem [GHR15].

4.3 Relation to ranking loss

To clarify the relation between PCI and ranking losses, we formally represent the ranking loss as

$$L_{\mathrm{RkL}}(\theta) = \sum_{k \in \mathcal{K}_{\setminus k^*}} \log_2 \left(1 + \exp \left(\alpha_k \sum_{j=1}^{n_a} \Delta_j^{(k)} \right) \right), \tag{16}$$

where $\alpha_k \in (0, \infty)$ is any non-negative number. Note that the original definition of the ranking loss in [ZBD+20] is equivalent to Eq. (16) when we set all α_k to the same value α. We first prove the relation between L_{RkL} and GE/SR that $\mathrm{GE}_{n_a} \leq 1 + \mathbb{E}L_{\mathrm{RkL}}$ and $\mathrm{SR}_{n_a} \geq 1 - \mathbb{E}L_{\mathrm{RkL}}$. This proof is presented in Appendix B. In this sense, Eq. (16) is a stronger inequality than that shown in [ZBD+20]. According to Eq. (16), we can determine α_k and then derive the PCI loss from it. Accordingly, Theorem 4 holds.

Theorem 4. *We define μ_k and $\mathcal{K}_{\setminus k^*}$ as in Theorem 2. Suppose $\mu_k < 0$ for any incorrect key $k \in \mathcal{K}_{\setminus k^*}$. We have*

$$\inf_{\boldsymbol{\alpha}} \mathbb{E}L_{\mathrm{RkL}}(\theta) \leq \log_2(e) \exp(n_a L_{\mathrm{PCI}}(\theta)),$$

where $\boldsymbol{\alpha} = (\alpha_k)_{k \in \mathcal{K}_{\setminus k^*}} \in (0, \infty)^{|\mathcal{K}_{\setminus k^*}|}$, and e is Napier's number.

Proof. See Appendix A.3. □

This theorem implies that the PCI loss can bound the minimum value of the ranking loss. The theorem also suggests that the PCI loss can also be derived by an analytical determination of the minimum parametrizing coefficient α of the ranking loss.

5 Learning method with PCI loss

5.1 Overview of learning method

This section proposes a new learning method that uses PCI loss. As explained in Section 4.2, the PCI loss increases the difference in likelihood between the correct key k^* and other wrong keys. Side-channel traces, however, do not contain any information about the wrong keys. The minimization of PCI loss would be useful because the intermediate values of the correct and incorrect keys are not entirely independent (e.g., there is an incorrect key that provides a similar tendency with the correct key about the Hamming weight of outputs), and this dependence gives the pseudo-relationship between side-channel traces and intermediate values corresponding to wrong keys (i.e., the assumption of the Markov chain $\boldsymbol{X} \to Z^{(k^*)} \to Z^{(k)}$).

Accordingly, our method trains two models with NLL and PCI losses. With the NLL loss, the first model extracts the intermediate value of the correct key from the side-channel trace. This corresponds to the Markov chain $\boldsymbol{X} \to Z^{(k^*)}$. Then, according to the PCI loss, the second model estimates the *adjusted* probability of the intermediate value using the output of the first model. We can handle the pseudo-relationship mentioned above using this model because the second model is trained to reduce the negative effect of wrong keys using the PCI loss. The proposed learning procedures are summarized as follows:

1. We first train a model $p_{\mathrm{approx}}(Z^{(k^*)} \mid \boldsymbol{X}; \theta_{\mathrm{approx}})$ to approximate the true distribution $q_{Z^{(k^*)}|\boldsymbol{X}}$ with NLL loss to extract the intermediate value $Z^{(k^*)}$ from side-channel traces $\boldsymbol{X}$. Learning can be performed in the same way as in conventional DL-SCAs.

2. Let $\boldsymbol{Y} = (\log p_{\mathrm{approx}}(z \mid \boldsymbol{X}; \hat{\theta}_{\mathrm{approx}}))_{z \in \mathcal{Z}}$ be the vector of log-output of the first model (i.e., $\boldsymbol{Y}$ is usually a 256-dimensional vector in the case of AES.). Here, $\hat{\theta}_{\mathrm{approx}}$ is the first model parameter obtained in the previous step. Then, we train the subsequent model $p_{\mathrm{PCI}}(Z^{(k^*)} \mid \boldsymbol{Y}; \theta_{\mathrm{PCI}})$ with PCI loss to estimate the probability of the intermediate values while considering the dependence between the side-channel information and the intermediate values of the incorrect keys.

3. In the attack phase, we use the concatenated model of p_{approx} and p_{PCI} to estimate the correct key. The key estimation with the concatenated model can be performed in the same manner as in the conventional attack phase.

5.2 Calculation method of PCI loss

This subsection explains the calculation of the PCI loss in the profiling phase for the proposed learning method. We omit the explanations of Steps 1 and 3 because they are basically the same as in the conventional DL-SCA. Suppose that the key-independence condition allows us to use PCI loss, which further allows us to fix the key value. Let k^* be the correct key, and let k be another wrong key. We need the parameters μ_k, σ_k^2, and R_k to compute the PCI loss. These parameters should be obtained using an infinite number of traces; however, this is impossible in practice. Therefore, we estimate them from a finite number of traces as follows:

$$\begin{aligned}
\hat{\mu}_k &= \frac{1}{n_p} \sum_{i=1}^{n_p} (-\log p_{\mathrm{PCI}}(z_i^{(k)} \mid \boldsymbol{y}_i; \theta) + \log p_{\mathrm{PCI}}(z_i^{(k^*)} \mid \boldsymbol{y}_i; \theta)), \\
\hat{\sigma}_k^2 &= \frac{1}{n_p} \sum_{i=1}^{n_p} (-\log p_{\mathrm{PCI}}(z_i^{(k)} \mid \boldsymbol{y}_i; \theta) + \log p_{\mathrm{PCI}}(z_i^{(k^*)} \mid \boldsymbol{y}_i; \theta))^2 - \hat{\mu}_k^2, \\
\hat{R}_k &= \max_{1 \le i \le n_p} \left| (-\log p_{\mathrm{PCI}}(z_i^{(k)} \mid \boldsymbol{y}_i; \theta) + \log p_{\mathrm{PCI}}(z_i^{(k^*)} \mid \boldsymbol{y}_i; \theta)) - \hat{\mu}_k \right|,
\end{aligned} \tag{17}$$

where n_p is the number of traces in the profiling phase, and $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_{n_p}$ are the output vectors of the first model (i.e., $\boldsymbol{y}_j = (\log p_{\mathrm{approx}}(z \mid \boldsymbol{x}_i; \hat{\theta}_{\mathrm{approx}}))_{z \in \mathcal{Z}}$, where $\boldsymbol{x}_i$ is the i-th side-channel trace) and $z_1^{(k^*)}, z_2^{(k^*)}, \dots, z_{n_p}^{(k^*)}, z_1^{(k)}, z_2^{(k)}, \dots, z_{n_p}^{(k)}$ are the intermediate values obtained in the profiling phase. We compute Eq. (17) for all wrong keys $k \in \mathcal{K}_{\setminus k^*}$, we obtain the set $\hat{\mathcal{K}}' = \{k \mid \hat{\mu}_k \ge 0, k \in \mathcal{K}_{\setminus k^*}\}$, and calculate the empirical PCI loss as

$$\hat{L}_{\mathrm{PCI}}(\theta) = \frac{1}{n_p} \log \left(|\hat{\mathcal{K}}'| + \sum_{k \in \mathcal{K}_{\setminus k^*}} \exp \left(-n_p \frac{\hat{\sigma}_k^2}{\hat{R}_k^2} h \left(\frac{-\hat{R}\hat{\mu}_k}{\hat{\sigma}_k^2} \right) \right) \right).$$

The number of parameters to be calculated is $3(|\mathcal{K}| - 1)$.

6 Experimental attacks

6.1 Experimental setup

In the experiment, we employ the two datasets ASCAD and AES_HD according to the conventional study on the ranking loss [ZBD+20]. We note that both datasets satisfy the key-independence condition. The ASCAD dataset was the first public dataset introduced as a benchmark for DL-SCA [BPS+20]. It contains the power traces from the masked AES software on an 8-bit AVR microcontroller (ATmega8515). The leakage model is the Sbox output of the third byte of the first-round SubBytes. The dataset also contains trace data with different random delay countermeasures (0, 50, and 100). In this experiment, as in [ZBD+20], we used 45,000 traces for training, 5,000 traces for validation, and 10,000 traces for testing.

The AES_HD dataset contains electromagnetic (EM) radiations measured from an unprotected AES hardware on an FPGA [CK09]. The leakage model used in this dataset is information from the last round of register writing. In this experiment, we used 45,000 traces for training, 5,000 traces for validation, and 25,000 traces for testing, as in [ZBD+20].

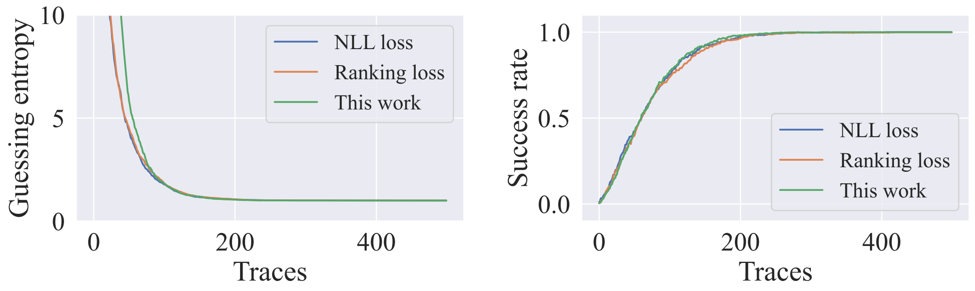
The proposed method employs two consecutive models as mentioned in Section 5.1: the former model trained with NLL loss (i.e., p_{approx}) and the latter model trained with PCI loss (i.e., p_{PCI}). In this experiment, we use the model with NLL loss in [ZBD+20] as p_{approx} , which is released in the GitHub repository[7]. The model for p_{PCI} is an MLP with four layers, where the activation function is the Softmax function for the last layer and SELU [HNL+19] for the other layers. The number of nodes in each layer is 256. The optimizer used is the same as that used by Adam [KB15], and the learning rate was set to 0.001. The batch size was set to 500 for ASCAD and 1,000 for AES_HD. The number of epochs was set to 10, and the model with the smallest PCI loss to the validation data was used for the attack. The experiment was run on a workstation with an Intel Xeon Gold 6130 CPU, 128 GB memory, and GeForce RTX 3090. The libraries used in this experiment were Tensorflow 2.4.1 [AAB+15] and Keras 2.4.0 [C+15]. To calculate the SR and GE, we need to estimate the average rank of the correct key for each trace used in the attack. In this experiment, we performed 1,000 attacks while shuffling the test data and obtained the average rank value empirically. To validate our analysis and the proposed method, we compared the proposed model trained using the proposed learning method (i.e., $p_{\text{PCI}}(Z \mid \boldsymbol{Y}; \theta_{\text{PCI}})$) along with the models trained using the NLL loss and the ranking loss found in the GitHub repository.

6.2 Experimental results

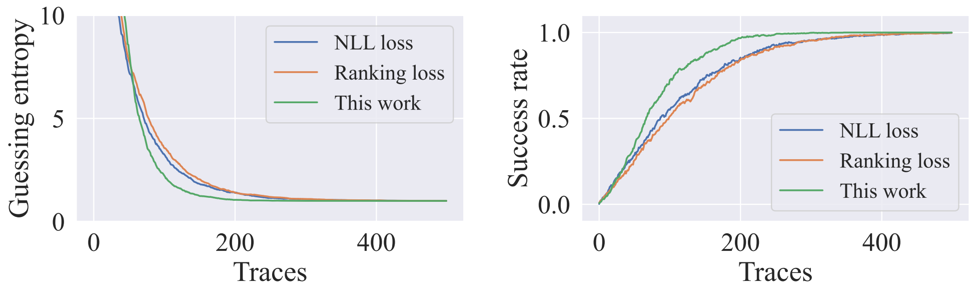
Figure 1 shows the experimental attack results, where the horizontal axis denotes the number of traces for the attack phase, and the vertical axis denotes the SR or GE. As can be seen from the figure, the SRs and GEs for NLL loss and ranking loss are almost the same. The previous study [ZBD+20] showed that there was negligible difference in the number of traces required to reach GE = 1 between the models trained with the ranking loss and NLL loss, when 45,000 traces were used in the profiling phase for any dataset. Thus, these results are consistent with [ZBD+20].

Figure 1 also shows that the proposed method is advantageous to the conventional methods in terms of SR and GE for ASCAD with desynchronization (i.e., ASCAD 50 and ASCAD 100). This is because p_{approx} is not trained adequately to acquire the true distribution q owing to the SCA countermeasure, which makes the training difficult. Therefore, p_{approx} would be away from the optimal distinguisher, and thus the proposed model p_{PCI}, which may be closer to the optimal distinguisher owing to the training via the PCI loss, is more efficient for the attack on the ASCAD dataset with desynchronization compared with the other two models. In contrast, there is negligible difference in using ASCAD without desynchronization and AES_HD. From Theorem 1, the proposed method does not improve the performance of DL-SCAs if p_{approx} is well trained to sufficiently approximate the true probability distribution q. This implies that the optimal distinguisher may be derived by training with NLL loss for the cases of ASCAD without desynchronization and the AES_HD dataset. Note that the SR of our method for ASCAD with desynchro-

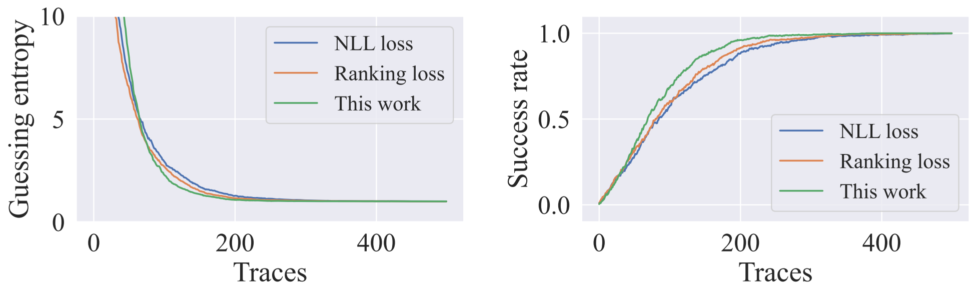
[7]https://github.com/gabzai/Ranking-Loss-SCA



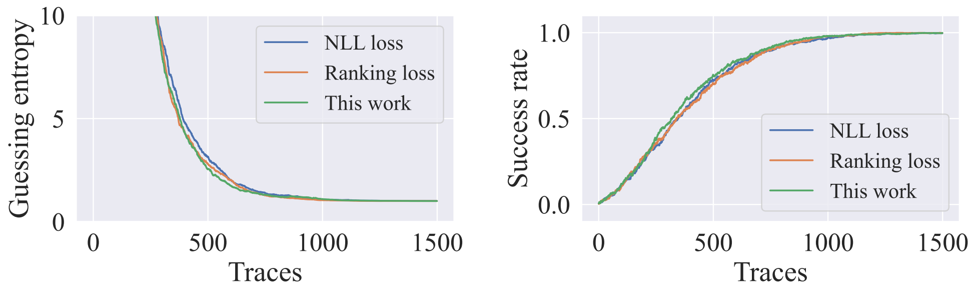
(a) Attack results of ASCAD dataset without desynchronization.



(b) Attack results of ASCAD dataset with desynchronization 50.



(c) Attack results of ASCAD dataset with desynchronization 100.



(d) Attack results of AES_HD dataset.

Figure 1: Experimental results

nization is very close to that for ASCAD without desynchronization. There should be no difference in the amount of information contained in a trace between the datasets with and without desynchronization because the desynchronized ASCAD dataset is created by adding time shifts to the ASCAD dataset without desynchronization. In this sense, these results indicate that our method can close the gap in SRs between them.

The GEs in Fig. 1 show that the performance of the proposed method may be worse than that of the other methods when the number of traces used in the attack is quite small. We explain the reason for this finding. As described in Section 4.2, the convergence rates of the SR and GE curves are determined by the key candidate with the smallest c_k when the number of traces used in the attack n_a is sufficiently large. However, all the key candidates would have almost equal influence when n_a is small, and the influence of a single key candidate does not determine the convergence rate of SR and GE. In this experiment, the batch sizes of ASCAD and AES_HD were set to 500 and 1,000, respectively. Thus, the PCI loss mainly trained the models to reduce the influence of the key candidate with the smallest c_k. This would be preferable in the sense of increasing SR, although the convergence rate tends to be worse when n_a is small.

From the experimental results, we confirm that our learning method can improve the DL-SCA performance by deforming the output probability distribution of the approximate model when the NLL loss does not sufficiently train the model. This deformation makes the trained model output close to the optimal distinguisher, which validates the utility of the PCI loss as the performance evaluation metric in DL-SCA.

7 Conclusion

In this paper, we provided solutions to two important open problems in DL-SCA: (i) effect of the key value used in profiling on the attack performance, and (ii) clarification on the optimality of the loss function used during training, such as NLL loss. Furthermore, we proposed the PCI loss function for DL-SCA based on the solutions to problems (i) and (ii) above; we also introduced a new learning method using both PCI and NLL losses. Finally, we validated our analyses and theorems and demonstrated the effectiveness of our method through experimental attacks on two public datasets.

An interesting future work on the proposed method is to investigate the effect of the imbalanced data problem in DL-SCA. A comparison with some methods for solving imbalanced data problems, such as CER loss, is important. In addition, the application of the proposed method to various cryptographic hardware is worth investigating. While we focused on typical AES software and hardware implementations, more sophisticated implementations, such as hardware implementations with various countermeasures should also be investigated.

Acknowledgment

This section is intentionally left blank for the double-blind review.

A Proofs

A.1 Proof of Proposition 1

Heuser et al. showed in [HRG14] that the optimal distinguisher is given as

$$d(\mathcal{S}_a) = \arg\max_k \sum_{j=1}^{n_a} \log q_{\boldsymbol{X} \mid M, K^*}(\boldsymbol{X}_j \mid M_j, k).$$

With the Markov chain $\boldsymbol{X} \to Z^{(k^*)} \to (M, k^*)$, we have

$$d(\mathcal{S}_a) = \arg\max_k \sum_{j=1}^{n_a} \log \left(\frac{1}{q_{K^*,M}(k, M_j)} \sum_z q_{K^*,M|Z^{(k^*)}}(k, M_j \mid z) q_{Z^{(k^*)}|\boldsymbol{X}}(z \mid \boldsymbol{X}_j) q_{\boldsymbol{X}}(\boldsymbol{X}_j) \right).$$

From the assumption where the selection function is a bijection function,

$$q_{K^*,M|Z^{(k^*)}}(k, M_j \mid z) = \begin{cases} 1/|\mathcal{K}| & (g(k, M_j) = z) \\ 0 & (\text{otherwise}) \end{cases}.$$

Note that there only exists z such that $q_{K^*,M|Z^{(k^*)}}(k, M_j \mid z) \neq 0$ because kand M_j are the fixed value for j. Let $Z_j^{(k)} = g(k, M_j)$. Then,

$$d(\mathcal{S}_a) = \arg\max_k \sum_{j=1}^{n_a} \log q_{Z^{(k^*)}|\boldsymbol{X}}(Z_j^{(k)} \mid \boldsymbol{X}_j).$$

A.2 Proof of Lemma 1

It is necessary to show that SR_{n_a} and GE_{n_a} are independent of the value of β. This holds if and only if $\mathrm{rank}(k^*, n_a)$ is invariant to β. When we use probability distribution r', the rank of the secret key can be written as

$$\mathrm{rank}(k^*, n_a) = 1 + \sum_{k \in \mathcal{K}_{\setminus k^*}} \mathbb{1}_{\left\{ \frac{1}{n_a} \sum_{j=1}^{n_a} \Delta_j^{(k)} \geq 0 \right\}},$$

where $\Delta_j^{(k)} = -\log r'(Z_j^{(k^*)} \mid \boldsymbol{X}_j) + \log r'(Z_j^{(k)} \mid \boldsymbol{X}_j)$. We can rewrite the argument for the indicator function as follows:

$$\begin{aligned} \frac{1}{n_a} \sum_{j=1}^{n_a} \Delta_j^{(k)} \geq 0 &\Leftrightarrow \frac{1}{n_a} \sum_{j=1}^{n_a} \left(-\log \frac{r(Z_j^{(k^*)} \mid \boldsymbol{X}_j)^\beta}{\sum_l r(z \mid \boldsymbol{X})^\beta} + \log \frac{r(Z_j^{(k)} \mid \boldsymbol{X}_j)^\beta}{\sum_z r(z \mid \boldsymbol{X})^\beta} \right) \geq 0 \\ &\Leftrightarrow \frac{1}{n_a} \sum_{j=1}^{n_a} \left(-\log r(Z_j^{(k^*)} \mid \boldsymbol{X})^\beta + \log r(Z_j^{(k)} \mid \boldsymbol{X})^\beta \right) \geq 0 \\ &\Leftrightarrow \frac{1}{n_a} \sum_{j=1}^{n_a} \left(-\log r(Z_j^{(k^*)} \mid \boldsymbol{X}_j) + \log r(Z_j^{(k)} \mid \boldsymbol{X}_j) \right) \geq 0. \end{aligned}$$

Thus, $\mathrm{rank}(k^*, n_a)$ is invariant to β.

A.3 Proof of Theorem 4

From the inequality $\log(x + 1) \leq x$, we have

$$\begin{aligned} \mathbb{E} L_{\mathrm{RkL}}(\hat{\theta}) &\leq \sum_{k \in \mathcal{K}_{\setminus k^*}} \log_2(e) \mathbb{E} \exp \left(\alpha_k \sum_{j=1}^{n_a} \Delta_j^{(k)} \right) \\ &= \sum_{k \in \mathcal{K}_{\setminus k^*}} \log_2(e) \exp(\alpha_k n_a \mu_k) \prod_{j=1}^{n_a} \mathbb{E} \exp \left(\alpha_k (\Delta_j^{(k)} - \mu_k) \right). \end{aligned}$$

Using the inequality $1+x \le e^x$, we obtain

$$\mathbb{E}\exp\left(\alpha_k(\Delta_j^{(k)}-\mu_k)\right) = \sum_{i=0}^{\infty}\frac{\alpha_k^i\mathbb{E}(\Delta_j^{(k)}-\mu_k)^i}{i!} \le 1+\sum_{i=2}^{\infty}\frac{\alpha_k^i\sigma_k^2R_k^{i-2}}{i!}$$
$$= 1+\frac{\sigma_k^2}{R_k^2}\left(\exp\left(\alpha_kR_k\right)-\alpha_kR_k-1\right) \le \exp\left(\frac{\sigma_k^2}{R_k^2}\left(\exp\left(\alpha_kR_k\right)-\alpha_kR_k-1\right)\right).$$

Thus,

$$\exp(\alpha_kn_a\mu_k)\prod_{j=1}^{n_a}\mathbb{E}\exp\left(\alpha_k(\Delta_j^{(k)}-\mu_k)\right) \le \exp(\alpha_kn_a\mu_k)\exp\left(\frac{n_a\sigma_k^2}{R_k^2}\left(e^{\alpha_kR}-\alpha_kR-1\right)\right).$$

Therefore, we have

$$\inf_{\boldsymbol{\alpha}}\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta}) \le \inf_{\boldsymbol{\alpha}}\sum_{k\in\mathcal{K}_{\setminus k^*}}\log_2(e)\exp(\alpha_kn_a\mu_k)\exp\left(\frac{n_a\sigma_k^2}{R_k^2}\left(e^{\alpha_kR}-\alpha_kR-1\right)\right).$$

We have $\alpha_k = (1/R_k)\log(-R_k\mu_k/\sigma_k^2+1)$ by minimizing the right-hand side. Substituting this, we obtain:

$$\inf_{\boldsymbol{\alpha}}\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta}) \le \log_2(e)\exp(n_aL_{\mathrm{PCI}}(\hat{\theta})).$$

B Definition of the ranking loss

Proposition 4 (Generative form of ranking loss)**.** *Define the ranking loss by*

$$L_{\mathrm{RkL}}(\hat{\theta}) = \sum_{k\in\mathcal{K}_{\setminus k^*}}\log_2\left(1+e^{\alpha_k\sum_{j=1}^{n_a}\Delta_j^{(k)}}\right).$$

We have $\mathrm{GE}_{n_a} \le 1+\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta})$ *and* $\mathrm{SR}_{n_a} \ge 1-\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta})$.

Proof. Let $\phi_\alpha(s) = \log_2(1+e^{\alpha s})$ be a logistic loss function. Note that $\mathbb{1}_{X\ge0} \le \phi_\alpha(X)$ holds for any random variable X. Thus,

$$\mathrm{rank}(k^*,n_a) = 1+\sum_{k\in\mathcal{K}_{\setminus k^*}}\mathbb{1}_{\{\frac{1}{n_a}\sum_{j=1}^{n_a}\Delta_j^{(k)}\ge0\}} \le 1+\sum_{k\in\mathcal{K}_{\setminus k^*}}\phi_{\alpha_k}\left(\sum_{j=1}^{n_a}\Delta_j^{(k)}\right).$$

From this, we immediately have $\mathrm{GE}_{n_a} \le 1+\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta})$. In addition, Lemma 2 yields that $\mathrm{SR}_{n_a} \ge 1-\mathbb{E}L_{\mathrm{RkL}}(\hat{\theta})$. □

C Bennett's inequality

Theorem 5 (Bennett's inequality [Ver18, Ben62])**.** *Let $Y_1, Y_2, \ldots, Y_n$ be independent random variables. Assume that $|Y_i-\mathbb{E}Y_i| \le R$ almost surely for every i. Then, for any $t>0$, we have*

$$\Pr\left(\frac{1}{n}\sum_{i=1}^{n}(Y_i-\mathbb{E}Y_i)\ge t\right) \le \exp\left(n\frac{\sigma^2}{R^2}h\left(\frac{Rt}{\sigma^2}\right)\right),$$

where $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}(Y_i-\mathbb{E}Y_i)^2$, *and* $h(u) = (1+u)\log(1+u)-u$.

References

[AAB+15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, Lecture Notes in Computer Science, pages 16–29, Berlin, Heidelberg, 2004. Springer.

[Ben62] George Bennett. Probability inequalities for the sum of independent random variables. *Journal of the American Statistical Association*, 57(297):33–45, 1962.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[BPS+20] Ryad Benadjila, Emmanuel Prouff, Rémi Strullu, Eleonora Cagli, and Cécile Dumas. Deep learning for side-channel analysis and introduction to ASCAD database. *Journal of Cryptographic Engineering*, 10(2):163–188, June 2020.

[C+15] Francois Chollet et al. Keras, 2015.

[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*, volume 10529 of *Lecture Notes in Computer Science*, pages 45–68. Springer, 2017.

[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, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, volume 5747 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 2009.

[CT06] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006.

[dCGRP19] Eloi de Chérisey, Sylvain Guilley, Olivier Rioul, and Pablo Piantanida. Best information is most successful. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2019, Issue 2:49–79, 2019.

[DZFL14] A. Adam Ding, Liwei Zhang, Yunsi Fei, and Pei Luo. A statistical model for higher order dpa on masked devices. In *CHES*, pages 147–169. Springer, 2014.

[FDLZ15] Yunsi Fei, A. Adam Ding, Jian Lao, and Liwei Zhang. A statistics-based success rate model for DPA and CPA. *Journal of Cryptographic Engineering*, 5(4):227–243, November 2015.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[GHR15] Sylvain Guilley, Annelie Heuser, and Olivier Rioul. A Key to Success. In Alex Biryukov and Vipul Goyal, editors, *Progress in Cryptology – INDOCRYPT 2015*, Lecture Notes in Computer Science, pages 270–290, Cham, 2015. Springer International Publishing.

[HHGG20] Benjamin Hettwer, Tobias Horn, Stefan Gehrer, and Tim Güneysu. Encoding power traces as images for efficient side-channel analysis. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 46–56, 2020.

[HNL$^+$19] Zhen Huang, Tim Ng, Leo Liu, Henry Mason, Xiaodan Zhuang, and Daben Liu. SNDCNN: Self-normalizing deep CNNs with scaled exponential linear units for speech recognition. October 2019.

[HRG14] Annelie Heuser, Olivier Rioul, and Sylvain Guilley. Good is not good enough - deriving optimal distinguishers from communication theory. In *CHES*, pages 55–74. Springer, 2014.

[HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning — Data Mining, Inference, and Prediction.* Springer, second edition, 2009.

[ISUH21] Akira Ito, Kotaro Saito, Rei Ueno, and Naofumi Homma. Imbalanced data problems in deep learning-based side-channel attacks: Analysis and solution. *IEEE Transactions on Information Forensics and Security*, pages 1–1, 2021.

[KB15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of 3rd International Conference on Learning Representations*, 2015.

[LPR$^+$14] Victor Lomneacute;, Emmanuel Prouff, Matthieu Rivain, Thomas Roche, and Adrian Thillard. How to estimate the success rate of higher-order side-channel attacks. In *CHES*, pages 35–54. Springer, 2014.

[MDP19] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A comprehensive study of deep learning for side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, Issue 1:348–375, 2019.

[MHM14] Zdenek Martinasek, Jan Hajny, and Lukas Malina. Optimization of power analysis using neural network. In Aurélien Francillon and Pankaj Rohatgi, editors, *Smart Card Research and Advanced Applications*, pages 94–107, Cham, 2014. Springer International Publishing.

[MOS11] S. Mangard, E. Oswald, and F.-X. Standaert. One for all – all for one: Unifying standard differential power analysis attacks. *IET Information Security*, 5(2):100–110, June 2011.

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

[Riv09] Matthieu Rivain. On the Exact Success Rate of Side Channel Analysis in the Gaussian Model. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography*, pages 165–183, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[RWPP21] Jorai Rijsdijk, Lichao Wu, Guilherme Perin, and Stjepan Picek. Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021.

[RZC$^+$21] Damien Robissout, Gabriel Zaid, Brice Colombier, Lilian Bossuet, and Amaury Habrard. Online performance evaluation of deep learning networks for profiled side-channel analysis. In Guido Marco Bertoni and Francesco Regazzoni, editors, *Constructive Side-Channel Analysis and Secure Design*, pages 200–218, Cham, 2021. Springer International Publishing.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In *Cryptographic Hardware and Embedded Systems*, Lecture Notes in Computer Science, pages 30–46. Springer, 2005.

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

[TPR13] Adrian Thillard, Emmanuel Prouff, and Thomas Roche. Success through confidence: Evaluating the effectiveness of a side-channel attack. In *CHES*, pages 21–36. Springer, 2013.

[Ver18] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, first edition, September 2018.

[ZBD$^+$20] Gabriel Zaid, Lilian Bossuet, François Dassance, Amaury Habrard, and Alexandre Venelli. Ranking loss: Maximizing the success rate in deep learning side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021, Issue 1:25–55, 2020.

[ZBHV19] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli. Methodology for efficient cnn architectures in profiling attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, Issue 1:1–36, 2019.

[ZDF20] Ziyue Zhang, A. Adam Ding, and Yunsi Fei. A fast and accurate guessing entropy estimation algorithm for full-key recovery. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, Issue 2:26–48, 2020.

[ZZN$^+$20] Jiajia Zhang, Mengce Zheng, Jiehui Nan, Honggang Hu, and Nenghai Yu. A novel evaluation metric for deep learning-based side channel analysis and its extended application to imbalanced data. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, Issue 3:73–96, 2020.