Cross-Algorithm Deep Learning-based Non-Profiled Side-Channel Attacks Exploiting Symmetric Leakage

Jintong Yu, Yuxuan Wang, Zixin He, Yihan Nie, Yubo Zhao, Zhiliang An, Yipeng Shi, Pei Cao, Chi Zhang, and Dawu Gu *Member, IEEE*

***Abstract*—Deep Learning-based Non-profiled Side-Channel Analysis (DL-NSCA) enables automatic feature extraction without a profiling device, but existing approaches mainly target non-linear operations, requiring prior knowledge of the algorithm's unique non-linear structure and computable non-linear intermediate values. These limit applicability in analyzing proprietary or undisclosed implementations and in settings where plaintext/ciphertext are masked by unknown randomness (e.g., tweaks or nonces).**

We observe that linear operations are fundamental as common cryptographic primitives appearing at the beginning or end of algorithms in conjunction with the secret key, and are widely used to mask sensitive input/output. Motivated by this observation, we propose a new DL-NSCA perspective that targets the outputs of linear operations, referred to as symmetric leakage, to enable cross algorithm attacks. The main limitation of the prior distinguisher lies in their reliance on a simplistic correspondence between deep learning metrics and side channel information. This leads to two issues: the effectiveness of the distinguisher varies significantly with the chosen training epoch, and a wrong key inducing a negative correlation may be indistinguishable from the correct key under symmetric leakage. To address this, we provide a formal algebraic characterization of the relationship between the structure of the leakage function and the number of maxima given from the distinguisher. Guided by this theory, we propose a new distinguisher, VS-GBA, an epoch-invariant distinguisher that interprets SCA information from deep learning metrics and approaches the theoretical optimum. It is applicable to both the asymmetric leakage and symmetric leakage through a structure-aware screening criterion. Our experiments show that, on the high-noise 32-bit ARM Cortex-M4 device, when the core operations are protected, asymmetric leakage analysis fails to recover the keys for all three evaluated algorithms within the maximum trace budget ($GE = 70$ for masked AES, $GE = 27$ for masked PRESENT, and $GE = 66$ for masked ASCON). In contrast, VS-GBA, which targets symmetric leakage, recovers the key with a 100% success rate using 8,000, 8,500, and 16,000 traces, respectively. Furthermore, we present a DL-NSCA attack on XTS-AES (NIST SP 800-38E), extending DL-NSCA to scenarios where plaintext/ciphertext is masked by a secret tweak.

***Index Terms*—Side-Channel Analysis, Deep Learning, Non-profiled Attack, Cross-Algorithm Attack, Linear Operations.**

I. Introduction

SIDE-CHANNEL Analysis (SCA) poses a persistent threat to cryptographic implementations by exploiting unintended physical leakage, such as power consumption and electromagnetic emanations, to recover secret information. Over the past two decades, SCA has evolved into a powerful class of attack techniques, capable of breaking implementations that are theoretically secure at the algorithmic level [1]. SCA is typically categorized into two attack models: profiled and non-profiled. In profiled attacks, the attacker must have access to a fully controllable device that is identical to the target. The attacker characterizes the side-channel behaviors on the profiling device, and then leverages them to recover the secret key from the target device. Under this prerequisite, profiled SCA has demonstrated remarkable effectiveness, such as template attacks [2]–[4] and deep learning-based attacks [5]–[9]. However, this prerequisite is often infeasible to meet in practice, especially for closed devices. Non-profiled SCA relaxes this assumption and therefore is more applicable to many practical scenarios [10], [11].

Non-profiled SCA employs a divide-and-conquer strategy, typically targeting subkeys at the byte level. Traditionally, statistical metrics such as Pearson correlation in CPA [12] or mutual information in MIA [13] are used to identify the correct key guess through correlation maxima. However, modern countermeasures [1], [14] and the difficulty of manual Point of Interest (PoI) identification in unknown implementations have limited these methods. A pivotal shift occurred with the introduction of Differential Deep Learning Analysis (DDLA) [15], which leverages neural networks to automate feature extraction and PoI discovery. Subsequent advancements [16]–[19] have focused on optimizing computational efficiency, culminating in state-of-the-art frameworks like ConvWIN-MCR [19], which successfully analyzes masked raw traces exceeding 100,000 feature points.

Existing SCA research predominantly targets the outputs of non-linear operations, such as the first round SubBytes in AES. However, this focus imposes two critical limitations that hinder applicability in real-world scenarios. First, analyzing non-linear operations requires prior knowledge of the cryptographic algorithm. Since non-linear designs are often the defining characteristic of a cipher, they vary significantly across different primitives. For example, while AES, ASCON, and PRESENT all utilize SBoxes, their respective mapping tables are entirely distinct. However, this requirement is difficult to meet when evaluating proprietary or undisclosed algorithms, IP-protected hardware modules, or security evaluations of IoT devices [20]–[23]. Second, this approach depends on the computability of the non-linear output. For instance, given a plaintext p and a key guess k, the AES SBox output is calculated as $\mathrm{SBox}[p \oplus k]$. In modern cryptographic designs, however, plaintext/ciphertext is frequently obscured by random values, such as nonces in authenticated encryption (e.g., Ascon) [24] and tweaks in

Jintong Yu, Yuxuan Wang, Zixin He, Yihan Nie, Yubo Zhao, Zhiliang An, Yipeng Shi, Pei Cao, Chi Zhang, and Dawu Gu (Corresponding authors: Chi Zhang; Dawu Gu.) are with the School of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, Shanghai Pudong Institute of Cryptology (SPIC), China (e-mail: jintongyu@sjtu.edu.cn; 18588297218@sjtu.edu.cn; zcsjtu@sjtu.edu.cn; dwgu@sjtu.edu.cn).

tweakable block ciphers (e.g., XTS-AES) [25], [26], which effectively masks the input/output data required to compute these non-linear intermediate values.

To address these limitations, we offer an alternative perspective for Deep Learning-based Non-profiled Side-Channel Analysis (DL-NSCA), namely analysis targeting leakages from the output of linear operations, which offers significant potential for broader applicability. First, many cryptographic primitives employ identical linear components; for instance, the XOR operation is ubiquitous across block ciphers. These common linear operations may generate more universal and fundamental key-dependent leakage features, thereby enabling cross-algorithm SCA. Second, the key-mixing linear operations in most standard symmetric ciphers are structurally positioned at the beginning or end of the encryption procedure, for example, the AddRoundKey at the start of AES, the first-round XOR in PRESENT, and the tag-generation XOR at the end of ASCON. This structural regularity means that an attacker can identify key-mixing linear operation's region by simply examining the beginning or end of the trace. Third, due to their lower computational overhead, linear operations are frequently used to mask the plaintext/ciphertext upon which non-linear computations depend. A prominent example is the XTS-AES algorithm (recommended by NIST SP 800-38E), where the input is XORed with a random value before being processed by the internal AES module [27].

However, the distinguisher used in all existing DL-NSCA works targets non-linear operations, and how to analyze linear operations under this framework remains an open question. The primary challenge for the prior distinguisher lies in the simple correspondence between deep learning metrics and side channel results. This leads to two issues, the first is that the effectiveness of the distinguisher varies significantly with the chosen training epoch, and the second is that the implicit assumption of a unique correlation maximum does not hold for linear operations under DL-NSCA.

Typically, for each key guess k, a distinguisher calculates the correlation between the observed traces and the hypothetical leakage (e.g., the Hamming weight of the attack point $f(k)$). It is known that, in traditional CPA, targeting the output of an XOR operation may yield two correlation extrema with opposite signs. One extremum corresponds to the correct key, while the other corresponds to its complement-related candidate [1], [12], [28], [29]. A signed CPA distinguisher can separate these extrema by the sign of the correlation. In DL-NSCA, however, the correlation is evaluated through neural network training metrics. Specifically, the model trained with strongly dependent labels converges faster, yielding a lower loss or higher accuracy regardless of whether the dependency is positive or negative. This criterion is therefore closer to maximizing the absolute correlation, rendering the two signed extrema of an XOR operation indistinguishable to the prior distinguisher. Moreover, no existing DL-NSCA work has explicitly modeled this sign-discarding ambiguity for linear key-mixing targets.

Contributions. We use **asymmetric leakage** to denote targets for which the absolute-correlation maximum is unique at the correct key under the stated leakage model and input distribution. We use **symmetric leakage** to denote the common invertible linear key-mixing case in which two complement-related key candidates attain the same absolute-correlation maximum. The main contributions of this work are as follows:

1. **Formal characterization of absolute-correlation maxima.** We derive how the maxima of absolute-correlation depend on the target function, leakage model, input distribution. We prove that asymmetric leakage corresponds to a unique maximizer of the absolute correlation, whereas symmetric leakage gives rise to two maximizers of the absolute correlation, namely the correct key and its complement.
2. **A robust DL-NSCA distinguisher, VS-GBA.** We propose VS-GBA, a new key distinguisher for DL-NSCA. Guided by theoretical analysis, it establishes a principled link between deep learning metrics and side channel information. This yields two concrete advantages: (i) it is epoch-invariant, automatically identifying discriminative epochs without manual selection; and (ii) it does not require a unique correlation maximum, explicitly handling both the single-maximum case (asymmetric leakage) and the dual-maximum case (symmetric leakage) through a structure-aware screening criterion. Both advantages are empirically validated through experiments on public datasets.
3. **Cross-algorithm evaluation of symmetric leakage.** We evaluate masked AES, PRESENT, and ASCON on a real-world, high-noise 32-bit ARM Cortex-M4 device. On this platform, the evaluated protected asymmetric targets remain poorly ranked by Guessing Entropy(GE) ($GE = 70$ for masked AES, $GE = 27$ for masked PRESENT, $GE = 66$ for masked ASCON), while symmetric leakage enables candidate-pair recovery within 8,000, 8,500, and 16,000 traces, respectively.
4. **A DL-NSCA attack on tweakable block ciphers.** We present a DL-NSCA method that successfully recovers secret keys from XTS-AES (NIST SP 800-38E), a standardized tweakable block cipher widely deployed in disk encryption, where the plaintext/ciphertext is masked by a secret tweak. Prior DL-NSCA methods cannot compute the non-linear intermediate values under this scenario; by instead targeting symmetric leakage, our method recovers the true AddRoundKey input without knowledge of the tweak, xtending DL-NSCA to scenarios where plaintext/ciphertext is masked by a secret tweak.

II. Background

A. Notation

We use calligraphic letters to represent sets like $\mathcal{X}$, with bold upper-case letters (e.g., $\mathbf{X}$) denoting matrices, bold lower-case letters (e.g., $\mathbf{x}$) representing vectors, and lowercase letters (e.g., x) indicating the scalar values or random variables.

B. Threat model and scope

We consider a non-profiled attacker who can collect side-channel traces from the target device under a fixed secret key, together with the corresponding public inputs or outputs. The

attacker does not possess a profiling clone device and does not know mask values. The attacker is assumed to know the cipher family or to have identified a key-dependent linear key-mixing pattern, such as an XOR between a public byte and a secret byte, but is not assumed to know implementation-level details such as memory layout or instruction scheduling.

The attack is therefore not a fully black-box analysis of an arbitrary binary. It requires a trace segment containing the targeted operation. In our cross-algorithm experiments, this segment is available because the attack points are at the beginning or end of the primitive. Automatic window localization for unknown implementations is an important but separate problem; this work evaluates the key-distinguisher behavior once a plausible target window has been selected.

For symmetric leakage, the output of an invertible linear key-mixing operation identifies a candidate pair $(k_*, k_* \oplus \mathbf{1})$ under the HW model, not necessarily the exact byte key.

C. Non-profiled SCA

Non-profiled SCA takes a divide-and-conquer strategy recovering a n-bit key each time. Denote k_* the fixed but unknown secret key, key space $\mathcal{K} = \{0, 1, \ldots, 2^n - 1\}$. An attacker has access to a set of N known, random data $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ (typically plaintexts or ciphertexts), and their corresponding set of N side-channel traces $\mathcal{T} = \{\bar{\mathbf{x}}_\mathbf{1}, \bar{\mathbf{x}}_\mathbf{2}, \ldots, \bar{\mathbf{x}}_\mathbf{N}\}$. Each trace $\bar{\mathbf{x}}_\mathbf{i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,M})$ is a vector of M feature points and is assumed to comprise both the actual data-dependent leakage and noise. For each key guess $k \in \mathcal{K}$ and random data d, the output of a specific operation $f(d, k)$ is targeted. For example, in the case of non-linear leakage involving a SBox output, defined as $f(d, k) = SBox[d \oplus k]$, the SBox mapping is algorithm-specific. While this operation is common to algorithms like AES, PRESENT, and ASCON, their underlying mathematical expressions differ. Conversely, linear leakage associated with the XOR operation follows a universal form, $f(d, k) = d \oplus k$, regardless of the algorithm. Finally, assuming a leakage model H, the attacker can then generate a set of hypothetical leakage $\mathcal{H}_k = \{h_{k,1}, h_{k,2}, \ldots, h_{k,N}\}$, with each being derived from the formula $h_{k,i} = H(f(d_i, k))$. The commonly used model is the Hamming Weight (HW) model, i.e., $h_{k,i} = HW(f(d_i, k))$.

1) Traditional non-profiled SCA: Traditional non-profiled SCA methods utilize the key distinguisher, usually a statistical tool, denoted as Corr to measure the correlation between the hypothetical leakage vector $\mathbf{h_k} = (h_{k,1}, h_{k,2}, \ldots, h_{k,N})$ and the actual leakage at leakage point p_0, i.e., $\mathbf{x_{p_0}} = (x_{1,p_0}, x_{2,p_0}, \ldots, x_{N,p_0})$, where the key guess corresponding to the maximum correlation can be considered as the correct key guess:

$$k_* = \arg\max_{k \in \mathcal{K}} \mathrm{Corr}(\mathbf{h_k}, \mathbf{x_{p_0}}). \tag{1}$$

As implied by the equation (1), traditional non-profiled SCA primarily targets single-point leakage. Alternatively, they can analyze multi-point leakage, but this typically requires prior knowledge of the specific implementation. To this end, researchers have proposed various statistical tools to serve as the key distinguisher, such as the Pearson Correlation Coefficient for CPA [12] and Mutual Information for MIA [30]. As Doget et al. have demonstrated, most single-point attacks can be reformulated as a CPA with a different hypothetical power model [31]. In this study, Corr is calculated using the Pearson correlation coefficient.

2) DL-NSCA: In contrast to statistical tools used in traditional methods, DL-NSCA utilizes deep learning training metrics to evaluate the correlation between side-channel traces and hypothetical leakage. We term this the Deep Learning (DL) key distinguisher. Since the hypothetical leakage corresponding to the correct key guess exhibits the highest correlation with the traces, its neural network converges faster than those for incorrect key guesses. This is reflected in the training metrics that as the number of epochs increases, the model for the correct key guess shows a more rapid decrease in the loss value or a faster increase in accuracy [15]. As both metrics reflect the network's training progress, we primarily consider the loss value, noting that the conclusions drawn are equally applicable to the accuracy metric.

For each key guess k, DL-NSCA trains a neural network, denoted $f_{\mathbf{\Theta}_k}$, which takes the set of side-channel traces $\mathcal{T} = \{\bar{\mathbf{x}}_\mathbf{1}, \bar{\mathbf{x}}_\mathbf{2}, \ldots, \bar{\mathbf{x}}_\mathbf{N}\}$ as input and the corresponding hypothetical leakage $\mathcal{H}_k = \{h_{k,1}, h_{k,2}, \ldots, h_{k,N}\}$ as labels. Let L_t be the loss function value at training epoch t. The training is typically performed up to a maximum number of epochs, t_{max}, such that $t \in \{0, 1, \ldots, t_{max}\}$. Existing methods assume that the network converges at a specific epoch t_0, and the correct key guess, k_*, is then determined as follows:

$$k_* = \arg\min_{k \in \mathcal{K}} L_{t_0}(f_{\mathbf{\Theta}_k}(\mathcal{T}), \mathcal{H}_k). \tag{2}$$

3) SCA evaluation metrics: Guessing Entropy (GE) is the commonly used metric in SCA, as standardized in ISO/IEC 17825:2024. In DL-SCA, we utilize the output L_{t_0} as the score for each key candidate. By sorting these scores in descending order, we obtain a key ranking vector $\mathbf{G} = (G_0, G_1, \ldots, G_{|\mathcal{K}|-1})$, where G_0 represents the most probable candidate. GE is defined as the average index of the correct key within $\mathbf{G}$ over multiple experiments. A $GE = 0$ implies that $G_0 = k_*$ is consistently in all experiments, which means a successful attack.

Exact-key GE and candidate-pair GE are different metrics. The exact-key rank is the index of k_* in the full key ranking, whereas the candidate-pair rank is the best rank of the equivalence class containing k_* and its complement-related candidate. In the symmetric-leakage setting, the correct key and its complement are indistinguishable. Therefore, results under symmetric leakage report candidate-pair GE or equivalence-class recovery; any remaining ambiguity can be resolved by a residual search over the finite candidate space or by combining multiple attack stages.

D. Peak detection algorithm

Peak detection constitutes a fundamental task in signal processing, essential for identifying critical features such as event timings, spectral components, or transient anomalies within temporal or spectral data. The FIND_PEAKS algorithm serves as a widely adopted computational tool for this purpose [32],

and it has become integral to diverse applications, including spectral analysis [33], biomedical signal processing [34], and vibration monitoring [35].

This approach operates by systematically identifying local maxima within a discrete signal sequence. The core methodology involves evaluating each data point against its local neighborhood, typically defined by a symmetric window, to confirm its status as a peak. To ensure robustness, threshold-based constraints such as minimum peak prominence are applied to distinguish significant features from noise or minor fluctuations.

E. Rank aggregation algorithm

Rank aggregation is a critical problem in information retrieval, social choice theory, and data fusion, where the goal is to combine multiple preference rankings (e.g., from voters, search algorithms, or feature-based models) into a single consensus ranking. This task arises in diverse applications such as meta-search engines, ensemble learning, recommendation systems, and collective decision-making. A fundamental challenge lies in resolving conflicts between input rankings while preserving meaningful global orderings.

Among classical rank aggregation methods, the Borda Count algorithm, proposed by Jean-Charles de Borda, remains a widely studied and practical approach due to its simplicity, computational efficiency, and theoretical properties [36]. The core mechanism assigns scores to candidates based on their positions in each input ranking. For a ranking of n candidates, a candidate ranked k-th receives a score of $n - k$ from that ranking. Scores from all input rankings are aggregated (typically summed), and candidates are ordered by their total Borda scores in descending order to form the final consensus ranking. The Borda Count algorithm offers several notable advantages. Firstly, it leverages the full spectrum of preference information provided by each voter, making the final result more comprehensive and robust. Secondly, it tends to favor consensus candidates who are ranked highly by many, rather than polarized candidates who may be controversial.

F. ASCADv1 public dataset

The ASCADv1 dataset is a public benchmark containing electromagnetic traces from an 8-bit ATmega8515 microcontroller[1]. The device implements an AES-128 encryption algorithm protected with a first-order Boolean masking scheme [37]. The dataset is provided in two main versions: fixed key (ASCAD_F_R) and variable key (ASCAD_V_R). As our work targets a non-profiled scenario, we utilize the attack traces from the ASCAD_V_R subset, which are captured when the device operates under a single, constant key. The raw traces from the ASCAD_F_R and ASCAD_V_R datasets consist of 100000 and 250000 feature points, respectively.

III. Limitations of the Key Distinguisher in DL-NSCA

In this section, we analyze the limitations of the prior distinguisher, which primarily manifest in two aspects. First, the loss values in equation (2) do not have the capability to distinguish the correct key guess from incorrect key guesses at every epoch t, and the effective epoch t_0 at which it becomes capable of doing so is **unknown and varies** with the number of input traces, hyperparameter settings, and other factors. This limitation applies to both asymmetric and symmetric leakage, and is elaborated in Section III-A. Second, when targeting symmetric leakage, an additional and more fundamental failure arises: the DL key distinguisher cannot uniquely identify the correct key due to the **non-uniqueness** of key guesses achieving maximum absolute correlation, regardless of the epoch chosen. This issue is rooted in a property previously observed in CPA where XOR targets can produce two signed extrema [1], [12], [28], [29], while loss-based DL metrics discard the sign and therefore merge them into equal absolute-correlation maxima. This sign-discarding ambiguity has not been explicitly addressed within the DL-NSCA framework. We present its formal theoretical derivation in Section III-B.

A. The Condition for an Effective DL Key Distinguisher

DL key distinguisher replaces traditional distinguishers by using the loss values L_{t_0} at the model's convergence epoch t_0. This loss value characterizes the correlation between the side-channel traces and the hypothetical leakage, as shown in equation (2). The training process of the neural network comprises three distinct phases [38]:

1. Underfitting phase: In the initial stage of training, the model fails to learn the features from the data effectively.
2. Good fit phase: In the good fit phase, both training loss and validation loss decrease and stabilize at a low level, reaching an optimal balance where the model has successfully learned generalizable patterns without capturing excessive noise.
3. Overfitting phase: In the overfitting phase, the training loss continues to decrease to very low levels while the validation loss begins to rise, indicating that the model is memorizing the specific noise of the training set rather than learning generalizable features.

To prevent overfitting, the most common strategy in supervised learning is to monitor the loss on a held-out validation set and apply early stopping when validation loss begins to rise. However, as we demonstrate below, the absence of a validation set is not the only fundamental difficulty. The deeper issue is that the epoch at which the model enters the good fit phase is not a fixed quantity: it shifts substantially with the number of input traces, the neural network architecture, and other hyperparameter settings. Existing methods attempt to handle this by manually fixing the value of t_0 through observation of the loss curves, which is imprecise and non-generalizable across experimental configurations.

Figure 1 presents the evaluation results of the ConvWIN-MCR model on the ASCAD_F_R dataset under different numbers of training epochs and trace quantities. The right subplot displays the loss curve when training with 20000 input traces. From this single-condition view, the three training phases are visually identifiable, and one might expect that a statistical method, such as a change-point detector applied to

[1]https://github.com/ANSSI-FR/ASCAD

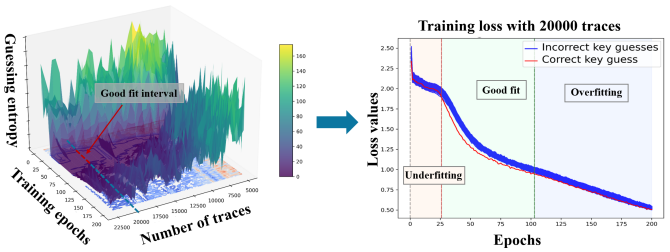

Fig. 1. Unfixed good fit interval on the ASCAD_F_R dataset

the loss trajectories, could automatically locate the good fit interval.

However, the right subplot reflects only one experimental condition. The left subplot reveals a fundamentally different picture: it plots a heatmap of GE across varying numbers of input traces and training epochs. The contour where $GE = 0$ precisely delineates the good fit region, and a critical observation emerges. The good fit boundaries shift substantially as the number of traces changes, and they are even more sensitive to changes in the neural network architecture and hyperparameters.

Crucially, this epoch-dependency is orthogonal to model capability. The same trained model can appear to succeed or fail depending solely on which epoch is selected for evaluation. Any fixed epoch chosen by inspecting one experimental condition will inevitably fall outside the good fit region under different configurations, rendering it an unreliable and non-generalizable evaluation criterion.

We define the vector of loss values for all key guesses at a given epoch t, $\mathbf{l_t} = (L_t(f_{\mathbf{\Theta}_0}(\mathcal{T}), \mathcal{H}_0), L_t(f_{\mathbf{\Theta}_1}(\mathcal{T}), \mathcal{H}_1), \ldots, L_t(f_{\mathbf{\Theta}_{255}}(\mathcal{T}), \mathcal{H}_{255}))$ as the **Key Guess Vector** (KGV). The core challenge lies in identifying the epochs for which the KGV accurately represents the correlation between traces and hypothetical leakage. Formally, we aim to find a set of epochs $\mathcal{T}_{conv} \subseteq \{0, 1, \ldots, t_{max} - 1\}$, where for any epoch t_0 within this set, the DL key distinguisher is effective.

B. Limitations of prior key distinguisher

As defined in equation (1), the CPA distinguisher measures the Pearson correlation coefficient $\rho_k = \mathrm{Corr}(h_k, x_{p_0})$ between the hypothetical leakage h_k and the observed leakage x_{p_0}. Prior studies have found that when targeting the output of an XOR operation, CPA may produce two extrema with opposite signs: the correct key induces a positive correlation peak, while a structurally related wrong key can induce a negative correlation peak [1], [12], [28], [29]. In a signed CPA distinguisher these two extrema are distinguishable by the sign of ρ_k. The difficulty for DL-NSCA arises because the loss- or accuracy-based distinguisher does not preserve this sign information.

With the introduction of deep learning, the Pearson statistic is replaced by neural network training metrics (the lowest loss value at a given epoch) as established in Section II-C. Since a neural network's loss function measures the strength of statistical dependency regardless of whether the induced relation is positively or negatively correlated, this criterion is theoretically closer to maximizing the absolute value of the correlation coefficient:

$$k_* = \arg\max_{k \in \mathcal{K}} |\mathrm{Corr}(\mathbf{h_k}, \mathbf{x_{p_0}})| = \arg\max_{k \in \mathcal{K}} |\rho_k|. \quad (3)$$

Thus, the ambiguity appears whenever a second key guess induces a hypothetical leakage vector that is negatively correlated with the correct-key leakage and the distinguisher discards the sign. Linear XOR key mixing is the most common and practically relevant source of this ambiguity in our setting, because complementing the XOR output maps the HW leakage to its negative affine counterpart. Existing DL-NSCA methods commonly target non-linear SBox outputs, for which the absolute-correlation maximum is usually expected to be unique at the correct key. However, non-linearity alone is not sufficient to guarantee this property. A non-linear function may preserve Hamming weight, or may produce wrong-key correlations under a particular leakage model and input distribution. We therefore treat uniqueness as a property of the complete tuple consisting of the target function, leakage model, and input distribution, rather than as a direct consequence of non-linearity.

Definition 1 (Asymmetric leakage). For a target function f, leakage model H, and input distribution $\mathcal{D}$, let $\mathbf{h_k} = (H(f(d_1, k)), \ldots, H(f(d_N, k)))$ denote the hypothetical leakage vector for key guess k. We say that this target induces *asymmetric leakage* if $|\rho_k|$ in equation (3) has a unique global maximum at $k = k_*$.

Following the standard linear leakage model [12], the observed leakage can be written as

$$\mathbf{x_{p_0}} = a\mathbf{h_{k_*}} + \boldsymbol{\epsilon},$$

where a represents the gain, and $\boldsymbol{\epsilon}$ is independent noise. For any k, we have

$$|\rho_k| = \left| \frac{aCov(\mathbf{h_{k_*}}, \mathbf{h_k})}{|a|\sigma_{\mathbf{h_{k_*}}}\sigma_{\mathbf{k}}} \right| = |\mathrm{Corr}(\mathbf{h_{k_*}}, \mathbf{h_k})|. \quad (4)$$

The following theorem gives a precise characterization for the conditions of $|\rho_k| = 1$.

Theorem 1. *Let $f(\mathbf{d}, \mathbf{k}) = g(\mathbf{d} \oplus \mathbf{k})$, where $g : GF(2)^n \to GF(2)^n$ is a bijection, $\mathbf{k}$ is the constant key, and $\mathbf{d}$ is uniformly distributed over $GF(2)^n$. Define*

$$S_+ = \{\delta \in GF(2)^n \mid HW(g(x \oplus \delta)) = HW(g(x)),\ \forall x\},$$
$$S_- = \{\delta \in GF(2)^n \mid HW(g(x \oplus \delta)) = n - HW(g(x)),\ \forall x\}.$$

Then $|\rho_k| = 1$ if and only if $k \oplus k_* \in S_+ \cup S_-$. Moreover, for any bijection g, $S_+ = \{0\}$. Hence the absolute-correlation maximum is unique at $k = k_*$ if and only if $S_- = \emptyset$.

Proof. Let $x = \mathbf{d} \oplus k_*$ and $\delta = k \oplus k_*$. Then $\mathbf{h_{k_*}} = HW(g(x))$ and $\mathbf{h_k} = HW(g(x \oplus \delta))$. Since g is a bijection and x is uniform, $HW(g(x))$ and $HW(g(x \oplus \delta))$ both range over $\{0, \ldots, n\}$ with variance $n/4$.

If $\delta \in S_+$, then $\mathbf{h_k} = \mathbf{h_{k_*}}$ pointwise and $\rho_k = 1$. If $\delta \in S_-$, then $\mathbf{h_k} = n - \mathbf{h_{k_*}}$ pointwise and $\rho_k = -1$. Thus $\delta \in S_+ \cup S_-$ implies $|\rho_k| = 1$.

Conversely, $|\rho_k| = 1$ holds only when $\mathbf{h_k}$ and $\mathbf{h_{k_*}}$ are in an exact affine relation. Because both variables have range $\{0, \ldots, n\}$, the only possible relations are $\mathbf{h_k} = \mathbf{h_{k_*}}$ and $\mathbf{h_k} = n - \mathbf{h_{k_*}}$, corresponding respectively to $\delta \in S_+$ and $\delta \in S_-$.

It remains to show $S_+ = \{0\}$ for bijective g. Let $x_0 = g^{-1}(0)$. If $\delta \in S_+$, then $HW(g(x_0 \oplus \delta)) = HW(g(x_0)) = 0$, so $g(x_0 \oplus \delta) = g(x_0) = 0$. By injectivity, $\delta = 0$. Therefore $S_+ = \{0\}$, and the maximum is unique exactly when no nonzero element lies in S_-. □

Remark. The non-linear SBoxes evaluated in this work (AES, PRESENT, and ASCON) satisfy $S_- = \emptyset$, and therefore induce asymmetric leakage under the HW leakage model and uniform input distribution. This is a property of these specific SBoxes and assumptions; it should not be read as a theorem that all non-linear functions yield a unique maximum. The characterization above concerns global maxima in the ideal HW model.

Theorem 2. *Let $f(\mathbf{d}, \mathbf{k}) : GF(2)^n \times GF(2)^n \to GF(2)^n$ be a linear map of the form $f(\mathbf{d}, \mathbf{k}) = \mathbf{A}\mathbf{d} \oplus \mathbf{B}\mathbf{k}$, where $\mathbf{d}$ is uniformly distributed over $GF(2)^n$ and $\mathbf{A}$ is invertible. Let $\mathbf{1}$ denote the all-one vector in $GF(2)^n$, set $Im(\mathbf{B}) = \{\mathbf{B}\mathbf{x} | \mathbf{x} \in GF(2)^n\}$, and let $r_B = \mathrm{rank}(\mathbf{B})$. The key guesses attaining $|\rho_k| = 1$ are exactly*

$$\mathcal{M}_0 = \{k \mid \mathbf{B}(k \oplus k_*) = \mathbf{0}\}$$

and, if $\mathbf{1} \in Im(\mathbf{B})$,

$$\mathcal{M}_1 = \{k \mid \mathbf{B}(k \oplus k_*) = \mathbf{1}\}.$$

Consequently,

(1) *if $\mathbf{1} \notin Im(\mathbf{B})$, there are 2^{n-r_B} maximum points;*
(2) *if $\mathbf{1} \in Im(\mathbf{B})$, there are 2^{n-r_B+1} maximum points;*
(3) *in particular, when $\mathbf{B}$ is invertible, there are exactly two maximum points: k_* and $k_* \oplus \mathbf{B}^{-1}\mathbf{1}$.*

Proof. We use lowercase letters to denote the integer representation of binary vectors; for instance, a vector $\mathbf{d}$ is represented as a decimal integer $d \in \{0, 1, \ldots, 2^n - 1\}$. It can be observed from equation (4) that $|\rho_k| \leq 1$, and $k = k_*$ is a natural solution for $|\rho_k| = 1$. In the following, we investigate whether other solutions for $|\rho_k| = 1$ exist.

Let the key difference be $\delta = k \oplus k_*$ and the hypothetical intermediate-value difference be $\Delta = f(\mathbf{d}, \mathbf{k}) \oplus f(\mathbf{d}, \mathbf{k}_*) = \mathbf{B}\delta$. Denoting $\mathbf{s} = f(\mathbf{d}, \mathbf{k}_*) = \mathbf{A}\mathbf{d} \oplus \mathbf{B}\mathbf{k}_*$, we have $h_{k_*} = HW(\mathbf{s})$ and $h_k = HW(\mathbf{s} \oplus \Delta)$. Since $\mathbf{A}$ is invertible and $\mathbf{d}$ is uniformly distributed over $GF(2)^n$, $\mathbf{A}\mathbf{d}$ is uniform over $GF(2)^n$; adding the constant $\mathbf{B}\mathbf{k}_*$ preserves uniformity. Therefore, $\mathbf{s}$ is uniformly distributed on $GF(2)^n$ with mutually independent bits. That is, each bit s_i of $\mathbf{s}$ is an independent Bernoulli(1/2) random variable. Consequently, we obtain

$$E[h_k] = E[h_{k_*}] = \frac{n}{2}, \quad Var[h_k] = Var[h_{k_*}] = \frac{n}{4}.$$

Next, we analyze the covariance of $\mathbf{h_k}$ and $\mathbf{h_{k_*}}$, i.e., $Cov(\mathbf{h_k}, \mathbf{h_{k_*}}) = E[\mathbf{h_k}\mathbf{h_{k_*}}] - E[\mathbf{h_k}]E[\mathbf{h_{k_*}}]$. By expanding the Hamming weight, we have $\mathbf{h_{k_*}} = HW(\mathbf{s}) = \sum_{j=1}^{n} s_j$, $\mathbf{h_k} = HW(\mathbf{s} \oplus \Delta) = \sum_{i=1}^{n}(s_i \oplus \Delta_i)$. We can get

$$E[\mathbf{h_k}\mathbf{h_{k_*}}] = \sum_{i=1}^{n}\sum_{j=1}^{n} E[(s_i \oplus \Delta_i)s_j].$$

Following, we discuss the value of $E[(s_i \oplus \Delta_i)s_j]$ for different i and j:

1) When $i = j$: if $\Delta_i = 0$, then $E[(s_i \oplus 0)s_j] = E[s_i^2] = E[s_i] = \frac{1}{2}$; if $\Delta_i = 1$, then $E[(s_i \oplus 1)s_j] = E[(1 - s_i)s_i] = 0$.
2) When $i \neq j$: due to independence of s_i and s_j, $E[(s_i \oplus \Delta_i)s_j] = E[s_i \oplus \Delta_i]E[s_j] = \frac{1}{4}$.

Substituting the above expression into $E[\mathbf{h_k}\mathbf{h_{k_*}}]$ and setting $w = HW(\Delta)$, we get

$$\begin{aligned} E[\mathbf{h_k}\mathbf{h_{k_*}}] &= \sum_{i=j} E[(s_i \oplus \Delta_i)s_j] + \sum_{i \neq j} E[(s_i \oplus \Delta_i)s_j] \\ &= \frac{n-w}{2} + \frac{n(n-1)}{4} = \frac{n^2}{4} + \frac{n}{4} - \frac{w}{2}. \end{aligned}$$

Therefore, $Cov(\mathbf{h_k}, \mathbf{h_{k_*}}) = \frac{n}{4} - \frac{w}{2}$ and

$$\rho_k = \mathrm{Corr}(\mathbf{h_k}, \mathbf{h_{k_*}}) = 1 - \frac{2w}{n}. \tag{5}$$

Finally, we have $|\rho_k| = 1 \iff |1 - \frac{2w}{n}| = 1 \iff w \in \{0, n\} \iff \Delta \in \{\mathbf{0}, \mathbf{1}\}$. The case $\Delta = \mathbf{0}$ corresponds to a positive correlation maximum, while $\Delta = \mathbf{1}$ corresponds to a negative correlation minimum. The prior DL distinguisher merges these two cases because it operates on $|\rho_k|$.

It remains to count the solutions. The solutions to $\mathbf{B}(k \oplus k_*) = \mathbf{0}$ are exactly $k = k_* \oplus z$ for $z \in \ker(\mathbf{B})$, so $|\mathcal{M}_0| = 2^{n-r_B}$. The equation $\mathbf{B}(k \oplus k_*) = \mathbf{1}$ is solvable if and only if $\mathbf{1} \in Im(\mathbf{B})$. When it is solvable, its solution set is a coset of $\ker(\mathbf{B})$, so $|\mathcal{M}_1| = 2^{n-r_B}$. The two sets are disjoint because $\mathbf{0} \neq \mathbf{1}$. This proves the two counting cases. If $\mathbf{B}$ is invertible, then $\ker(\mathbf{B}) = \{\mathbf{0}\}$ and $\mathbf{1} \in Im(\mathbf{B})$, giving the two solutions k_* and $k_* \oplus \mathbf{B}^{-1}\mathbf{1}$. □

Theorem 3. *When $f(d, k) = d \oplus k$, $|\rho_k|$ attains its maximum value at two points, namely k_* and its bitwise complement.*

Proof. For $f(d, k) = d \oplus k$, we have $\mathbf{A} = \mathbf{B} = \mathbf{I}$. According to Theorem 2, the two solutions for $|\rho_k| = 1$ are k_* and $k_* \oplus \mathbf{1}$, i.e., the bitwise complement of k_*. □

Theoretical definitions of asymmetric and symmetric leakage. Definition 1 separates the leakage class from the syntactic form of the target function. Theorem 1 characterizes the standard bijective SBox setting under the HW model, while Theorem 2 analyzes the linear key-mixing setting. Together, they give the following leakage classes used in this work:

(1) When $|\rho_k|$ has a unique global maximum at $k = k_*$, the target induces **asymmetric leakage**. The non-linear SBoxes evaluated in this work fall into this class because their HW-complement set S_- is empty under the assumptions of Theorem 1. Certain linear operations may also induce asymmetric leakage, showing that asymmetry is a property of the leakage model and target distribution rather than of non-linearity alone.

(2) Conversely, when f is linear and $\mathbf{1} \in \mathrm{Im}(\mathbf{B})$, the number of global maxima for $|\rho_k|$ is 2^{n-r_B+1}, where $r_B = \mathrm{rank}(\mathbf{B})$. If $r_B < n$, the target identifies only an equivalence class of keys rather than a unique key or key pair, making it unfavorable for exact SCA key recovery.

(3) In the common invertible key-mixing case where f **is linear and $\mathbf{B}$ is invertible**, there are exactly two symmetric maxima located at k_* and $k_* \oplus \mathbf{B}^{-1}\mathbf{1}$. This phenomenon is defined as **symmetric leakage**.

Consequently, asymmetric and symmetric leakage are defined by the structure of the absolute-correlation maxima. Since the global maximum in equation (3) is non-unique under symmetric leakage, the prior distinguisher cannot uniquely identify k_* without an additional ambiguity-resolution mechanism.

Remark. Theorem 3 is consistent with the empirical observations reported in prior work [1], [12], [28], [29], where targeting an XOR output in traditional CPA was shown to produce two correlation extrema with opposite signs. The role of Theorem 2 is to place this known phenomenon in the DL-NSCA setting, where loss and accuracy behave as sign-insensitive metrics, and to state precisely how the ambiguity changes for general linear key-mixing matrices. Given that XOR is the most ubiquitous linear operation in block ciphers, this paper adopts it as the primary representative case for symmetric leakage analysis.

Remark. From a practical standpoint, this theoretical two-key result does not render the attack infeasible. For example in AES-128, when attacking 16 key bytes independently, each byte yields exactly two candidates, resulting in a residual search space of 2^{16}, which is computationally trivial.

IV. DL KEY DISTINGUISHER VS-GBA

To address the two issues identified in Section III, we propose a new DL key distinguisher, named Valley-Screened Gaussian Borda Aggregation (VS-GBA), as shown in Algorithm 1 and 2. It is designed for both symmetric and asymmetric leakage. The process consists of two main steps. First, based on leakage features, it filters the loss values to select discriminative KGVs, which is detailed in Section IV-A. Second, we employ a multi-voter rank aggregation method to derive the final key rank from the selected KGVs, as detailed in Section IV-B.

A. Step 1: screening epochs with discriminative KGVs

As established in Section III-A, our goal is to identify a set of valid epochs, $\mathcal{T}_{conv}$, from all training epochs that correspond to discriminative KGVs. The theoretical basis for what constitutes a *discriminative* KGV follows directly from Theorems 1–3. For asymmetric leakage, a discriminative KGV should exhibit a *unique* minimum; for symmetric leakage, it should exhibit *two symmetric* minima. We formalize this using a correlation vector, $\mathbf{C} = (|\rho_0|, |\rho_1|, \ldots, |\rho_{2^n-1}|)$, which we term the **theoretical KGV**.

For an asymmetric leakage framework, the theoretical KGV possesses a unique maximum, resulting in a single and distinct peak. Consequently, a discriminative KGV is expected to exhibit a unique minimum, the position of which indicates the correct key guess. For a symmetric leakage framework, according to Theorem 3, the theoretical KGV has two maxima, corresponding to the correct key guess and its bitwise complement. Thus, a discriminative KGV should feature two symmetric minimum points. While recovery by symmetric leakage is theoretically feasible using the prior distinguisher by identifying the candidate keys corresponding to the minimum and the second-smallest loss values, it comes at the cost that a greater number of traces is needed before the two symmetric loss trends become sufficiently distinct from those of all other key candidates. VS-GBA addresses this by automatically selecting epochs at which such discriminative two-minima structure is observable.

We assume that the global minimum of a theoretical KGV corresponds to the local minimum with the greatest valley prominence. Therefore, in both scenarios, the task of filtering for discriminative KGVs transforms into a problem of identifying local minima within maximum prominence. We employ the FIND_PEAKS algorithm, a standard method in signal processing introduced in Section II-D. We use this algorithm to first identify all local minima in a KGV and then apply a prominence-based condition to select the significant ones (1 for asymmetric leakage and 2 for symmetric leakage), thereby isolating the discriminative KGVs.

In the asymmetric leakage scenario, we iterate through the KGVs from all epochs. A KGV $\mathbf{l_t}$ is added to the set $\mathbf{L}_{\mathcal{T}_{conv}}$ if the prominence of its local minimum exceeds a prominence threshold τ_p. This procedure is detailed in lines 1-8 of Algorithm 1.

In the symmetric leakage scenario, in addition to the valley prominence, we introduce an additional symmetry condition. This condition requires that for the two key guesses with the highest valley prominence, the absolute difference between their sum and value $N-1$ must be less than a symmetry tolerance τ_s. Secondly, based on the symmetry property of the theoretical KGV, we divide the key space $\{0, 1, \ldots, N-1\}$

Algorithm 1 Valley-Screened Gaussian Borda Aggregation Algorithm for asymmetric leakage

Input: Key candidate space $\mathcal{K}$, KGVs for all epochs $\mathbf{L} = \{\mathbf{l_t}\}_{t=0}^{t_{max}-1}$, prominence threshold τ_p, decay coefficient α
Output: Final key ranking $\mathbf{r}$

1: $N \leftarrow len(\mathcal{K})$
2: $\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}} \leftarrow$ *empty list* ▷ Screening discriminative KGVs
3: **for** $t = 0$ **to** $t_{max} - 1$ **do**
4: $\quad\langle \mathbf{V}, \mathbf{P} \rangle \leftarrow$ FIND_PEAKS$(-\mathbf{l_t}, \tau_p)$
5: $\quad$**if** $|\mathbf{V}| > 0$ **then**
6: $\quad\quad$APPEND$(\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}}, \mathbf{l_t})$
7: $\quad$**end if**
8: **end for**
9: $\mathbf{R}[j, c] \leftarrow$ ARGSORT(ARGSORT$(\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}}[j]))[c]$ ▷ rank of candidate c in epoch j
10: $\mathbf{s}[c] \leftarrow \sum_j e^{-\alpha\, \mathbf{R}[j,c]^2}$
11: $\mathbf{r} \leftarrow$ ARGSORT$(-\mathbf{s})$
12: **return** $\mathbf{r}$

into two halves around the symmetric axis $\frac{N-1}{2}$: $\mathcal{K}_l = \{0, 1, \ldots, \frac{N}{2}-1\}$ and $\mathcal{K}_r = \{\frac{N}{2}, \frac{N}{2}+1, \ldots, N-1\}$. For any key guess k_2 in the set $\mathcal{K}_r$, there exists a corresponding key $k_1 = (N-1) \oplus k_2$ in the set $\mathcal{K}_l$ such that $|\rho_{k_1}| = |\rho_{k_2}|$. Therefore, we fold the KGV from $\mathcal{K}_r$ to $\mathcal{K}_l$ in the same way, and the two minimum points in the folded KGVs can be interpreted as two predictions for the correct key pair $(k_*, (N-1) \oplus k_*)$. This process is outlined in lines 1-14 of Algorithm 2.

B. Step 2: Key rank aggregation using a voting-based method

Regardless of whether the underlying leakage is symmetric or asymmetric, the procedure in Step 1 yields predicted rankings of the correct key (or key pair) across multiple epochs. This outcome allows us to frame the problem as one of ranking aggregation within a multi-voter system. In this model, we treat each discriminative epoch as a "voter" and each potential key as a "candidate". The core task is to aggregate the individual rankings provided by all voters to produce a final, consolidated rank for the key guesses. In this paper, we utilize a modified Borda count method to determine the optimal ordering of a set of candidates based on their loss values across multiple valid epochs.

Firstly, we assign a rank based on the ascending order of loss values for each epoch, thereby constructing the ranking matrix $\mathbf{R}$ that contains the ordered preference of candidates for each epoch, as described in lines 9 of Algorithm 1 and lines 15 of Algorithm 2. Subsequently, the algorithm calculates an aggregate score for each candidate. For a candidate's rank i in a given epoch, a score is awarded according to the formula $e^{-\alpha i^2}$, and this step is described in lines 10-11 of Algorithm 1 and lines 16-17 of Algorithm 2.

Unlike the classical Borda count algorithm[2], which assigns scores that decrease linearly with rank, we employ a Gaussian function, $g(i) = e^{-\alpha i^2}$, where α is the decay coefficient. This approach assigns significantly higher weight to top-ranked positions, thereby avoiding the shortcoming of the Borda count algorithm, which violates the Majority Criterion [39]. Finally, these scores are summed for each candidate across all epochs to produce a final cumulative score. We then sort these cumulative scores in descending order to obtain the final key ranking.

Algorithm 2 Valley-Screened Gaussian Borda Aggregation Algorithm for symmetric leakage

Input: Key candidate space $\mathcal{K}$, KGVs for all epochs $\mathbf{L} = \{\mathbf{l_t}\}_{t=0}^{t_{max}-1}$, prominence threshold τ_p, symmetry tolerance τ_s, decay coefficient α
Output: Final key ranking $\mathbf{r}$

1: $N \leftarrow len(\mathcal{K})$
2: $\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}} \leftarrow$ *empty list* ▷ Screening for discriminative KGVs
3: **for** $t = 0$ **to** $t_{max} - 1$ **do**
4: $\quad\langle \mathbf{V}, \mathbf{P} \rangle \leftarrow$ FIND_PEAKS$(-\mathbf{l_t}, \tau_p)$
5: $\quad$**if** $|\mathbf{V}| > 1$ **then**
6: $\quad\quad j_1, j_2 \leftarrow$ TOP2INDICES$(\mathbf{P})$ ▷ Prominence condition
7: $\quad\quad k_1, k_2 \leftarrow \mathbf{V}[j_1], \mathbf{V}[j_2]$ ▷ Key guesses of top-2 prominence
8: $\quad\quad SymErr \leftarrow |k_1 + k_2 - (N-1)|$ ▷ Symmetry condition
9: $\quad\quad$**if** $SymErr \leq \tau_s$ **then**
10: $\quad\quad\quad$APPEND$(\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}}, \mathbf{l_t}[0 : \frac{N}{2}))$
11: $\quad\quad\quad$APPEND$(\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}}, Reverse(\mathbf{l_t}[\frac{N}{2} : N)))$
12: $\quad\quad$**end if**
13: $\quad$**end if**
14: **end for**
15: $\mathbf{R}[j, c] \leftarrow$ ARGSORT(ARGSORT$(\mathbf{L}_{\mathcal{T}_{\mathbf{conv}}}[j]))[c]$ ▷ rank of candidate c in epoch j
16: $\mathbf{s}[c] \leftarrow \sum_j e^{-\alpha \mathbf{R}[j,c]^2}$
17: $\mathbf{r} \leftarrow$ ARGSORT$(-\mathbf{s})$
18: **return** $\mathbf{r}$

V. Experimental results

In this section, we present a comprehensive experimental evaluation of the proposed framework. Section V-A describes the experimental setup and evaluation criteria. In Section V-B, we validate the correctness of our theoretical analysis using public benchmark datasets. Section V-C conducts cross-algorithm experiments on three classes of standard cryptographic algorithms, demonstrating the advantages of the symmetric leakage analysis framework on real devices and in implementations where nonlinear operations are protected while the evaluated linear key-mixing target remains exposed. Finally, Section V-D shows that the proposed framework is capable of recovering the secret key of tweakable block ciphers where plaintext/ciphertext is masked. We provide our reproduction code[3].

A. Experimental setup and metrics

We employ GE as the evaluation metric, measured at intervals of 500 traces to determine the minimum trace count required. For each trace set, experiments are repeated 10 times to compute the average key rank, mitigating variability from random weight initialization and other factors. If the average key rank reaches zero across all runs, the corresponding trace number is recorded as $NTGE$; otherwise, if recovery fails within the maximum 50,000 traces, model performance is evaluated using the GE value at 50,000 traces.

For asymmetric leakage, GE denotes exact-key GE. For symmetric leakage, the folded VS-GBA ranking represents the equivalence class $\{k, k \oplus \mathbf{1}\}$, so the reported $NTGE$ denotes candidate-pair recovery. This metric is not directly identical to exact-key recovery: exact recovery requires either a residual search over the per-byte candidate pairs or an additional disambiguation source. When comparing with the prior distinguisher on symmetric leakage, a full-ranking exact-key result of rank 0 or rank 1 is therefore interpreted as the same candidate-pair recovery event.

In our experiments, the VS-GBA key distinguisher's hyper-parameters are fixed to verify its generality. The prominence

[2]It is detailed in Section II-E.

[3]https://github.com/jintongyucrypto/symmetric_leakage__DLNSCA.git

TABLE I
ATTACK PERFORMANCE UNDER DIFFERENT KEY DISTINGUISHER UNDER ASYMMETRIC AND SYMMETRIC LEAKAGE SCENARIOS

Attack points	Key distinguisher	VS-GBA	Prior key distinguisher					
	Epoch		0	10	20	30	40	50
asymmetric leakage	ASCAD_F_R	10000	$GE = 47$	11500	18500	29500	36000	38000
	ASCAD_V_R	6000	$GE = 35$	7000	12000	12000	15000	19500
symmetric leakage	ASCAD_F_R	25500	$GE = 12$	49000	42000	$GE = 238$	$GE = 105$	$GE = 12$
	ASCAD_V_R	35000	$GE = 96$	$GE = 8$	49500	$GE = 236$	$GE = 160$	$GE = 96$

threshold is $\tau_p = 0.1$ for AVR devices and $\tau_p = 0.001$ for ARM devices, the symmetry tolerance is $\tau_s = 10$, and the decay coefficient is set to $\alpha = 1$. These values are kept unchanged within each device class and are not tuned per byte or per key guess. All networks were implemented using the PyTorch framework with an AMD EPYC 7763 64-Core Processor, an NVIDIA GeForce RTX 4090 GPU, and 1 TB of RAM.

B. Validation of theoretical analysis and VS-GBA

In the field of artificial intelligence, the performance of neural networks is evaluated by DL metrics, such as loss and accuracy. The principle of the prior distinguisher relies on this **simple correspondence**: the key guess corresponding to the model with the minimum loss value is regarded as the correct key. As discussed in Section III-A, the drawback of this approach is that the efficacy of the prior distinguisher exhibits a strong dependence on the selected training epochs. For symmetric leakage, it is further limited by the non-uniqueness of the correlation maxima, as established in Section III-B.

Therefore, when performing DL-NSCA, it is critical to investigate how to leverage the data provided by DL metrics and interpret them as SCA information, where building this communication bridge is essential. We observe that the prior distinguisher fails to adequately perform this interpretation task, meaning that the attack outcome remains concealed in the DL metrics after neural network training. To address this issue, we propose the novel VS-GBA distinguisher, which extracts SCA information from DL metrics under the **guidance of the theoretical KGV distribution**, as illustrated in Figure 2. Unlike the prior distinguisher, VS-GBA does not require manually selecting a single epoch t_0; instead, it screens multiple checkpoints and aggregates the resulting rankings. In the evaluated settings, the selected KGVs are empirically closer to the expected theoretical KGV structure.

In the following part, we provide empirical validation of these findings using public datasets.

Settings. We analyze the asymmetric and symmetric leakage of the ASCAD_F_R and ASCAD_V_R datasets. We employ the state-of-the-art model, ConvWIN-MCR, and use the same parameters as the original authors on both datasets [19]. The target intermediate values are $f_s(d,k) = p[2] \oplus k[2]$ and $f_{as}(d,k) = \text{SubBytes}[p[2] \oplus k[2]]$ respectively, where $p[2]$ is the third byte of the plaintext and $k[2]$ is the third byte of the key.

Results under symmetric leakage. Using the prior distinguisher to analyze symmetric leakage, we find that the exact-key GE value fluctuates between 0 and 1, which is consistent with the analysis in Section III-A. This is because the theoretical KGV yields maximum absolute correlation for two indistinguishable key guesses. Therefore, Table I should be read as a candidate-pair comparison for symmetric leakage: for the prior distinguisher, exact-key ranks 0 and 1 correspond to the same recovered candidate pair, while VS-GBA reports the folded candidate-pair rank. Under this metric, VS-GBA outperforms the prior distinguisher for most epochs.

We next further analyze how discriminative KGVs selected by the VS-GBA method approach the theoretical KGV under the symmetric leakage scenario. We perform attacks on the ASCAD_F_R dataset to recover the third byte key 224. First, we compute the theoretical KGV. From Equation (5), we have $|\rho_k| = |1 - \frac{2HW[f(d,k)\oplus f(d,k_*)]}{n}| = |1 - \frac{HW[k\oplus k_*]}{4}|$, as shown in the right subplot of Figure 3. According to results on DL metric, we then adopt the VS-GBA method to select discriminative KGVs. The left subplot in Figure 3 shows the loss values of a discriminative KGV across all key guesses. Specifically, the two highest points of the theoretical KGV (with maximum correlation) correspond to the lowest points of the discriminative KGV (with fastest convergence),which validates that the proposed VS-GBA well approximates and approaches the theoretical optimum. Furthermore, both the theoretical KGV and the discriminative KGV are symmetric about the middle key guesses, which further supports our conclusion in Theorem 2 and 3.

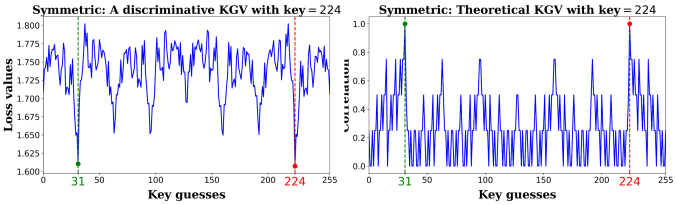

Fig. 3. Visualization of results for VS-GBA on symmetric leakage

Results under asymmetric leakage. The analysis results for the asymmetric leakage are presented in Table I. It is observed that the performance of the prior key distinguisher varies significantly across different training epochs, yet their best results are comparable to those of the proposed VS-GBA. We next examine how discriminative KGVs selected by the VS-GBA method approach the theoretical KGV under the asymmetric leakage scenario. For the recovery of the third-byte key 224 on the ASCAD_F_R dataset, we calculate the

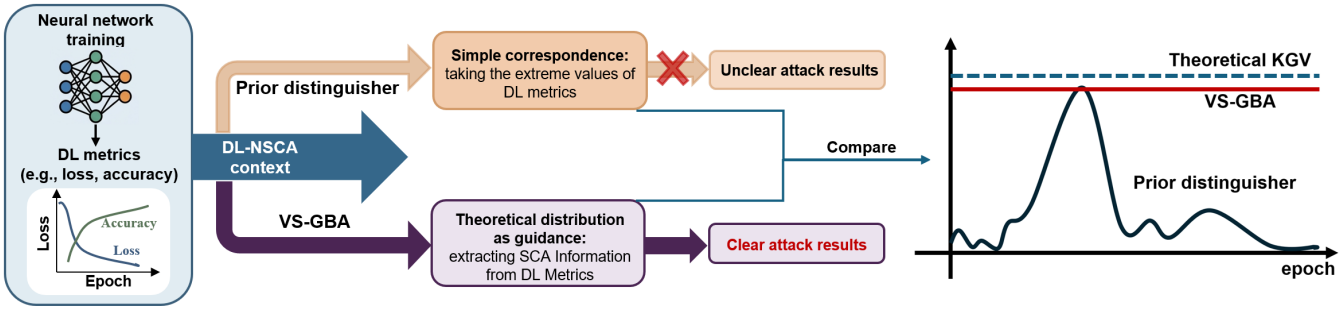

Fig. 2. VS-GBA: Building a bridge from DL metrics to SCA information

theoretical KGV. Based on Equation (5), it follows that $|\rho_k| = |\frac{HW[Sbox[\bar{d}\oplus k]\oplus Sbox[\bar{d}\oplus k_*]]}{4} - 1|$[4], and the results are presented in the right subplot of Figure 4. Subsequently, we conduct DL-NSCA analysis on the input traces, and a discriminative KGV filtered by the VS-GBA is displayed in the leftmost part of Figure 4. It can be seen that the theoretical KGV and the discriminative KGV exhibit analogous distributions: the highest point of the theoretical KGV (corresponding to the maximum correlation under the stated model) aligns with the lowest point of the discriminative KGV (corresponding to the fastest convergence). This indicates that VS-GBA has successfully extracted the SCA information transmitted by the neural network in this experiment.

C. Cross-algorithm evaluation

Section V-B demonstrates that the VS-GBA distinguisher successfully builds a bridge from DL metrics to SCA information. In this section, we explore how to perform cross-algorithm analysis under symmetric leakage. Our analysis typically targets symmetric leakage occurring at the beginning or end of encryption algorithms, which is common in symmetric cryptography.

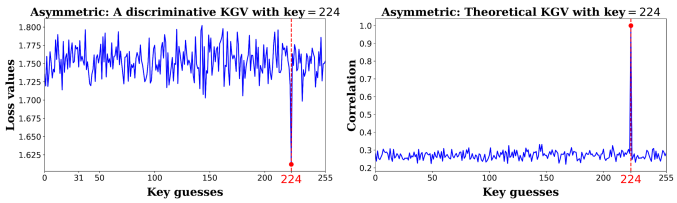

Fig. 4. Visualization of results for VS-GBA on asymmetric leakage

To demonstrate practical feasibility, we experimentally analyze three standard cryptographic algorithms on real devices and high-noise traces, including traces from masked implementations. First, we observe that multiple standard algorithms share the same symmetric leakage pattern $f_s(d,k) = d \oplus k$. This work takes three algorithms as examples: AES, a symmetric block cipher widely adopted as a global encryption standard; ASCON, the winner of the CAESAR competition and NIST's recommended lightweight cipher for resource-constrained environments; and PRESENT, a lightweight block cipher specifically designed for low-power and compact hardware implementations. Second, while prior studies primarily focused on simulations or 8-bit AVR microcontrollers [15]–[18], this work transitions to the 32-bit ARM Cortex-M4 architecture to better reflect real-world scenarios. Third, although DL-NSCA methods have shown strong performance on masked implementations, their application has been mainly limited to Boolean-masked AES, as used in the public ASCADv1 dataset. Our work targets the more challenging masked implementations of AES, and further extends to masked implementations of ASCON and PRESENT.

Notably, for AES, a more challenging public benchmark, ASCADv2, employs a combination of affine masking and shuffling countermeasures. Existing work has shown that even in the profiled setting, where the attacker has access to a fully controlled clone device and knows the values of affine masks during profiling, attacking ASCADv2 requires either a 27-layer multi-task deep learning architecture [40] or large language models [41]. In the non-profiled setting, these challenges are substantially compounded: the absence of a profiling device increases the learning difficulty, and the limited capacity of practically deployable neural networks poses a bottleneck [19]. Consequently, non-profiled analysis of ASCADv2 remains an open problem, and affine masking, higher-order masking, and shuffling countermeasures are beyond the scope of this work. Instead, we focus on implementations protected by Rivain-Prouff masking [42] and evaluate whether attacking exposed linear targets can avoid the protected nonlinear lookup or permutation operations in the tested setting.

1) AES: Standardized by NIST in 2001, AES is a symmetric block cipher based on a Substitution-Permutation Network (SPN). This study utilizes AES-128 as the primary benchmark for analysis.

Attack points. For each key byte K, the Round 0 AddRoundKey operation yields $A = P \oplus K$, where P is the corresponding plaintext byte. This value forms the input to the first-round SubBytes transformation, as illustrated in Figure 5. For asymmetric leakage, we target the first-round SBox output, a primary target of SCA works [5]–[7], [16]–[18], [18], defined as $f_{as}(P,K) = SubBytes[P \oplus K]$. For symmetric leakage, we focus on the direct output of AddRoundKey, given by

[4]As stated in Theorem 1, the theoretical KGV possesses a single unique maximum when the volume of output data is sufficient. Consequently, we utilize 500 plaintext samples and calculate the average value of input data $\bar{d}$.

$f_s(P, K) = P \oplus K$.

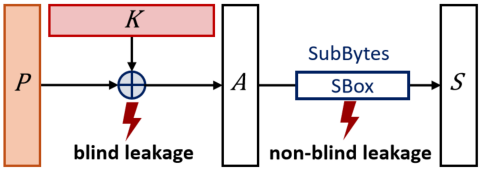

Fig. 5. The two initial operations of the AES algorithm. We employ a color-coding scheme to show different elements in $f(d, k)$: red denotes the unknown secret key K, blue blocks represent the function f itself, and orange blocks signify the known data involved in the calculation.

Datasets. Existing methods can successfully recover keys from the ASCADv1 dataset, mainly due to the low-noise 8-bit AVR ATMega platform and simple first-order Boolean masking implementation. Neural networks typically learn secret values by automatically recombining masked shares [43]. Instead, we evaluate the more secure Rivain-Prouff masking [42] on a 32-bit ARM Cortex-M4 platform. This scheme protects nonlinear SBox operations by a dedicated field-multiplication gadget (SecMult), creating a challenging task comparable to masking countermeasures in ASCADv2 and extending DL-NSCA beyond Boolean masking. Our implementation uses a public first-order version of the scheme[5].

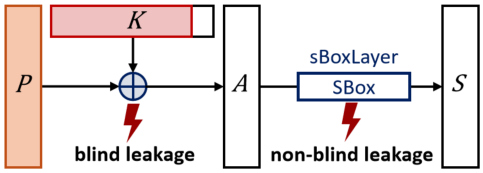

Fig. 6. The two initial operations of the PRESENT algorithm.

Key recovery. Experimental results are summarized in Table II. As shown in Section V-B, the choice of distinguisher strongly affects key recovery in DL-NSCA, which consists of a neural network and a distinguisher. Since this work does not focus on network architecture, we use the same network as ConvWIN-MCR[6], differing only in the distinguisher. Under symmetric leakage, our method outputs key-candidate pairs with few traces, whereas analysis fails for the evaluated asymmetric leakage target, where the correct key is poorly ranked. Notably, substantial progress has been made in securing AES nonlinear operations [14], [37], [44]–[47], greatly increasing SCA difficulty.

2) PRESENT: Standardized by ISO/IEC in 2012 as a lightweight cryptographic benchmark, PRESENT was specifically engineered for hardware-constrained environments such as RFID tags and sensor networks [48]. It features a compact 64-bit block size and supports 80 or 128-bit keys, utilizing a bit-oriented permutation and a 4-bit SBox for high efficiency. This study focuses on PRESENT-80 as a representative target.

Attack Points. Focusing on each byte K within the initial 64 bits of the key, the first-round operations begin with AddRoundKey, where the plaintext P is XORed with the key K to produce the intermediate state A. Subsequently, the sBoxLayer transformation is applied to A, resulting in $S = \text{sBoxLayer}[A]$. As illustrated in Figure 6, we define the target intermediate values for asymmetric and symmetric leakage as $f_{\text{as}}(P, K) = \text{sBoxLayer}[P \oplus K]$ and $f_{\text{s}}(P, K) = P \oplus K$, respectively.

Datasets. We implemented two versions of the PRESENT algorithm on the ARM Cortex-M4 platform: an unprotected version and a first-order Boolean masked version. For the unprotected implementation, we utilized a public library[7] to generate the PRESENT dataset, which corresponds to the side-channel traces captured at the locations of symmetric and asymmetric leakage, respectively. Regarding the masked implementation, given the scarcity of public code specifically for masked PRESENT, we adapted a well-documented first-order Boolean masked AES library[8] to ensure experimental reproducibility. This approach is justified by the shared SPN architecture of both ciphers, and obtains the PRESENT_masked dataset.

Key Recovery. The experimental results are summarized in Table II. For the unprotected implementation, our method requires fewer traces regardless of whether symmetric or asymmetric leakage is considered. It can be observed that under the unprotected scenario, asymmetric leakage exhibits stronger intensity, thus requiring slightly fewer traces for a successful attack. However, in traces from the first-order masked implementation, attacking the exposed linear key-mixing target makes the advantage of symmetric leakage evident. Although asymmetric leakage is marginally stronger in intensity than symmetric leakage, this difference is insufficient to overcome the confounding effects introduced by high-noise environments and the protected nonlinear layer.

3) ASCON: ASCON was selected as the winner of both the CAESAR competition and the NIST Lightweight Cryptography standardization process, and is currently undergoing standardization for widespread deployment [24]. As a sponge-based authenticated encryption scheme, it guarantees message confidentiality and ensures integrity by generating an authentication tag over the ciphertexts and associated data. ASCON takes a key K, a nonce N, an associated data A, and a plaintext P as inputs to generate the authenticated ciphertext C and authentication tag T. The internal logic progresses through four phases: initialization, associated data processing, plaintext processing , and finalization.

Attack points. In the initialization phase, the 320-bit ASCON state $\{X_0, \ldots, X_4\}$ is composed of an initial constant IV_0, a 128-bit secret key $K = K_1||K_2$ and a 128-bit nonce, and a 128-bit unique nonce $M = M_0||M_1$. Its permutation includes round constant addition, column-wise 5-bit SBox substitution and linear diffusion, as shown on the left of Figure 7. In the finalization phase, the tag T is derived by XORing the key K with the last 128 bits of the state (right of Figure 7). As the final step of ASCON, this leakage point locates at the end of side-channel traces and can be easily identified

[5]https://github.com/coron/htable/tree/master

[6]The ConvWIN-MCR network has only one hyperparameter to be searched, where we made grid search for the pooling stride over $[5, 10, 20, 50, 100, 200, 500, 1000, 2000]$.

[7]https://github.com/michaelkitson/Present-8bit/tree/master

[8]https://github.com/Secure-Embedded-Systems/Masked-AES-Implementation/tree/master/Byte-Masked-AES

TABLE II
PERFORMANCE COMPARISON ACROSS DIFFERENT ALGORITHMS UNDER HIGH-NOISE CONDITIONS AND MASKING COUNTERMEASURES.

Datasets	Attack points	DDLA [15]	ConvWIN-MCR [19]	this work
AES_rp_masked	asymmetric leakage symmetric leakage	GE=236 GE=23	GE=205 24500	GE=70 8000
PRESENT	asymmetric leakage symmetric leakage	45500 48500	3500 5500	3000 4500
PRESENT_masked	asymmetric leakage symmetric leakage	GE=210 GE=138	GE=90 9500	GE=27 8500
ASCON	asymmetric leakage symmetric leakage	GE=195 GE=5	GE=122 39500	GE=46 3000
ASCON_masked	asymmetric leakage symmetric leakage	GE=211 GE=39	GE=196 GE=3	GE=66 16000

without prior implementation details. For asymmetric leakage, we target the first-round permutation SBox output [49], [50]. Y_4 is computed as $Y_4 = K_1 \& (255 \oplus IV_0 \oplus M_0) \oplus M_0 \oplus M_1$, with the target intermediate value $f_{as}(IV_0, m_0, m_1, k) = k \& (255 \oplus IV_0 \oplus m_0) \oplus m_0 \oplus m_1$, recovered in 8-bit chunks. For symmetric leakage, we attack the last XOR operation in the tag generation phase; notably, this attack point has not been previously investigated. The target intermediate value is $f_s(T, k) = T \oplus k$.

Datasets. Firstly, we execute an unprotected implementation of ASCON-128. This implementation is derived from the 8-bit optimized version of the official ASCON reference code[9], which is designed for efficient execution on eight-bit microprocessors while maintaining compatibility with 32-bit architectures. The dataset ASCON consists of traces corresponding to symmetric leakage and asymmetric leakage. Furthermore, we consider the masked implementation scenario, which is based on the public masked implementation scheme of ASCON[10]. The dataset ASCON_masked contains traces from the masked implementation, where we extract the end part of traces for a protected asymmetric target in the permutation and for the exposed final-XOR target in tag generation.

Key recovery. The experimental results are detailed in Table II. For the unprotected implementation, recovering the secret key from asymmetric leakage in the initial phase of ASCON remains non-trivial under low-noise conditions, whereas symmetric leakage during the tag generation phase achieves candidate-pair recovery. In traces from the protected implementation, the exposed final-XOR target recovers the corresponding candidate classes, whereas attacking the protected asymmetric target fails to do so. This supports the use of exposed symmetric leakage as an avoidance target for the protected nonlinear computation in the evaluated setting.

[9]https://github.com/ascon/ascon-c
[10]https://github.com/ascon/simpleserial-ascon

D. Key recovery in masked I/O scenarios

While the cross-algorithm experiments in the preceding section demonstrate candidate-pair recovery from public plaintext/ciphertext-dependent targets, they rely on the assumption that the attacker has direct access to plaintext/ciphertext of the cryptographic primitive. Although this is a foundational premise in SCA literature, real-world implementations often obfuscate these values with random variables, resulting in what we define as masked plaintext/ciphertext. A prime example of this is found in disk encryption, a critical mechanism for protecting data at rest on personal computers and servers. To address the unique security and performance requirements of block-oriented storage, NIST released SP 800-38E in 2010, formally recommending the XTS-AES mode of operation as a standardized and secure solution [51].

Unlike traditional modes like CBC or CTR, XTS-AES operates on arbitrary-sized data without requiring padding, leveraging a tweak value T derived from the data's logical block address to ensure that the same plaintext in different blocks produces distinct ciphertexts, eliminating the risk of pattern leakage. The encryption procedure of XTS-AES is illustrated in the right part of Figure 8. It uses two AES keys: K_1 for the core encryption and K_2 for generating the tweak.

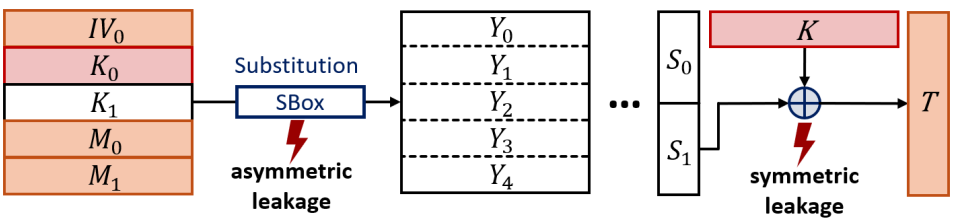

Fig. 7. Left: ASCON Substitution Layer. The SBox operation processes a 5-bit input formed by taking one bit from each word. It then produces a 5-bit output, where each bit corresponds to a word Y_i . Right: Tag generation step in finalization phase. The tag T derives from the last 128-bit state S_1 XORed with 128-bit key K.

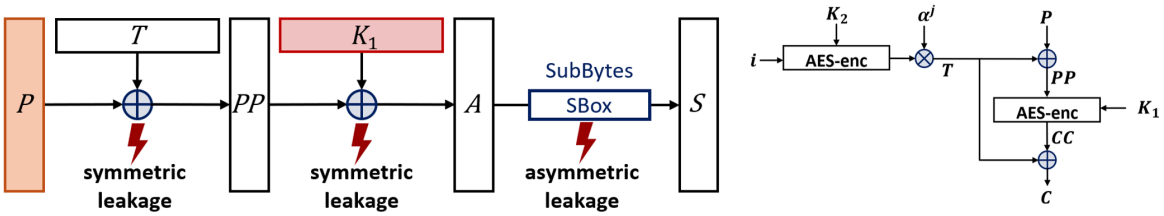

Fig. 8. Left:The first three primary operations of the XTS-AES algorithm. Right: The encryption procedure of XTS-AES.

We demonstrate that an attacker can collect multiple side-channel traces under a constant tweak value T to obtain the corresponding masked plaintext/ciphertext[11]. For the purposes of this study, we utilize AES-128 as the underlying primitive, where both K_1 and K_2 are 128 bits in length.

Attack points. Given that the attacker only possesses masked plaintexts, the attack on XTS-AES is partitioned into two independent stages, as illustrated in Figure 8. Following the algorithmic execution order, we define these stages as follows:

- **Phase 1: Pre-AES operations.** This phase involves the operations occurring before the core AES encryption. Here, the masked input P is XORed with the tweak value T to produce the actual AES input PP. This stage yields symmetric leakage, where the target intermediate value is defined as: $f_s(P,T) = P \oplus T$.
- **Phase 2: AES encryption.** This phase encompasses the internal AES transformation. Consistent with the attack points identified in Section V-C1, we consider both symmetric and asymmetric leakage. By substituting $PP = P \oplus T$, the target functions can be expressed using public data P and the effective secret $T \oplus K_1$: $f_s(PP, K_1) = PP \oplus K_1 = P \oplus (T \oplus K_1) = f_s(P, T \oplus K_1)$ and $f_{as}(PP, K_1) = \text{SubBytes}[PP \oplus K_1] = \text{SubBytes}[P \oplus (T \oplus K_1)] = f_{as}(P, T \oplus K_1)$.

Datasets. The XTS-AES algorithm was implemented on an ARM Cortex-M4 platform utilizing a public library[12] to generate our XTS-AES dataset. The captured power traces encompass the entire execution window, from the initiation of the XTS-AES mode to the SubBytes operation in the first round of the internal AES engine. Since these traces contain all three targeted feature points, the boundary between Phase 1 and Phase 2 can be located using engineering experience about the execution timeline or standard leakage-localization methods. Each trace consists of 9000 samples; for the purpose of evaluating our attack, we partitioned the first 5000 samples into Phase 1 and the subsequent 4000 samples into Phase 2.

Key recovery. The experimental results are detailed in Table III, where our method recovers the secret key using the fewest traces. Furthermore, while the specific leakage source in Phase 2 differs, since only symmetric leakage is present in Phase 1, the successful exploitation of symmetric leakage remains the fundamental component that enables secret key recovery for XTS-AES.

[11]The first phase involves encrypting the Logical Block Address (LBA), represented by index i. The LBA serves as a critical component of the tweak mechanism, ensuring that identical data stored at different physical locations results in unique ciphertexts, thereby maintaining the confidentiality of data at rest. In the context of SCA, adversaries typically target leakages at fixed locations; thus, it is assumed that the attacker can acquire multiple traces for different plaintexts under constant i and j values.

[12]https://github.com/heisencoder/XTS-AES/tree/master

TABLE III
PERFORMANCE COMPARISON ON XTS-AES UNDER HIGH-NOISE CONDITIONS.

Datasets	Attack points	DDLA [15]	ConvWIN-MCR [19]	this work
Phase 1	symmetric leakage	GE=8	16500	9500
Phase 2	asymmetric leakage	49500	13000	6500
	symmetric leakage	26500	4000	2500

VI. CONCLUSION

In summary, this work revisits DL-NSCA distinguishers from the viewpoint of absolute-correlation maxima. We show that the single-minimum assumption behind the prior minimum-loss distinguisher is not valid for symmetric leakage, formalize the resulting ambiguity for linear key-mixing targets, and propose VS-GBA to screen discriminative checkpoints and aggregate their rankings. Experiments on masked AES, PRESENT, ASCON, and XTS-AES show that symmetric leakage can provide a useful candidate-pair recovery target under the stated assumptions, especially when the nonlinear layer is protected but an exposed linear target remains available. The remaining challenges are to add stronger baselines, automate leakage-window selection, and evaluate whether the approach still works when linear key-mixing operations themselves are protected by affine masking, higher-order masking, or shuffling.

REFERENCES

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19*. Springer, 1999, pp. 388–397.

[2] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *International workshop on cryptographic hardware and embedded systems*. Springer, 2002, pp. 13–28.

[3] T. Bartkewitz and K. Lemke-Rust, "Efficient template attacks based on probabilistic multi-class support vector machines," in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2012, pp. 263–276.

[4] S. Picek, A. Heuser, and S. Guilley, "Template attack versus bayes classifier," *Journal of Cryptographic Engineering*, vol. 7, no. 4, pp. 343–351, 2017.

[5] L. Wu and S. Picek, "Remove some noise: On pre-processing of side-channel measurements with autoencoders," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 389–415, 2020.

[6] L. Wu, G. Perin, and S. Picek, "I choose you: Automated hyperparameter tuning for deep learning-based side-channel analysis," *IEEE Transactions on Emerging Topics in Computing*, vol. 12, no. 2, pp. 546–557, 2022.

[7] G. Perin, L. Wu, and S. Picek, "Exploring feature selection scenarios for deep learning-based side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 828–861, 2022.

[8] S. Picek, A. Heuser, A. Jovic, S. Bhasin, and F. Regazzoni, "The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 209–237, 2019.

[9] S. Picek, I. P. Samiotis, J. Kim, A. Heuser, S. Bhasin, and A. Legay, "On the performance of convolutional neural networks for side-channel analysis," in *International Conference on Security, Privacy, and Applied Cryptography Engineering*. Springer, 2018, pp. 157–176.

[10] J. Liu, Y. Yu, F.-X. Standaert, Z. Guo, D. Gu, W. Sun, Y. Ge, and X. Xie, "Small tweaks do not help: Differential power analysis of milenage implementations in 3g/4g usim cards," in *European Symposium on Research in Computer Security*. Springer, 2015, pp. 468–480.

[11] Tihmstar, "Using a magic wand to break the iphone's last security barrier," 2022, https://hardwear.io/netherlands-2022/speakers/tihmstar.php.

[12] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems-CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings 6*. Springer, 2004, pp. 16–29.

[13] N. Veyrat-Charvillon and F.-X. Standaert, "Mutual information analysis: how, when and why?" in *International workshop on cryptographic hardware and embedded systems*. Springer, 2009, pp. 429–443.

[14] M.-L. Akkar and C. Giraud, "An implementation of des and aes, secure against some attacks," in *Cryptographic Hardware and Embedded Systems—CHES 2001: Third International Workshop Paris, France, May 14–16, 2001 Proceedings 3*. Springer, 2001, pp. 309–318.

[15] B. Timon, "Non-profiled deep learning-based side-channel attacks with sensitivity analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 107–131, 2019.

[16] D. Kwon, S. Hong, and H. Kim, "Optimizing implementations of non-profiled deep learning-based side-channel attacks," *IEEE Access*, vol. 10, pp. 5957–5967, 2022.

[17] N.-T. Do, V.-P. Hoang, and V. S. Doan, "A novel non-profiled side channel attack based on multi-output regression neural network," *Journal of Cryptographic Engineering*, vol. 14, no. 3, pp. 427–439, 2024.

[18] I. Savu, M. Krček, G. Perin, L. Wu, and S. Picek, "The need for more: unsupervised side-channel analysis with single network training and multi-output regression," in *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 2024, pp. 113–132.

[19] J. Yu, Y. Wang, S. Qu, Y. Zhao, Y. Shi, P. Cao, X. Lu, C. Zhang, and D. Gu, "End-to-end non-profiled side-channel analysis on long raw traces," in *European symposium on research in computer security*. Springer, 2025.

[20] A. Zankl, H. Seuschek, G. Irazoqui, and B. Gulmezoglu, "Side-channel attacks in the internet of things: threats and challenges," in *Solutions for Cyber-Physical Systems Ubiquity*. IGI Global, 2018, pp. 325–357.

[21] M. S. Matúš Nemec, "Roca vulnerability," 2017, https://en.wikipedia.org/wiki/ROCA_vulnerability.

[22] I. Corporation, "Ibm 4769-001 cryptographic coprocessor security module non-proprietary security policy," https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp4079.pdf.

[23] M. Gao, Y. Liu, Y. Chen, Y. Li, Z. Ba, X. Xu, and J. Han, "Inertiear: Automatic and device-independent imu-based eavesdropping on smartphones," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1129–1138.

[24] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2: Lightweight authenticated encryption and hashing," *Journal of Cryptology*, vol. 34, no. 3, p. 33, 2021.

[25] M. Liskov, R. L. Rivest, and D. Wagner, "Tweakable block ciphers," in *Annual International Cryptology Conference*. Springer, 2002, pp. 31–46.

[26] T. Krovetz and P. Rogaway, "The ocb authenticated-encryption algorithm," Tech. Rep., 2014.

[27] N. Mouha and M. Dworkin, *Report on the block cipher modes of operation in the NIST SP 800-38 series*. US Department of Commerce, National Institute of Standards and Technology, 2024.

[28] K. Lemke, K. Schramm, and C. Paar, "Dpa on n-bit sized boolean and arithmetic operations and its application to idea, rc6, and the hmac-construction," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2004, pp. 205–219.

[29] A. Adomnicai, J. J. Fournier, and L. Masson, "Bricklayer attack: A side-channel analysis on the chacha quarter round," in *International Conference on Cryptology in India*. Springer, 2017, pp. 65–84.

[30] B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel, "Mutual information analysis: A generic side-channel distinguisher," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2008, pp. 426–442.

[31] J. Doget, E. Prouff, M. Rivain, and F.-X. Standaert, "Univariate side channel attacks and leakage modeling," *Journal of Cryptographic Engineering*, vol. 1, no. 2, pp. 123–144, 2011.

[32] M. K. Steven, "Fundamentals of statistical signal processing," *PTR Prentice-Hall, Englewood Cliffs, NJ*, vol. 10, no. 151045, p. 148, 1993.

[33] J. G. Proakis, *Digital signal processing: principles, algorithms, and applications, 4/E*. Pearson Education India, 2007.

[34] P. Hamilton, "Open source ecg analysis," in *Computers in cardiology*. IEEE, 2002, pp. 101–104.

[35] R. B. Randall, *Vibration-based condition monitoring: industrial, automotive and aerospace applications*. John Wiley & Sons, 2021.

[36] I. McLean and A. Urken, "Classics of social choice. university of michiganpress," *Ann Arbor, Michigan*, 1995.

[37] E. Prouff and M. Rivain, "A generic method for secure sbox implementation," in *International Workshop on Information Security Applications*. Springer, 2007, pp. 227–244.

[38] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

[39] J. R. Pennock and J. W. Chapman, *Due Process: Nomos XVIII*. NYU Press, 1977, vol. 18.

[40] L. Masure and R. Strullu, "Side channel analysis against the anssi's protected aes implementation on arm. iacr cryptol. eprint arch," 2021.

[41] P. Kulkarni, V. Verneuil, S. Picek, and L. Batina, "Order vs. chaos: A language model approach for side-channel attacks." *IACR Cryptol. ePrint Arch.*, vol. 2023, p. 1615, 2023.

[42] M. Rivain and E. Prouff, "Provably secure higher-order masking of aes," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2010, pp. 413–427.

[43] S. Picek, G. Perin, L. Mariot, L. Wu, and L. Batina, "Sok: Deep learning-based physical side-channel analysis," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–35, 2023.

[44] J. Blömer, J. Guajardo, and V. Krummel, "Provably secure masking of aes," in *International workshop on selected areas in cryptography*. Springer, 2004, pp. 69–83.

[45] T. S. Messerges, "Securing the aes finalists against power analysis attacks," in *International Workshop on Fast Software Encryption*. Springer, 2000, pp. 150–164.

[46] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A side-channel analysis resistant description of the aes s-box," in *International workshop on fast software encryption*. Springer, 2005, pp. 413–423.

[47] H. Kim, S. Hong, and J. Lim, "A fast and provably secure higher-order masking of aes s-box," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2011, pp. 95–107.

[48] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9*. Springer, 2007, pp. 450–466.

[49] A. Rezaeezade, T. Yap, D. Jap, S. Bhasin, and S. Picek, "Breaking the blindfold: Deep learning-based blind side-channel analysis," *Cryptology ePrint Archive*, 2025.

[50] A. Rezaeezade, A. Basurto-Becerra, L. Weissbart, and G. Perin, "One for all, all for ascon: Ensemble-based deep learning side-channel analysis," in *International Conference on Applied Cryptography and Network Security*. Springer, 2024, pp. 139–157.

[51] M. J. Dworkin, "Recommendation for block cipher modes of operation: The xts-aes mode for confidentiality on storage devices," 2010.

VII. Biography Section

Jintong Yu received the B.S. degree from Harbin Institute of Technology. She is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. Her research interests include deep-learning based side-channel analysis, post-quantum cryptography and cryptographic engineering.

Yuxuan Wang received the B.S. degree from Shanghai Jiao Tong University. He is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. His research interests include side-channel analysis, fault analysis, and post-quantum cryptography.

Zixin He received the B.S. degree from Shanghai Jiao Tong University. She is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. Her research interests include far field side-channel analysis, deep-learning based side-channel analysis and cryptographic engineering.

Yihan Nie received the B.S. degree from Shanghai Jiao Tong University. He is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. His research interests include side-channel analysis and post-quantum cryptography.

Yubo Zhao received the B.S. degree from Harbin Institute of Technology. He is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University. His research interests include side-channel analysis, machine learning and cryptographic engineering.

Zhiliang An received the B.S. degree from Beijing University of Posts and Telecommunications. He is currently pursuing the Ph.D. degree with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. His research interests include side-channel analysis and cryptographic engineering.

Yipeng Shi received from Northwestern Polytechnical University of China, Xi'an, his B.S. degree in software engineering. He is currently a Ph.D. candidate at School of Computer Science, Shanghai Jiao Tong University, Shanghai. His research interests include white-box cryptography and cryptographic engineering.

Pei Cao received the B.S. degree in information security from the University of Electronic Science and Technology of China and the Ph.D. degree in computer science and technology from Shanghai Jiao Tong University. He is currently an Assistant Research Fellow at Shanghai Jiao Tong University. His current research interests include hardware security and side-channel analysis.

Chi Zhang received the B.S. degree from Southeast University and the Ph.D. degree from Shanghai Jiao Tong University. He is currently an Assistant Research Fellow with the School of Computer Science, Shanghai Jiao Tong University, Shanghai, China. His research interests include side-channel analysis and cryptographic engineering.

Dawu Gu (Member, IEEE) received the B.S. degree in applied mathematics from Xidian University, Xi'an, China, in 1992, and the M.S. and Ph.D. degrees in cryptography in 1995 and 1998, respectively. He is a Distinguished Professor with the School of Computer Science, Shanghai Jiao Tong University (SJTU), Shanghai, China. He is the Vice President of the Chinese Association for Cryptologic Research, China. He has got over 200 scientific papers in top academic journals and conferences, such as CRYPTO, IEEE Security and Privacy, ACM CCS, NDSS, and TCHES. His research interests include crypto algorithms, crypto engineering, and system security. Dr. Gu was the winner of Chang Jiang Scholars Distinguished Professors Program in 2014 by Ministry of Education of China. He won the National Award of Science and Technology Progress in 2017. He also served as PC Member of international conferences for more than 30 times.