Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.074473 Published Online: 19 December 2025

REVIEW

Recent Advances in Deep-Learning Side-Channel Attacks on AES Implementations

Junnian Wang[1], Xiaoxia Wang[1], Zexin Luo[1], Qixiang Ouyang[1], Chao Zhou[1] and Huanyu Wang[2,*]

[1]School of Physics and Electronic Science, Hunan University of Science and Technology, Xiangtan, 411201, China
[2]School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, 411201, China

*Corresponding Author: Huanyu Wang. Email: huanyu@hnust.edu.cn

Received: 11 October 2025; Accepted: 18 November 2025

ABSTRACT: Internet of Things (IoTs) devices are bringing about a revolutionary change our society by enabling connectivity regardless of time and location. However, The extensive deployment of these devices also makes them attractive victims for the malicious actions of adversaries. Within the spectrum of existing threats, Side-Channel Attacks (SCAs) have established themselves as an effective way to compromise cryptographic implementations. These attacks exploit unintended, unintended physical leakage that occurs during the cryptographic execution of devices, bypassing the theoretical strength of the crypto design. In recent times, the advancement of deep learning has provided SCAs with a powerful ally. Well-trained deep-learning models demonstrate an exceptional capacity to identify correlations between side-channel measurements and sensitive data, thereby significantly enhancing such attacks. To further understand the security threats posed by deep-learning SCAs and to aid in formulating robust countermeasures in the future, this paper undertakes an exhaustive investigation of leading-edge SCAs targeting Advanced Encryption Standard (AES) implementations. The study specifically focuses on attacks that exploit power consumption and electromagnetic (EM) emissions as primary leakage sources, systematically evaluating the extent to which diverse deep learning techniques enhance SCAs across multiple critical dimensions. These dimensions include: (i) the characteristics of publicly available datasets derived from various hardware and software platforms; (ii) the formalization of leakage models tailored to different attack scenarios; (iii) the architectural suitability and performance of state-of-the-art deep learning models. Furthermore, the survey provides a systematic synthesis of current research findings, identifies significant unresolved issues in the existing literature and suggests promising directions for future work, including cross-device attack transferability and the impact of quantum-classical hybrid computing on side-channel security.

KEYWORDS: Side-channel attacks; deep learning; advanced encryption standard; power analysis; EM analysis

1 Introduction

The Internet of Things (IoT) is bringing about a revolution in society with advanced connectivity and real-time analytics, turning sensor data into immediate and actionable insights for better operations. Real-time analytics is key to preventing downtime and managing risks, but integrating it with IoT involves challenges such as securing the system. However, since numerous embedded edge devices commonly execute encryption and decryption operations on-site, significant security concerns are triggered by Side-Channel Attacks(SCAs), as shown in Fig. 1. According to the inherent imperfections of the physical implementations, there might be some physical leakage, such as power consumption and electromagnetic (EM) emissions, and such leakages could disclose sensitive data in IoT embedded devices. Adversaries can exploit these non-intentional physical leakages to analyze and extract the secret. These are called SCAs. The secret key

leaking from a cryptographic module could cause information security to be entirely compromised across IoT systems [1].

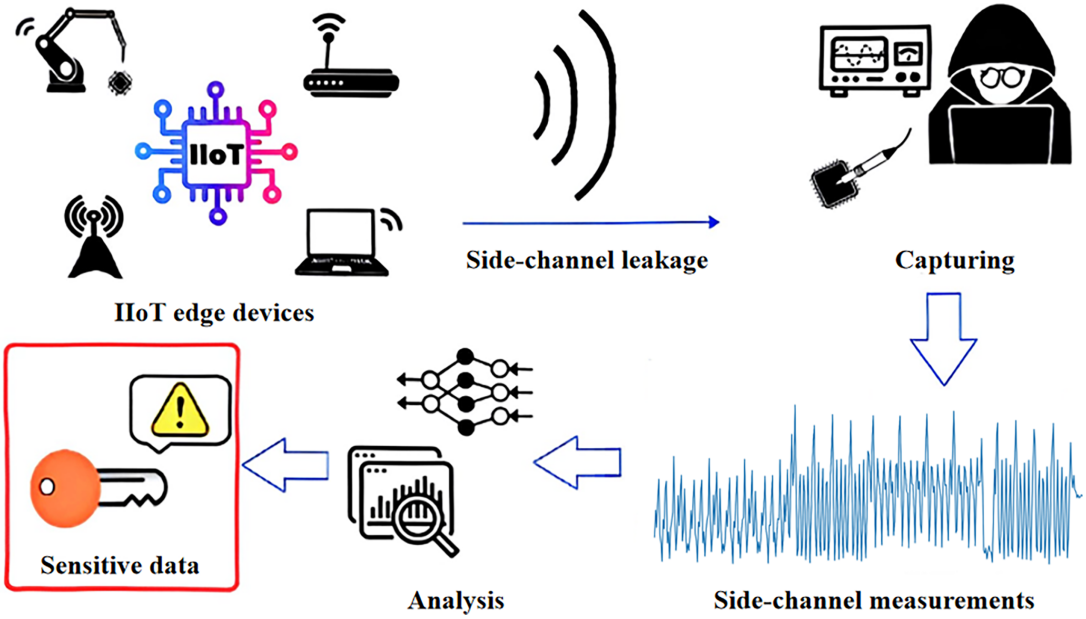


Figure 1: Illustration of how SCAs extract secrets from physical devices. It depicts the process where attackers capture side-channel leakage from devices, analyze the measurements via techniques like deep learning, and ultimately extract sensitive data

Over the past two decades, diverse side channels have been exploited across various applications. These include compromising cryptographic implementations [2,3], reverse-engineering neural network architectures [4,5], stealing intellectual property [6], monitoring user-browser information [7,8], predicting the output generated by random number generators [9], tracking how code executes [10,11] and intercepting victims' password from the keystroke [12,13] and fingerprints [14].

Different side channels can be exploited in various practical scenarios. Consider far field EM SCA [15–17] (also called screaming channel attacks), which pose a security threat due to their remote execution capability. These attacks remain effective even when the target device operates in a seemingly secure office environment without direct physical access. Adversaries could position themselves in a neighboring office and capture the EM traces using specialized radio equipment. By analyzing these captured traces, the attacker may achieve various malicious objectives. In another scenario, the adversary might not be interested in the sensitive data itself but rather in the behavior of the victim device. In this case, the attacker could track the execution path of the victim's code by analyzing the traces of specific blocks or even library functions used by the device [10].

However, the attack scenarios can vary significantly depending on the type of side-channel being exploited. For instance, when power consumption is used as the side channel, the attacker generally requires physical access to the target device to measure power consumption during its operation. A practical example of this could involve an adversary legally acquiring a car, which grants them access to the car's digital key. Once in possession of the key, the adversary could capture power traces while using the key to communicate with the car. By analyzing these traces, the cryptographic information shared between the car and the key could be extracted. This would enable the adversary to create unauthorized copies of the car key, which

could allow them to illegally replicate and sell backup car keys to customers, without the car company's authorization. This would undoubtedly cause significant damage to the company's profits and reputation.

Presently, there exist six commonly utilized side channels within both academia and industry, as shown below.

- Power consumption [18], involves exploiting the inherent variations in power consumption exhibited by logic circuits during different operations and with varying data inputs.
- Time consumption [19], exploiting differences that depend on the data in the execution time.
- Optical leakage [20], involves exploiting differences of optical properties of silicon due to changes in voltage or current.
- Acoustic leakage [21], which entails exploiting the piezoelectric characteristics of ceramic capacitors, functions in power supply filtering and the conversion of AC to DC.
- Near field EM emissions [22], resulting from the rapid change of current in the logic components. These emissions exhibit high-frequency components and are typically identified by a close-proximity probe placed in the immediate vicinity of the chip.
- Far field EM emissions (screaming channels) [15], resulting from the coupling that occurs between distinct components on mixed-signal chips. These far-field EM emissions are detectable at a certain distance away from the target device. Thus, attackers do not have to physically approach the victim.

This paper focuses on two side channels: power consumption and EM emissions.

In recent years, Deep Learning (DL) techniques have become extremely prevalent owing to their remarkable ability to find complex patterns and make accurate predictions or classifications from large volumes of data. This popularity is attributed to their success in multiple fields like computer vision [23], natural language processing [24], edge computing [25], and resource management [26]. However, like any great scientific discovery, deep-learning techniques have the potential to be used for malicious purposes. For example, in most Deep-Learning Side-Channel Attacks (DLSCAs), attackers start by building a deep learning model as a leakage profile, aiming to establish a relationship between sensitive data and side-channel traces collected from a copy of the target device—a replica referred to as the profiling device, over which the adversaries have complete control. Afterwards, the profiled model can be utilized to categorize the traces collected from the target device, enabling adversaries to extract sensitive data. Typically, a thoroughly trained deep-learning model is capable of enhancing attack efficiency by several orders of magnitude, which is substantially greater than the performance of traditional signal processing methods [16], such as Correlation Power Analysis (CPA) [27] and Template Attacks (TA) [28]. Furthermore, various deep-learning models have demonstrated their capability in assisting adversaries to bypass diverse countermeasures in side-channel attacks. For example, Convolutional Neural Networks (CNNs) have proven to be effective in addressing misaligned traces and overcoming countermeasures based on jitter [29].

As deep-learning based SCAs continue to grow in threat and significance, the foremost concern revolves around effectively mitigating these attacks. It is essential to comprehend the capacities and restrictions of deep-learning based side-channel attacks to develop robust defensive measures in the future. Therefore, comprehensively reviewing literature of DLSCAs is crucial. Although there are certain reviews within the realm of side-channel attacks, their coverage remains inadequate. Reference [30] concludes the conventional SCAs approaches, such as Differential Power Analysis (DPA) [18], template attacks, correlation power analysis, Mutual Information Analysis (MIA) [31], and Test Vector Leakage Assessment (TVLA) [32]. Afterwards, Hettwer et al. review the attacks based on Machine Learning (ML) techniques in [33], for instance the Support Vector Machine (SVM) [34], Decision Trees (DTs) [35] and random forest [36], as presented in the 2020 JCEN journal. In 2023, Picek et al. [37] provided a systematically review of DLSCAs in ACM

Computing Surveys, across a broad range of applications and side channels. Reference [38] reviews the current research status of EM-SCA in cryptographic attack scenarios, with a focus on three core dimensions: cryptographic algorithms vulnerable to cryptography, designs and device implementations resistant to attack, and promising emerging EM-SCA attack paradigms. Notably, the review does not place emphasis on the role and application impact of DL techniques in the field of SCA. Reference [39] focuses on attack analysis in SCA based on deep learning techniques, wherein it conducts systematic evaluation and comparative analysis of diverse deep learning-enhanced SCA schemes. Specifically, the performance of these schemes is assessed and compared against the ANSSI SCA database (ASCAD) as the benchmark testbed. Reference [40] provides a comprehensive review of recent advances in attack techniques targeting the Advanced Encryption Standard (AES). Specifically, the reviewed attack methods are systematically categorized into four distinct research domains, namely SCAs, fault injection attacks (FIAs), attacks based on machine learning and artificial intelligence (ML/AI), and quantum computing-enabled threats. Reference [41] investigates SCAs targeting implementations of Post-Quantum Cryptography (PQC) algorithms, and categorizes these attacks from an adversarial perspective to identify the most vulnerable components in the implementations of such algorithms. Building on these foundations, we take a step further by presenting a comprehensive summary of cutting-edge DLSCAs on AES [42] with a particular focus on power analysis and EM analysis across diverse attack scenarios, since AES is the symmetric cryptographic algorithm most widely employed. Our analysis examines these attacks from multiple perspectives, including deep-learning model architectures, hyperparameter optimization, and differences between hardware and software implementations. We need to first stress out that some approaches and works might fit into more than one categories in our survey.

Contributions and Paper Structure

We provide a comprehensive review of how deep-learning techniques can be used in power and EM based analysis to compromise different implementations of AES. AES stands as the most extensively employed symmetric cryptographic algorithm across IoT devices due to its pivotal role in maintaining a specific degree of information security.

For every attack vector, we conduct a systematic review on methodologies presented in the literature and make comparisons across various criteria, including the quantity of measurements needed for the attack, target physical implementations, leakage models, and countermeasures applied. Our primary contribution lies in identifying the DL methods and corresponding parameters that are suitable for specific attack scenarios.

The paper's structure is outlined below. Section 2 offers a thorough overview of AES, DLSCAs, and various DL approaches. In Section 3, we review research studies on DLSCAs that utilize power consumption as the side channel. In the last, Section 4 exploits open questions related to DLSCAs on AES. Sections 5 and 6 provide a summary of the paper and discuss future works.

2 Background

This section begins with a review of the AES. Following that, it introduces the concepts of deep learning and explains the role that deep learning plays in enabling side-channel attacks. In addition, it covers various leakage models and commonly utilized evaluation metrics for SCAs.

2.1 Advanced Encryption Standard

AES, a symmetric encryption algorithm, was standardized by the U.S. National Institute of Standards and Technology (NIST) in 2001. It is often the preferred choice for implementing cryptographic modules in IoT embedded devices for applications that requiring secure communications and data encryption. This preference is primarily due to the fact that AES is known for its speed, efficiency, and wide support within

the industry. In these cases, AES is implemented with different key sizes with n bits ($n = 128, 192,$ or 256), called AES-n. The device using AES-n has a secret key shared with other authorized parties, $\mathcal{K}$. For instance, AES-128 employs a key size comprising 128 bits. Operating as a block cipher, AES-128 employs the secret key $\mathcal{K}$ to encrypt a 128-bit block of plaintext denoted $\mathcal{P} \in 0, 1^{128}$ and generates a corresponding 128-bit block of ciphertext $\mathcal{C} \in 0, 1^{128}$. The quantity of encryption rounds in AES varies according to the size of the key, with 10 rounds for AES-128, 12 rounds for AES-192, and 14 rounds for AES-256.

Using AES-128 as an example, except for the final round, each round consists of four repetitive steps: non-linear substitution (*SubBytes*), transposition of rows (*ShiftRows*), mixing of columns (*MixColumns*), and round key addition (*AddRoundKey*). The final encryption round is not included the *MixColumns* instruction. In the AES algorithm, the SubBytes operation involves the replacement of an 8-bit symbol with another symbol, and this substitution relies on a lookup table known as the *SBox*. AES-128 operations follow matrices in a column-major order of 4×4, with each element representing a byte. In every encryption round, excluding the final round, initially, the 16-byte intermediate state X is rearranged into a 4×4 matrix and undergoes processing to obtain the subsequent 16-byte intermediate state Y. In each round, a unique round key RKi, $i \in \{1, \ldots, 10\}$, derived from the original key K, is applied in the *AddRoundKey* operation.

A practical SCA aims to obtain secrets, such as the encryption key used in AES implementations, where the collection of all potential keys is denoted by $\mathcal{K}$. With the aim of retrieving the secret key, attackers often utilizes some known data (like plaintext and ciphertext) and side-channel measurements for the analysis, to identify the correlation between the measured data and the key-related sensitive value. An attack point refers to an intermediate state chosen to describe measurements obtained from side-channel analyses. The selection of attack points for the AES is determined by physical leakage characteristics under specific implementation scenarios, with the target being intermediate values whose computation induces measurable side-channel signals. For software-based AES implementations, the *SubBytes* operation involves substituting an 8-bit symbol with another via a pre-stored lookup table *SBox*, which serves as the core attack point. This is attributed to the fact that reading the *SBox* output to the data bus triggers charge/discharge processes of MOS capacitors—specifically, bit flips between the input address and output data exhibit a positive correlation with power consumption peaks. In contrast, for hardware-based AES implementations utilizing FPGAs or ASICs, the round key XOR operation is critical. Owing to its CMOS-based implementation, this operation generates switching currents proportional to the switching activity of transistors. Consequently, the attack point commonly selected for hardware implementations is the XOR operation between the input and output of the final round.

2.2 Deep Learning

Through machine learning techniques, systems can extract features from the provided data, allowing them to acquire their own knowledge. As a subset of machine learning, deep learning makes use of deep neural networks as models. While traditional machine learning models usually work on human-engineered features, deep-learning approaches aim to use the deep neural networks for learning features directly from raw data ahead of the task. DL techniques have become the privileged tool for dealing with the many tasks such as classification and prediction. SCAs leveraging deep learning demonstrate strong potential, with performance varying significantly across architectures

Multilayer Perceptron (MLP) ranks among the fundamental neural network types, comprising three components: one input layer, one or more hidden layer(s), and one final output layer. In each layer, except for the output layer, multiple bias neurons are included, and each neuron maintains a full connection to the subsequen layer. Usually a back-propagation training algorithm is applied to train MLPs, in which two steps are repeated on the training sets: forward pass and backward pass. The model assesses how much the true

output differs from the classified one, which is called the network output loss. During the backward pass stage, the weights of neurons are updated by calculating the gradients to reduce the loss.

CNN is a category of neural network that employs shape information by employing convolutional layers [43]. Typically, a CNN model is constructed with a series of convolutional modules, where each module is made up of a convolutional layer with a pooling layer coming after it. Following the final module, there are one or more dense layers. The ultimate dense layer incorporates one neuron per class, employing a softmax activation function for classification. One convolutional layer employs a defined number of convolution filters on the raw data of side-channel traces to extract and learn higher-level features, which the model subsequently utilizes for classification. The output of the convolution operation is often referred to as a feature map. A pooling layer serves to reduce the dimensionality of feature maps extracted by convolutional layers through downsampling. In the context of the side-channel attack, CNN models are usually employed to overcome countermeasures, to break masked AES and, to handle noise in traces [29].

Transformer Network (TN) is a neural network architecture that captures long-range dependencies via self-attention mechanisms, distinguishing it from local-feature-focused models like CNNs [44]. Typically, a Transformer model for side-channel tasks consists of stacked encoder layers, where each encoder layer comprises a multi-head self-attention sublayer and a feedforward neural network (FFN) sublayer. Following the encoder stack, there are one or more dense layers; the final dense layer integrates task-specific outputs, employing a softmax activation function for key-related classification in side-channel attacks. The multi-head self-attention sublayer computes relevance weights between all positions in side-channel trace sequences, enabling the model to focus on leakage-correlated segments across the entire trace length. The FFN sublayer then transforms the attention-augmented features to enhance their discriminative power. In the context of side-channel attacks, Transformer can readily capture dependencies between distant PoIs, making them a choice for countering cryptographic implementations protected by measures like masking [45].

A Graph Neural Network (GNN) [46] is a type of neural network used for graph-related deep learning, comprising three core components: an input layer, one or more message-passing hidden layers, and an output layer. In hidden layers, each node aggregates neighbor features via predefined functions while retaining its own to preserve node characteristics. Typically, GNNs are optimized via gradient-based training with two iterative steps on training data: forward and backward propagation. In backward propagation, parameter gradients are computed to update parameters iteratively, minimizing loss and enhancing GNN's ability to capture graph correlations. Notably, GNNs are emerging as a promising option for enhancing the accuracy and effectiveness of SCA detection models, a advantage rooted in their demonstrated efficacy in capturing relational information inherent to graph-structured data [47].

An autoencoder (AE) [48] converts the inputs to an efficient internal representation and then generates the output that is very similar to the input. Thus, In an autoencoder, the output layer's neuron count generally matches the number of inputs. An autoencoder is typically made up of two parts: an encoder and a decoder. The encoder part converts the inputs to an internal representation, and the decoder converts the obtained internal representation to the output. One way to use the autoencoder in SCA scenarios is to denoise the side-channel trace in side-channel measurements [49].

Attention Mechanism and Multi-Scale Convolutional Neural Network (AMCNNet) [50] is a specialized neural network architecture designed for SCA, which enhances feature extraction from multi-dimensional trace data by fusing convolutional layers with attention mechanisms. Typically, an AMCNNet model is composed of a sequence of attention-convolution modules, where each module consists of a multi-channel convolutional layer followed by a channel-wise attention sublayer.The multi-channel convolutional layer applies task-specific filters to raw side-channel traces across multiple data channels, capturing both local temporal features and cross-channel correlations. The channel-wise attention sublayer then assigns

adaptive weights to different feature channels, emphasizing leakage-relevant channels while suppressing noise-interfered ones.

2.3 Deep Learning Side-Channel Attack

Side-channel attacks are commonly classified into two categories: *non-profiled* and *profiled*.

Non-profiled attacks seek to directly compromise the implementations of cryptography, exemplified by techniques including DPA [18] or CPA [27].

Profiled attacks begin by learning a leakage profile that establishes a connection between the captured side-channel measurements and the cryptographic algorithm's sensitive intermediate value that is dependent on the key. This learned profile is then utilized to carry out the attack. Profiled attacks generally demonstrate superior efficiency compared to non-profiled attacks when subjected to identical attack conditions. In some specific scenarios, the attack efficiency of profiled attacks can even be several orders of magnitude higher. Nevertheless, it is essential to highlight that setting up profiled attacks generally involves more preparation than non-profiled attacks. In order to understand the leakage profile of the device across all potential values of the sensitive intermediate state, adversaries are required to have full control over one or more copies of the victim device, called profiling device(s), to capture an extensive number of side-channel traces and associated data to construct the leakage profile. Template attacks [28], as an example, utilize traces obtained from the profiling device to generate a set of probability distributions. These distributions are then employed to characterize traces based on the relevant key-dependent intermediate value. This allows the attacker to compare the traces obtained from the targeted device against the templates, facilitating the most probable key used in the cryptographic algorithm [51].

Barring a small number of exceptions, [52], deep learning techniques are frequently employed for side-channel attacks in profiled scenarios in most instances [29,53–56]. Typically, deep-learning side-channel attacks involve the following two steps (as shown in Figs. 2 and 3):

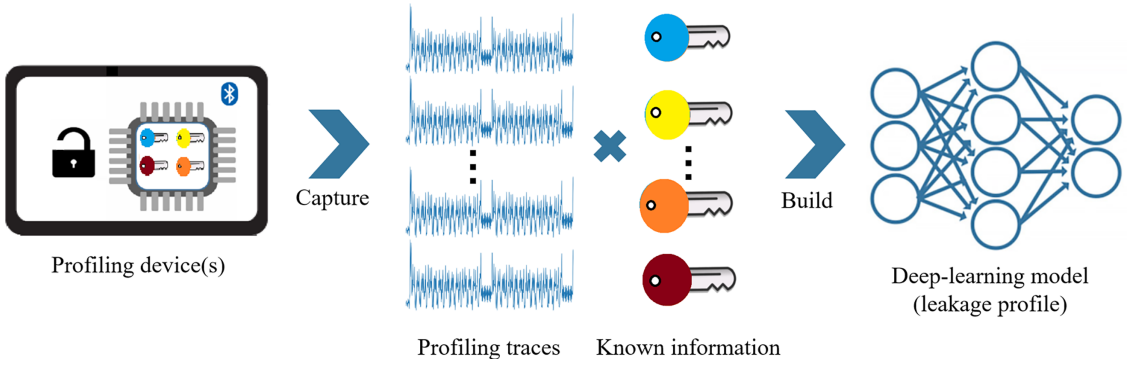


Figure 2: The overview of the profiling stage in DLSCAs on implementations of AES. It illustrates the process: capture side-channel profiling traces from a device with keys, then build a DL model to learn the leakage profile for subsequent attacks

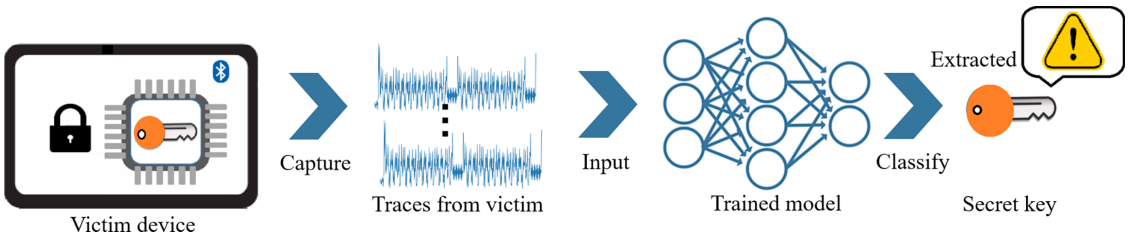


Figure 3: The overview of the profiling stage in DLSCAs on implementations of AES, capture traces from a victim device, input them into a trained deep-learning model, and classify to extract the key

Profiling stage. During this stage, the common assumption is that adversaries possess full control over at least a single profiling device. The device resembles the victim device and programmed with the identical version of the cryptographic algorithm. Therefore, the attacker can capture numerous side-channel traces and gather related information from the profiling device(s) such as known plaintexts and keys.

We use $\boldsymbol{\mathcal{T}} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{|\boldsymbol{\mathcal{T}}|}\}$, where $\mathcal{T}_i \in \mathbb{R}^m$, and $i \in \{1, \ldots, |\boldsymbol{\mathcal{T}}|\}$, to indicate a set of traces for detailed analysis. Every trace denotes the entire or partial execution process of AES. The plaintext $\mathcal{P}_i \in \{0,1\}^{128}$, secret key $\mathcal{K}_i \in \{0,1\}^{128}$ and ciphertext $\mathcal{C}_i \in \{0,1\}^{128}$ are all the corresponding information for the trace $\mathcal{T}_i$. We use $\mathcal{P}_{i,j}$ and $\mathcal{C}_{i,j}$ to represent the jth byte of the 16-byte plaintext $\mathcal{P}_i$ and ciphertext $\mathcal{C}_i$, with $j \in \{0, 1, \ldots, 15\}$. Subsequently, the selected deep-learning model $\mathcal{M}_j$ for the jth subkey is trained to understand the leakage profile linking traces and the jth key-dependent labels. Based on the chosen attack point and leakage model, the label is determined (as described below).

Attack stage. At this phase, we presume that the adversary can identify a means to capture a restricted set of side-channel traces from the victim device. In addition, the attacker can also record some known information $\hat{\mathcal{X}}_i$. Afterwards, the DL model $\mathcal{M}_j$, after being trained, classifies the traces obtained from the target device. Next, the attacker can use the known information recorded in collaboration with the classification results to deduce the secret subkey $\mathcal{K}_j$.

Notice that a well-optimized deep learning model is able to notably boost how efficient side-channel attacks are, often outperforming traditional signal processing methods by several orders of magnitude. For example, to compromise an AES implementation, the template attack used in [15] requires 52K traces from the target device at 1 m distance. Each trace amounts to an average of 500 measurements conducted under identical encryption conditions. When it comes to the deep-learning based method, reference [16] only requires 350 traces by using the CNN model to compromise the same AES implementation at 15 m distance, which is a five-order improvement in attack efficiency, even with a longer attack distance. However, this increased efficiency comes at a cost: DLSCAs typically demand substantially more preparation resources, such as computational power, training data, and time, compared to conventional techniques. For some cases the adversaries are unable to obtain the copy of the victim device as a profiling device to train the model, the challenge rises. Thus, a big concern for scenarios where deep learning offers clear advantages in SCAs is that adversaries have to obtain one or more profiling devices that resemble the victim device and can be programmed with the identical version of the cryptographic algorithm.

2.3.1 Attack Point and Leakage Model

In a side-channel attack, the attack point refers to a specific point or component in the cryptographic algorithm or its implementation that is targeted by the attacker to exploit side-channel information. This attack point is selected based on its potential to leak information related to the secret key or other sensitive data.

The choice of attack point is determined by various factors, such as the side channel type, the characteristics of the target device or algorithm, and the attacker's knowledge and resources. In software implementations of AES (microcontrollers, microprocessors), frequently exploited points of attack include both the initial and final rounds' *SBox* output. This is because in software implementations, the resulting 8-bit symbol from the SBox procedure is typically transferred from the memory to a data bus, which often leads to a higher power consumption compared to other processes. It is acknowledged that the predominant share of power use in CMOS devices stems from switching activity within a hardware implementation (such as FPGA or ASIC) of AES. As a result, the attack point for hardware implementations of AES is commonly selected as the XORed value derived from the input and output of the last round.

Within a side-channel attack, a leakage model refers to a mathematical or statistical representation that captures how side-channel measurements of a device relate to the underlying operations or data being processed. The leakage model can capture how side-channel measurements correlate with the internal states or computations of the cryptographic algorithm. A leakage model typically describes how the side-channel measurements of a device changes based on the values of specific variables or intermediate data during the algorithm's execution. Some commonly used leakage models in SCAs include:

1. Identity (ID) model. Under the assumption of the ID model, at the attack point side-channel, measurements are considered to be directly proportional to the processed data's value. For instance, when the data contains one byte, the ID model generates a set of $2^8 = 256$ classes.
2. Hamming weight (HW) model. Within the HW model, the assumption is that side-channel measurements demonstrate proportionality to the number of 1*s* in the data processed at the attack point.
3. Hamming distance (HD) model. Within the HD model, it is assumed that side-channel measurements exhibit proportionality to how many $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions occur between two states in the cryptographic algorithm undergoing processing. Two transitions are considered to have the same impact.

2.3.2 Leakage Detection

In practical scenarios, the traces obtained to represent the execution of AES can consist of a large number of samples, reaching into the thousands or even millions. Performing an attack in such circumstances can require significant resources and time. To streamline the attack process, adversaries often employ leakage detection methods to identify Points of Interest (PoIs) within the traces [57]. This allows them to concentrate on a smaller subset of points where information leakage is more pronounced. The act of pinpointing the leakage interval in side-channel measurements, aimed at extracting information tied to secrets, is known as leakage detection. The TVLA technique [32], utilizing the widely recognized Welch's t-test [58], has gained significant prominence as a statistical method for detecting information leakage [59,60]. This approach has become widely used among the available methods to spot information leaks.

To conduct a TVLA, the captured side-channel measurements are initially segregated into two distinct groups based on the associated key-dependent intermediate value, known as the label, processed by the device. This division entails creating a set $\boldsymbol{\mathcal{T}}_0$ of traces where $HW(label) > \beta$, and another set $\boldsymbol{\mathcal{T}}_1$ containing traces where $HW(label) < \beta$. Here, $HW(label)$ denotes the HW of the label set to undergo processing, representing the count of 1s in the binary form of the processed intermediate value. Afterwards, the t-test is utilized by selecting a sample from each of the two sets of traces is used to check for a significant difference, operating under the null hypothesis that the two sets have equal means.

In TVLA, the Second-Order Statistical Test (SOST) is employed to evaluate the difference between the two trace groups, which is defined by Formula (1).

$$SOST = \left(\frac{\mu_0 - \mu_1}{\sqrt{\frac{\sigma_0^2}{n_0} + \frac{\sigma_1^2}{n_1}}} \right)^2 \tag{1}$$

where μ_i and σ_i represent the mean and standard deviation of the set $\mathcal{T}_i$, respectively, while N_i denotes the number of data within the set.

Fig. 4 shows an example of how leakage detection allocates PoIs for side-channel traces. The upper picture Fig. 4 illustrates the leakage detection results for 16 subkeys in the first round of a STM32F3 MCU implementation of AES-128. The bottom picture Fig. 4 shows an example trace that represents an encryption round. The dashed red line illustrates the PoI allocation for the first SubByte operation in the first round of AES-128. By doing this, adversaries can focus on the specific trace segment and ignore other information to make the profiling stage more efficient.

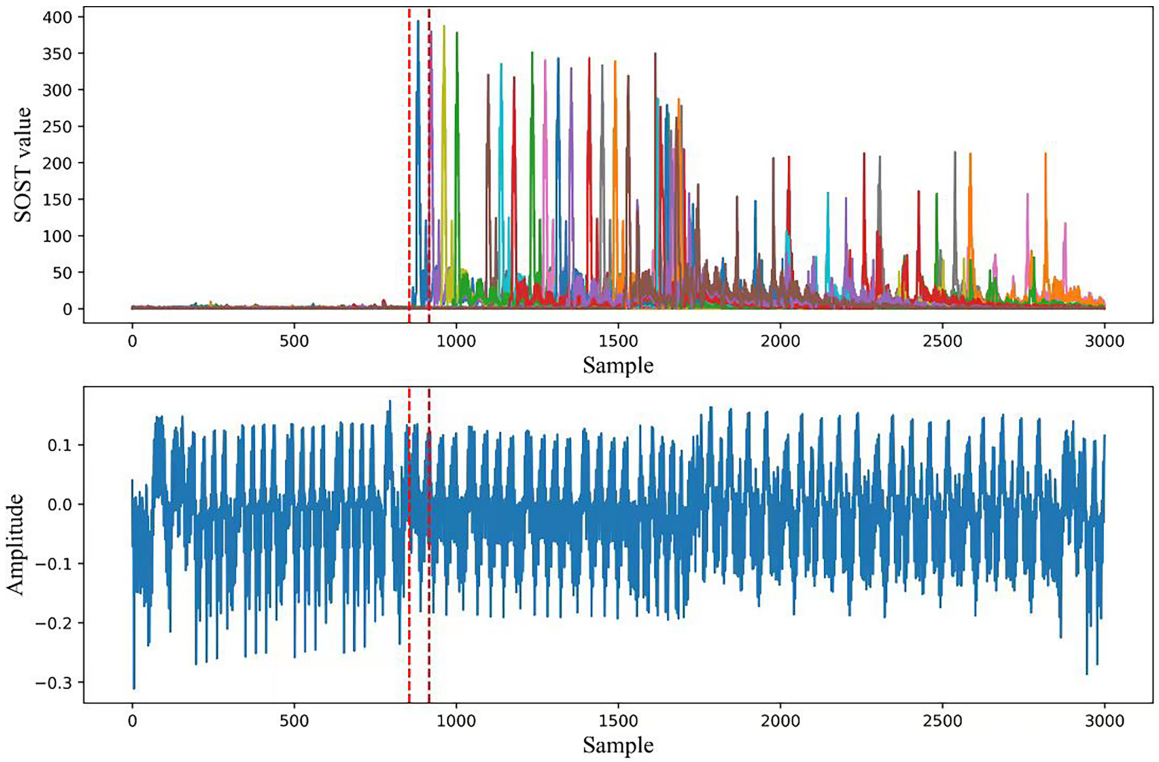


Figure 4: An example of how leakage detection works for side-channel traces. The top picture shows the leakage detection results for 16 subkeys in the first round of a STM32F3 MCU implementation of AES-128. The bottom picture shows an example trace which representing an encryption round

2.3.3 Evaluation Metrics

Within the side-channel community, Success Rate (SR) and Guessing Entropy (GE) [61] stand out as two extensively employed evaluation metrics [62]. These metrics provide a fair and standardized framework for evaluating how effectively an attacker can exploit side-channel leakage to recover cryptographic keys. SR gauges the likelihood of successfully identifying the correct key within a specified number of traces, while GE

quantifies the average uncertainty by calculating the expected rank of the correct key within a key-ranking scenario. Together, they offer complementary insights on how efficient and robust side-channel attacks are under different conditions.

The success rate quantifies the likelihood that the correct subkey can be successfully identified using a given set of side-channel traces. It reflects the model's capacity to reduce uncertainty and precisely pinpoint the subkey within a specified number of traces. A higher success rate indicates that the adversary requires fewer traces to reliably recover the subkey, demonstrating how effective the model is at extracting keys from side-channel leakage. For instance, in the context of single-trace SR, this metric represents the probability that a model can correctly classify the key-dependent value using only one trace. In an experimental setup with 100 testing traces, if the model successfully recovers the key from 90 traces while failing in 10 cases, the single-trace SR for this model in that specific scenario is calculated as 90.0%.

In certain attack scenarios, relying solely on the SR might not be adequate as an evaluation metric, and it is advisable to include guessing entropy as an additional measure. GE offers information about the degree to which the secret key is disclosed given a specific number of traces. Guessing entropy means the degree of uncertainty or randomness associated with predicting the exposure of the secret key via side channels in side-channel attacks. As a common metric for evaluating attack complexity, guessing entropy implies that higher entropy corresponds to a lower likelihood of success for the attacker.

When recovering 8-bit subkeys individually, the guessing entropy is performed independently for every subkey, and the estimation metric used is the Partial Guessing Entropy (PGE), instead of GE [63]. Represented by PGE is the expected *rank* of the actual subkey. The *rank* of a subkey serves to measure the position of the correct subkey value k^* within the key guessing vector. For example, Through the classification of a trace set, the trained deep-learning model could generate a cumulative key guessing vector g, which denotes the probabilities of all possible subkey values. Once the correct subkey value k^* holds the highest probability in g (appearing first in $Sort(g)$), the rank of the correct subkey is denoted as 0. The PGE represents the average rank of k^*.

Within DLSCAs, adversaries typically begin by constructing a deep-learning model to serve as a leakage profile, which connects side-channel traces to key-related labels. During the profiling stage, various pre-defined loss functions in the deep-learning community are employed to evaluate the extent to which the trained deep-learning classifier fits the training traces. For example, in most attacks targeting software implementations of AES [16,64,65]. It is common to use *categorical cross-entropy loss* for quantifying classification errors (see below), as these attacks can often be simplified into multi-class classification tasks.

$$CE = -\sum_{i \in C} t_i \log\left(\frac{e^{s_i}}{\sum_{j \in C} e^{s_j}}\right), \tag{2}$$

where t_i and s_i are the ground truth and the classifier score for each class $i \in C$. To recover an 8-bit subkey $\mathcal{K}_j$, each trace can belong to one of $|C| = 256$ classes and the model's output is a probability array over the 256 classes for each trace.

3 Practical Attack Cases of DLSCAs

For identifying capabilities and enabling the comparison of different DLSCAs on AES implementations, power consumption is used as the side channel, we provide a comprehensive review of existing work in this section.

Paul Kocher first introduced power analysis in 1999 [18]. The power consumption of a device can depend on the particular data undergoing processing and the operations being executed. If an operation is

correlated with some key-dependent intermediate state, the attacker is able to examine the power consumed by the target device and deduce the key. Power-based side-channel attacks capitalize on fluctuations in power usage during the execution of encryption on the victim device, with such fluctuations potentially differing according to various input data and operations [66]. Fig. 5 shows the comparison of two power traces corresponding to different data calculations within the same instruction, indicating that different data executed by the victim device result in distinct side-channel measurements.

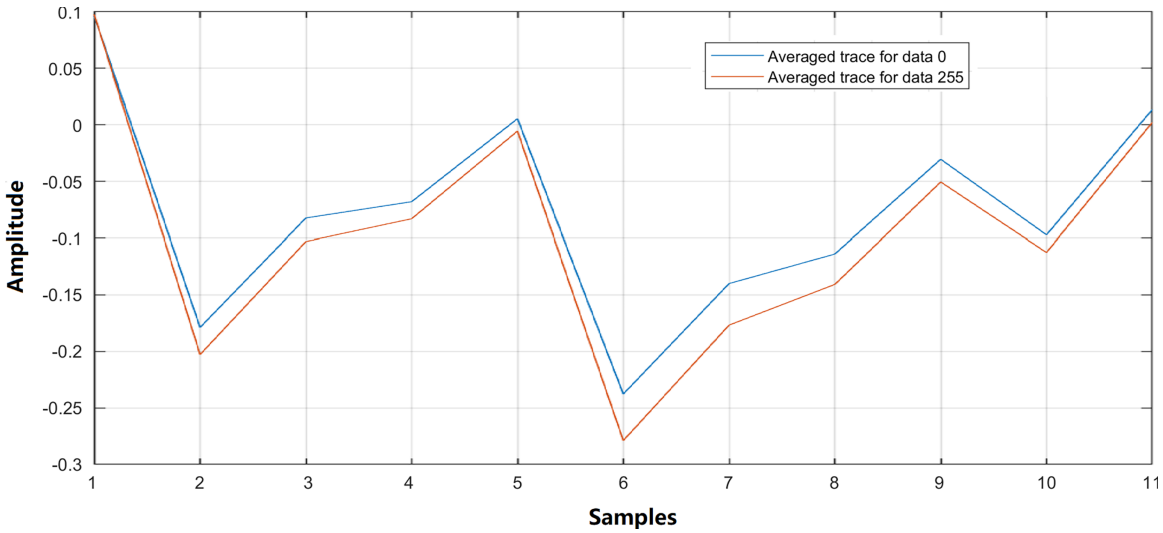


Figure 5: The comparison of two power traces (averaged over 100 times) indicates that different executed data by the victim device results in power consumption (side-channel measurements)

One method to gauge a device's power consumption is to insert a small resistor in series with the power or ground input, typically referred to as a shunt. Power consumption is determined by dividing the resistor's voltage drop by its resistance [67]. A series of power measurements obtained by sampling these voltage drops using an oscilloscope over a specific duration is referred to as a trace.

In the following subsections, we start by introducing three publicly available power-based SCA datasets. Afterwards, we show existing works of DLSCAs on both software and hardware AES implementations, employing power consumption as the side channel.

3.1 Publicly Available Datasets for Power Analysis

In power analysis, there are five widely exploited side-channel datasets: ASCAD, DPA_V2, DPA_V4, AES_RD, and AES_HD. These datasets provide traces collected from different cryptographic implementations under various scenarios, enabling the community to evaluate attack techniques and countermeasures.

3.1.1 Datasets of Software Implementations

DPA_V4 (DPA contest V4 Dataset). The DPA_V4 dataset[1] is collected from the DPA contest at a later stage. For the DPA_V4 dataset, the target is an Atmel ATMega-163 smart-card implementation of AES with Rotating Sbox Masking (RSM) [68]. Included in it are 16Kb of in-system programmable flash, 512 bytes of EEPROM, 1Kb of internal SRAM and 32 general purpose working registers. A simple reader interface mounted on the SASEBO-W board is used to read the smartcard. The traces are collected with a LeCroy

[1]The DPA_V4 dataset is publicly available at https://dpacontest.telecom-paris.fr/v4/index.php (accessed on 02 November 2025).

WaveRunner 6100A oscilloscope. The acquisition bandwidth reaches 200 MHz, while the sampling rate is adjusted to 500 MS/s. Containing 80K traces, the dataset has 16 different keys, with each key, there are 5K traces corresponding to the encryption of 5K different plaintexts per key. The dataset requires approximately 64 GB of storage.

ASCAD (ANSSI SCA Dataset). The ASCAD dataset[2] includes traces collected from two distinct victim devices: an ATMega8515 implementing a Boolean-masked AES and an STM32F303RCT7 implementing an affine-masked AES. These datasets are commonly denoted as ASCADv1 (for the ATMega8515 implementation) and ASCADv2 (for the STM32F303RCT7 implementation). ASCADv1 comprises 60K synchronized traces captured using a fixed secret key and 300K jitter-based traces obtained with variable keys. The traces' desynchronization is simulated artificially. To sample measurements in the ASCAD dataset, a digital oscilloscope operates at a sampling rate of 2G samples every second. A temporal acquisition window is set to record the first round of the AES only. Every trace is made up of 1.4K sample points, while the dataset takes up around 77 GB of storage space. ASCADv2 contains 800K traces, each generated with random plaintexts and keys. The traces consist of 1M sample points, covering the entire AES encryption process. The dataset requires approximately 807 GB of storage; however, a smaller extracted dataset is available, which is only 7 GB in size, offering a more convenient option for quick access.

AES_RD. AES_RD[3] contains power consumption traces captured from an 8-bit Atmel AVR microcontroller implementation of AES-128, where the algorithm is protected by a random delay countermeasure [69]. The AES_RD dataset contains 50K traces in total and each trace has 3.5K samples, which are captured with the LeCroy WaveRunner 104MXi oscilloscope. These power traces are compressed by selecting 1 sample of each CPU clock cycle. The dataset is compactly organized, occupying less than 1 GB of storage.

3.1.2 Datasets of Hardware Implementations

DPA_V2 (DPA contest V2 Dataset). The DPA_V2 dataset[4] is collected as part of the DPA Contest. As an international contest, the DPA Contest permits researchers across the globe to compete on a common basis. Introduced in 2008, this contest has seen four versions completed since that time. In the case of DPA_V2, The target here is an AES-128 hardware implementation, which is set up on the Xilinx Virtex-5 FPGA of a SASEBO GII evaluation board. A high-resolution oscilloscope is used to measure the power consumed by the device when the encryption process takes place. The dataset contains 1M traces in total with 3.2K sample points per trace. The dataset requires approximately 9 GB of storage. For research convenience, the data are well-aligned after the capturing process.

AES_HD. AES_HD[5] is a dataset captured from an unprotected AES-128 hardware implementation on a Xilinx Virtex-5 FPGA, integrated into a SASEBO GII evaluation board. Note that the acquisition method is not explicitly mentioned on the AES_HD dataset website. This dataset comprises 100K traces, with each linked to a distinct random plaintext. Each trace contains 1.25K sample points. The dataset is efficiently stored, requiring less than 1 GB of storage space.

[2] The ASCAD dataset is publicly available at https://github.com/ANSSI-FR/ASCAD (accessed on 02 November 2025).
[3] The AES_RD dataset is publicly available at https://github.com/ikizhvatov/randomdelays-traces (accessed on 02 November 2025).
[4] The DPA_V2 dataset is publicly available at https://dpacontest.telecom-paris.fr/v2/participate.php (accessed on 02 November 2025).
[5] The AES_HD dataset is publicly available at https://github.com/AISyLab/AES_HD (accessed on 02 November 2025).

The characteristics of the five datasets are summarized in the subsequent Table 1.

Table 1: Key characteristics of benchmark datasets in DLSCAs

Dataset	Sampling rate	Sampling piont	Trace count
ASCADv1	200 MS/s	1.4K	360k traces
ASCADv2	50 MS/s	1M	800K traces
DPA_v2	500 MS/s	3.2K	1M traces
DPA_v4	500 MS/s	1,704,402	80k traces
AES_RD	–	3.5K	50k traces
AES_HD	–	1.25K	100k traces

Note: '–' indicates that the sampling rate is not provided in the original dataset documentation.

3.1.3 Comparison on Datasets

When comparing side-channel analysis datasets such as ASCAD, AES_RD, AES_HD, DPA_V2, and DPA_V4, their relevance to real-world IoT environments can be assessed based on their characteristics and limitations.

ASCAD, AES_RD and DPA_V4, which provide traces from microcontrollers running AES, are highly representative of software-based cryptographic implementations in real-world IoT environment. Software-based AES is widely used in low-cost and resource-constrained IoT devices. A large number of such devices function using low-power chips that lack dedicated cryptographic accelerators, making software implementations the only viable option. Among them, AES_RD and parts of the ASCAD dataset contain traces captured from 8-bit MCU implementation of AES, which can be used to represent low-power, cost-sensitive IoT applications, such as smart sensors, simple automation systems, and basic communication modules, due to their simplicity, low energy consumption, and affordability. However, implementing AES in software on an 8-bit MCU presents significant performance challenges. AES functions as a block cipher, handling data blocks of 128 bits in size, which must be split into multiple 8-bit operations when executed on an 8-bit MCU. This results in a high number of memory accesses, increased computational overhead, and slow encryption speeds.

For the DPA_V4 and another part of the ASCAD dataset, traces are captured from 32-bit MCU implementations of AES. In this case, the data can represent a more scalable approach for software-based AES in IoT, which achieves a more reasonable balance between the security level and the cost. Real-world IoT scenarios that commonly rely on software AES in 32-bit MCUs include industrial automation controllers, connected medical devices, and smart home gateways.

AES_HD, DPA_V2 can be used to represent the IoT applications with hardware cryptographic module, as they both contain traces captured from AES implementations on FPGAs. In contrast, hardware-based AES is preferred for performance-critical applications that demand fast encryption and decryption. IoT devices involved in real-time video streaming, high-throughput communication, or industrial control systems benefit significantly from dedicated cryptographic accelerators, as hardware AES completes encryption operations faster. Security-critical IoT applications, including payment terminals, secure bootloaders, and biometric authentication systems, also rely on hardware AES due to its resistance to attacks.

3.2 Power Analysis of Software Implementations

A software-based AES implementation denotes the execution of the AES algorithm through software or programming code. In this approach, the AES algorithm is executed on a general-purpose computing device, such as a computer or a microcontroller, using software instructions to perform the required cryptographic operations. The software implementation typically involves translating the AES algorithm's steps, such as key expansion, substitution, permutation, and XOR operations, into programming instructions executable by the device's Central Processing Unit (CPU). Software implementations of AES are commonly used in various applications, including secure communication protocols, file encryption, and cryptographic libraries.

Advantages of software implementations of AES compared to hardware implementations include:

1. Flexibility: Software implementations offer great flexibility in terms of programmability and adaptability to different platforms and operating systems [70].
2. Cost-effectiveness: Software implementations typically require less upfront investment compared to hardware implementations, as they can utilize existing computing infrastructure without the need for dedicated hardware components [71].
3. Compatibility: Software implementations of AES can be developed to comply with standardized cryptographic libraries and protocols, ensuring compatibility and interoperability with other software systems [72].

In 2011, Hospodar et al. proposed, in the JCEN journal, one of the initial machine learning-based attacks, trains a Least Squares Support Vector Machine (LS-SVM) [73] to classify power traces captured from an AES software implementation based on the first round's SBox output. Instead of recovering an actual key, Hospodar et al. [74] focus on one single SBox lookup process. The investigation focuses on three significant properties of the SBox output: whether the HW is smaller or larger than 4, whether it is odd or even, together with the value of the fourth least significant bit. This analysis aims to illustrate the impact of LS-SVM hyperparameters on classification accuracy.

Afterwards, in 2013, deep-learning techniques began contributing to power analysis [75]. A three-layer MLP network is applied for training to compromise an AES-128 SmartCard implementation, featuring an 8-bit microcontroller PIC16F84. In [75], the MLP model undergoes training by using the standard sigmoid as the activation function and achieves a 85.2% single-trace attack accuracy to recover the first subkey of the implementation. Since then, the development of neural network-based power analysis targeting various implementations of AES has gradually emerged.

By training a MLP model on the generated patterns, reference [76] is able to improve classification accuracy to 96.5% for the same implementation of AES as in [75]. Afterwards, reference [77] compares the newly introduced MLP based attack in [75,76] with the conventional template attack on the same dataset as in [75,76]. However, in [77], only 2560 traces are captured for training the MLP model, which may lead to the result far away from the optimal. In [77], the PGE result of their MLP model is 1.04 on average, which indicates that the trained model's capability to extract the subkey from an AES-128 Smart Card implementation which features an 8-bit microcontroller PIC16F84.

In addition to MLPs, reference [56] explores the efficiency of various other deep-learning models in enhancing side-channel attacks using three distinct datasets. In [56], five different deep-learning attack approaches are compared in total: MLP with Principal Component Analysis (PCA), MLP without PCA, CNN, AE and Long and Short Term Memory (LSTM). For the sake of comparison, they also test a machine learning based approach (random forest) and the template attack on the same three datasets. They experimentally show the overwhelming advantage of the deep learning based when it comes to breaking both unprotected and protected AES.

To further examine research on power analysis based on CNNs, Cagli et al. [29] utilize the CNN model alongside a data augmentation method [78], introduced at CHES 2017. This approach aims to overcome the trace misalignment and handle countermeasures based on jitter. Cagli et al. [29] initially highlight the challenges of the traditional template attack strategy, particularly in addressing trace misalignment. This requires the attacker to meticulously realign the captured traces. Subsequently, they conduct experiments demonstrating that the CNN based strategy significantly streamlines the attack process by eliminating the need for trace realignment and precise selection of points that are of interest. In their initial trial, they used CNNs to compromise the implementation of AES on an ATmega328P microprocessor, which featured a uniform Random Delay Interrupt (RDI). By using the HW leakage model to train their CNN network, they achieve to use 7 traces on average to recover a subkey, which confirms that their CNN model is robust to RDI. Additionally, their second experiment introduces clock instability to misaligned traces on the adversary's side. The findings show that the CNN method is superior to Gaussian templates in performance, regardless of whether trace realignment is performed. Meanwhile, reference [79] also examines the advantages of CNNs in comparison to various ML techniques. Their experiments reveal that methods such as random forest may be a preferable choice over CNN in certain attack scenarios, casting doubt on CNN as the optimal approach for every profiled SCA setting. Another work [80] for CNN based attacks shows that the addition of manually added non-task-specific noise in training sets may prove advantageous to the attack efficiency of the trained network. The addition of such noise may be considered being on par with incorporating a regularization term. a regularization term. CNN models trained on data with additional non-task-specific noise in [80] can successfully recover the key by using 2 traces for the AES_RD dataset. For the case of DPA_V4 dataset, the addition of noise helps the pooled template to break the implementation with 2 traces, which outperforms other CNN-based approaches.

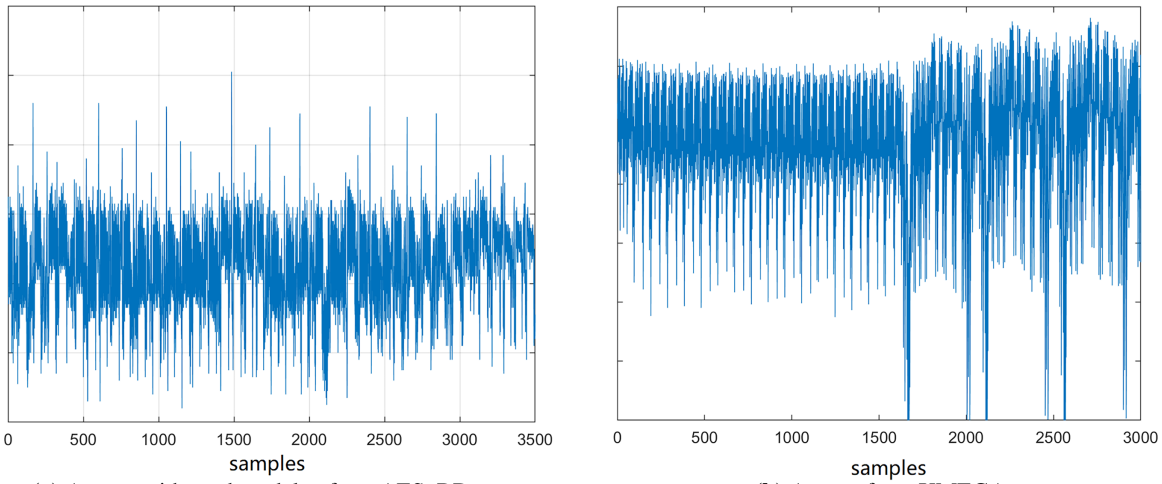
During the profiling stage, the HW leakage model as well as the HD leakage model face a common challenge of dealing with imbalanced data. For instance,if the labels of the data are evenly distributed across the range from 0 to 255, certain classes occur in 1/256 instances (when the HW parameter is set to 0) while one appears in 70/256 instances (When HW parameter is set to 4) by using the HW leakage model. With the aim of reducing the effect of imbalanced profiling data, reference [81] uses a data balancing approach called Synthetic Minority Oversampling Technique (SMOTE) [82] to ensure a balanced distribution of classes.

Table 2: Overview of current DLSCAs utilizing power consumption as the side channel on the AES_RD dataset

Work	Best classifier	Leakage model	Result
[81]	Random forest	HW	≈1619 traces
[80]	CNN	ID	≈600 traces to reach $PGE < 20$
[83]	CNN	ID	171 traces
[84]	CNN	One bit	10 traces
[85]	CNN	ID	5 traces
[45]	TransNet	ID	2 traces
[86]	InceptionNet	ID	3 traces
[87]	CNN	ID	1953 traces

In [88], neural networks are trained using each bit of the intermediate data processed at the attack point as a separate label. Consequently, for a subkey represented as a byte, a total of 8 labels are considered. To address the class imbalance challenge, this method is introduced, unlike the HW leakage model, every

bit exhibits almost a uniform distribution. They evaluate the multi-label approach in several datasets. For three publicly available datasets, called ASCAD, AES_RD and AES_HD, the proposed multi-label model requires 202, 10 and 831 traces, respectively, for the subkey recovery. Power traces in AES_RD are captured from an 8-bit Atmel AVR microcontroller implementation of AES-128, where the algorithm is protected via a random delay countermeasure [69]. In all, the dataset consists of 50,000 traces, and each trace is made up of 3500 samples. These power traces are compressed by selecting 1 sample from each CPU clock cycle. Fig. 6a shows an example trace of AES_RD dataset and Table 2 shows the summary of existing DLSCAs on AES_RD dataset.



(a) A trace with random delay from AES_RD. **(b)** A trace from XMEGA.

Figure 6: Plots of traces captured from software implementations of AES. The first figure shows a power trace with random delay captured from an 8-bit AVR microcontroller implementation of AES-128 (AES_RD dataset) and the second plot shows a trace captured from an 8-bit Atmel ATXmega128D4 microcontroller implementation of AES-128

In 2022, the denoising autoencoder is applied in [89] to reduce or remove noise and the countermeasure effect before the training process. The denoising autoencoder is trained by using noisy-clean trace pairs. Regarding the ASCAD dataset, the CNN model in [89] requires 831 traces to retrieve the correct key before the denoising process that uses the trained autoencoder. After degenerating the noise level, the results decrease to 751. Afterwards, Zaid et al. [90] proposed a Conditional Variational AutoEncoder in 2023 TCHES, which bridges DL models and SCA paradigms based on theoretical findings from stochastic attacks. By following the path of [89], Hu et al. [65] further design a multi-loss DAE model which makes the power and near-field EM based SCAs more efficient, presented in 2023 TIFS. Despite their strength, deep-learning techniques face a well-known limitation: reliance on large datasets. This issue has received relatively little focus within the side-channel community. However, within practical scenarios, the quantity of profiling traces available to adversaries is often far smaller than assumed, constrained by factors such as limited preparation time or scheme-specific restrictions. As a result, improving the efficiency of the profiling stage in DLSCAs is also an important area of focus. Reference [91] proposes a Label Correlation (LC) based profiling method, in which transferring the widely used one-hot labels toward their patterns to accelerate the convergence of model profiling. Their experiments demonstrated that both CNN and MLP models could successfully recover subkeys from ASCAD traces with only 10K profiling traces, underscoring the potential of this approach to improve profiling efficiency. Reference [92] proposes a novel CPA method applicable to scenarios where cryptographic algorithms employ parallel implementations of S-boxes, and narrow the performance gap between profiling and non-profiling SCA attacks.

Numerous existing architectures enhance model accuracy through the stacking of multiple network layers, which consequently introduces several challenges: elevated algorithmic and computational complexity, overfitting phenomena, reduced efficiency in the training process, and constrained feature representation capacity. In addition, deep learning methods depend on data correlation, and noise presence often reduces this correlation, thus making attacks more difficult. TN exhibit a strong capability to capture dependency relationships between distant POIs in side-channel traces; leveraging this advantage, Hajra et al. [45] employ TN to launch attacks against cryptographic implementations equipped with protective measures such as masking and desynchronization. Zhang et al. [93] propose a CNN-Transformer architecture, which integrates power analysis techniques to automatically identify and prioritize relevant POIs in side channel power consumption traces. Empirical evaluations on dataset demonstrate that, compared with LSTM and CNN models, this hybrid architecture achieves significantly higher attack efficiency in DL-SCAs. In 2024, reference [50] proposes an AMCNNet for better extraction of the feature and temporal information from traces. Reference [86] suggests applying a network structure based on InceptionNet to side-channel attacks. The proposed network employs a reduced number of training parameters, attains accelerated convergence via parallel processing of input data, and enhances attack efficiency. Additionally, a network architecture based on LU-Net is put forward to denoise side-channel datasets. On the AES_RD and ASCAD datasets, 3 and 30 traces are used to recover the subkeys. Reference [87] designs a lightweight deep learning model by incorporating random convolutional kernels to address the exponentially increasing training time caused by excessive features. Compared with the most advanced methods, the quantity of power traces required as well as the trainable parameters are reduced by over 70% and 94%, respectively. Reference [94] proposes a general side-channel evaluation metric called Leading Degree (LD) for assessing the performance of deep learning models. Through the use of LD as the reward function in the tuning of model hyperparameters based on reinforcement learning, a better model structure is obtained compared with previous state-of-the-art models.

3.2.1 Hyperparameter Tuning

While many existing works focus on the attack efficiency of the designed model, they always keep their hyper-parameterization as a secret. The comprehensive study on the selection of hyperparameters has not been fully explored when developing deep-learning models across various side-channel attacks' scenarios [95]. In order to address this limitation, reference [96] investigates the impact of modifying hyper-parameters in MLP and CNN models for side-channel attacks. The methodologies for selecting hyper-parameters can serve as a guidance for the following researchers to optimize their own deep-learning models. Specifically, the study not only puts forward a customized hyperparameter selection framework for DL models, encompassing pivotal parameters such as batch size, epochs, filter configurations, and fully-connected layers, but also systematically elucidates how adjustments to these hyperparameters directly modulate critical model attributes, thereby exerting a decisive influence on the reliability of DL-SCA models in practical key recovery tasks. The impact of some hyperparameters on the model is shown in the Fig. 7. In [96], their best CNN model is about to recover a single subkey from a masked AES implementation on an 8-bit ATMega8515 microcontroller utilizing around 200 traces without any desynchronization. For the case of traces with a maximal desynchronization value of 50, they succeeded in achieving a mean rank close to 20 with 5K traces. In the scenario where the maximum desynchronization value is 100, the model requires 5K traces to reach a mean rank near 40. In addition, reference [96] shows that attacks based on MLPs and CNNs exhibit notable superiority over template attacks when traces are desynchronized.

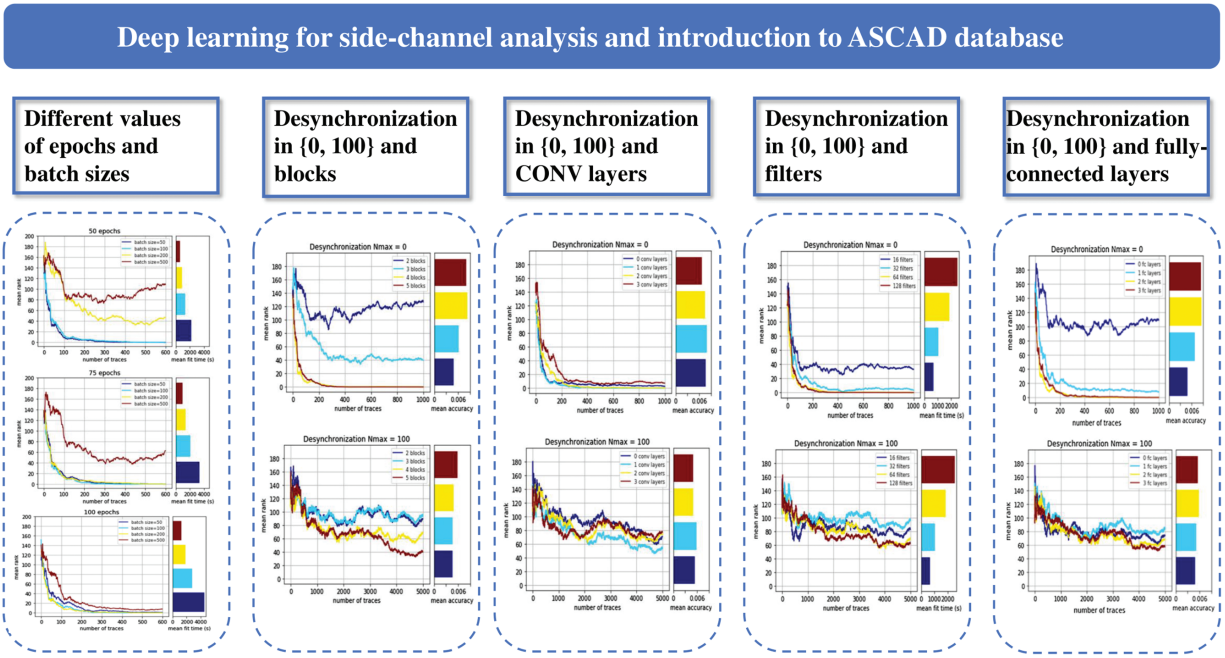


Figure 7: An analysis of deep learning for SCA using the ASCAD database, investigate the impacts of different hyperparameters (epochs, batch sizes, blocks, CONV layers, filters, fully-connected layers) and desynchronization levels on model performance

Hyperparameter optimization plays plays a vital role in enhancing both the performance and robustness of models used in SCAs. Traditional approaches like grid search and random search systematically explore different hyperparameter combinations, though they can be computationally expensive, as shown in many SCA cases [65,97,98]. More sophisticated techniques that include Bayesian optimization offer a more efficient method by modeling the hyperparameter space probabilistically, allowing the search to focus on promising regions of the attack [99,100]. A traditional CNN model is optimized by incorporating an attention mechanism into the convolutional layers, thereby strengthening the model's capability to capture global information and improving the extraction of leakage information [101]. Another effective strategy is learning rate scheduling, where the learning rate is adjusted dynamically during training, often using techniques like cosine annealing, step decay, or adaptive methods [102]. To elucidate the impact of each hyperparameter of neural networks in side-channel attack scenarios during the feature selection phase, reference [85] employs three visualization techniques to illustrate the internal operations of models: weight visualization [103], gradient visualization [104], and heatmaps [105]. By employing these visualization approaches, reference [85] demonstrates different methodologies to build suitable CNN architectures for different attack scenarios. They show that their optimal model achieves using 3 traces to recover a subkey from DPA_V4 dataset. Comparing this result to [80] and [81], they managed to lower the model's complexity without degenerating the model accuracy. For the case of ASE_HD, ASCAD and AES_RD datasets, the CNN models in [85] require 1050, 191 and 5 traces to recover the subkey. To address the threats posed by SCAs, one defensive measure involves injecting random noise to enhance security. Reference [106] proposes a novel denoising approachthat combines wavelet coefficient analysis with Generative Adversarial Networks (GANs). This integration is tailored to mitigate noise artifacts induced by side-channel countermeasures. Through the preprocessing of noise reduction on traces, only 108 traces are required on the ASCAD dataset compared with [85].

Afterwards, References [99,100] propose two distinct methods for automatically tuning neural networks' hyperparameters. Reference [100] builds a custom framework denoted as AutoSCA based on Bayesian

Optimization, in which the model is selected from 50 iterations of testing various hyperparameter combinations. During every iteration, the Bayesian Optimization function generates a group of hyperparameters for model construction, which is then followed by the training process. They compare different types of neural networks and show that the AutoSCA MLP model reaches the best performance for the ASCAD dataset with the shortest training time, which uses 129 traces to recover a subkey. Another automated hyperparameter tuning approach for SCA is proposed by [99], in which the reinforcement learning framework [107] is used with two reward functions for side-channel metrics to adjust the hyperparameters of convolutional neural network. In [99], 202 traces need to be used for the CNN model to recover a subkey for the ASCAD dataset. Reference [108] formulates SCA problems as graph signal processing (GSP) problems. Leveraging the inherent advantage of GNNs in modeling and analyzing graph-structured data, the study further applies GNNs to SCA tasks to enhance the extraction and utilization of leakage-related features. Table 3 shows the summary of existing DLSCAs on ASCAD dataset.

Table 3: Summary of existing DLSCAs on the ASCAD dataset

Work	Best classifier	Leakage model	Result
[49]	AE + TA	ID	160 traces to reach $PGE < 2$
[83]	CNN	ID	552 traces
[84]	CNN	One bit	202 traces
[85]	CNN	ID	191 traces
[86]	InceptionNet	ID	30 traces
[99]	CNN	ID	202 traces
[100]	MLP	ID	129 traces
[45]	TransNet	ID	≈200 traces
[50]	AMCNNet	ID	155 traces
[101]	CNN	ID	48 traces
[106]	CNN	ID	108 traces
[94]	CNN	ID	102 traces

3.2.2 Board Diversity

Most of reported deep-learning based side-channel attacks on AES by 2019 do not account for the influence of board diversity. They conduct training as well as testing of the deep-learning models using the same board's traces, which is not realistic in a genuine attack scenario.

To examine the impact of board diversity on the efficiency of DLSCAs, Das et al. [109] experimentally demonstrate the success rate gap when the trained model is applied to the profiling device vs. the victim device, as presented at the DAC conference. They train deep-learning models using traces obtained from one 8-bit ATxmega128D4 microcontroller implementation of AES and tests this model using traces obtained from another board with the same chip and the same version of AES. Afterwards, Wang et al. [97] investigate to which extent the trained models' attack efficiency can be mitigated when targeting the victim device with the same implementation but in a different Printed Circuit Board (PCB) as the profiling device. Although the MLP model in [97] demonstrates the ability to recover the key from the training board in 88. 5% of the cases, the success rate drops significantly to 13.7% for the testing board. This demonstrates the potential for overestimating classification accuracy if the training and testing the model on the traces obtained of the same equipment. Notably, the model structure in [97] is relatively simple compared to the complex architectures typically used in computer vision and natural language processing, particularly when the captured traces exhibit a high level of leakage. For instance, the peak leakage value, represented by the SOST value, for

the first subkey derived from 5K power traces collected from the ATxmega128D4 microcontroller's AES implementation, is approximately 70. This value is roughly 15 times higher than the detectable leakage threshold (SOST value < 4.5), enabling adversaries to construct relatively simple models.

To alleviate the influence of board diversity and improve the efficiency of deep-learning models, a data-level aggregation approach is proposed by [84,109,110]. The fundamental concept of the data-level aggregation approach is to train DL models using traces obtained from multiple boards, rather than one, as shown in Fig. 8a. For example, in [84], 9 profiling devices (the same implementations as in [97]) are used to train a MLP model and increase the probability of key recovery from a single trace, increasing it from 40.0% to 86.1%. To further increase the attack efficiency, reference [110] uses 30 devices for profiling and captured traces are preprocessed by using the PCA technique before the profiling stage. As a result, they successfully achieve a ≥90% single-trace attack accuracy to break the same 8-bit ATXmega128D4 microcontroller implementation of AES as in [84,97,109]. Fig. 6b shows an example power trace obtained from an 8-bit ATXmega128D4 microcontroller implementation of AES, which represents 16 SBox operations. Table 4 summarizes the analyzed DLSCAs on ATXmega128D4 microcontroller implementations of AES. Reference [116] proposes an automated trace segmentation approach based on reinforcement learning, which is applicable to a broad range of common implementations of public-key algorithms. The authors experimentally verified the transferability of the proposed framework on over ten datasets. Reference [115] introduces Switch-T, a neural architecture grounded in the Transformer framework. This approach integrates elastic weight consolidation (EWC) with a multi-task learning paradigm to facilitate coordinated multi-task adversarial attacks. The experimental results demonstrate that the Switch-T model is capable of retaining the knowledge acquired from prior tasks in cross-architecture and cross-channel SCAs scenarios. Reference [117] proposes a novel cross-device attack method, which integrates the Denoising Diffusion Probability Model (DDPM) for universal model construction, adopts an adaptive multi-task loss function to balance multiple training objectives, and conducts evaluation of the proposed strategy on five cross-device SCA datasets.

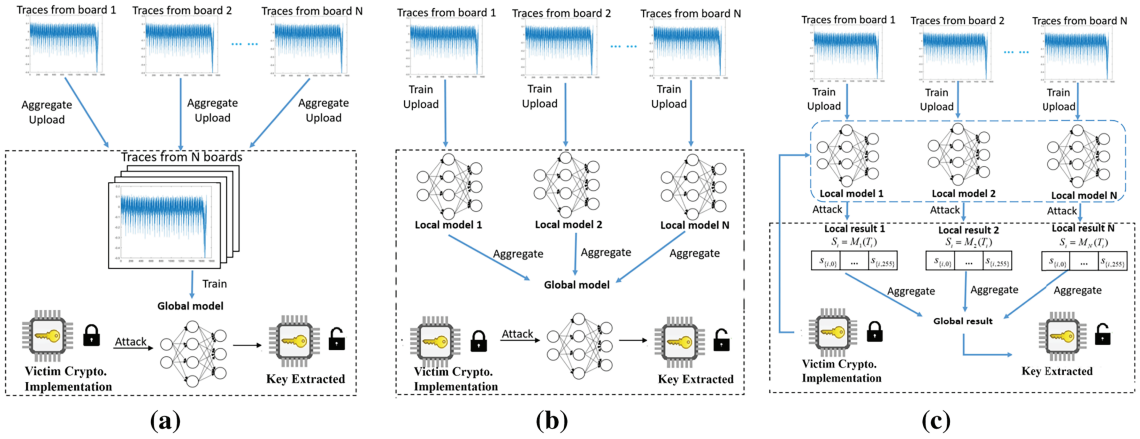


Figure 8: Illustration of different aggregation methods to mitigate the impact caused by the board diversity in DLSCAs on AES. Fig. 8a presents data-level aggregation, where a model is trained on traces from different devices. Fig. 8b shows model-level aggregation for SCA based on the horizontal federated learning framework, with N participants jointly building a federated deep-learning model. Fig. 8c illustrates output-level aggregation: N classifiers are independently trained on traces from N devices. (**a**) Data-level aggregation approach [109,110]. (**b**) Model-level aggregation approach [111]. (**c**) Output-level aggregation approach [111]

When datasets are dispersed across various sources, the task of matching data while preserving the privacy of the datasets is a widely demanded task. The preservation of the privacy of distributed data, particularly the data stored in individual edge devices, is shown to be consistently crucial [118]. Another technique to combine multiple deep-learning models is the well-known Federated Learning (FL) [119]. FL facilitates allows for training of a global deep-learning model by several participants without the need to share their individual local training data [120]. Like any significant scientific breakthrough, FL has the potential to be employed for malicious purposes. Reference [111] uses the FL framework as a model-level aggregation for the deep-learning based SCA. Fig. 8b illustrates the model-level aggregation of DLSCAs to reduce the impact brought about by board diversity. In their experiments, three 8-bit ATxmega128D4 microcontroller implementations of AES are used to train three MLP models and these models are aggregated into one global MLP. The federated model in [111] achieves a 77.7% average single-trace attack accuracy even though they take board diversity into account. Another aggregation approach proposed in [111] is at the output level, which uses the ensemble learning scheme [121] to integrate the classification outcomes of multiple models trained on various profiling devices as shown in Fig. 8c.

Table 4: Summary of existing DLSCAs on ATxmega128D4 microcontroller implementations of AES-128

Work	Best classifier	Leakage model	Result
[97]	MLP	ID	13.7% with 1 trace
[109]	MLP	ID	>90.0% with 1 trace
[110]	MLP	ID	91.72% with 1 trace
[111]	MLP	ID	77.7% with 1 trace
[84]	MLP	ID	88.7% with 1 trace
[112]	CNN	ID	93.0% with 1 trace
[93]	CNN-Transformer	ID	99.76% with 1 trace
[113]	DNN+VA	HW	901 trace
[114]	MLP	ID	1 trace
[115]	Swith-T Transformer	ID	98.7% with 1 trace

Subsequently, reference [64] achieves in demonstrating the first power analysis using deep learning to a commercial USIM card that implements MILENAGE (based on AES). The researchers successfully trained a CNN model on one USIM and demonstrated its ability to regain the key from a different USIM, using an average of only 20 traces. Afterwards, in 2023, reference [122] introduces a novel method to address the portability issue. This method presents a neural network layer evaluation method based on the ablation paradigm. By evaluating the sensitivity and resilience of each layer, this approach provides useful insights for constructing a Multiple Device Model from Single Device (MDMSD). Physical side-channel analysis generally works under the premise that plaintext or ciphertext is known, but this premise often breaks down in diverse scenarios. Blind SCA tackles this challenge by functioning without awareness of plaintext or ciphertext. Reference [113] proposes the first successful blind SCA on hiding countermeasures, introducing the Multi-point Cluster-based (MC) labeling technique. It is validated on four datasets, including symmetric-key algorithms (AES, ASCON) and the post-quantum cryptography algorithm Kyber. Reference [114] investigates the portability of deep learning-based side-channel attacks on EM traces and perform a comparative analysis of a set of

preprocessing and unsupervised domain shift methods. And a large-scale public dataset is provided for benchmarking and reproducing side-channel attacks that handle domain shifts over EM traces.

Although the varied power profiles of IoT devices present a unique challenge for DLSCAs, adversaries still pose a considerable threat. As mentioned earlier, an attacker can prepare a dedicated deep-learning model for a specific target by acquiring an identical device from the market, allowing them to train on highly similar power traces. This approach enables precise modeling of the device's power consumption characteristics, significantly improving the attack's effectiveness. Since many consumer IoT devices are mass-produced with identical components and firmware, attackers can exploit this uniformity to extract sensitive information like cryptographic keys or executed operations with high precision.

However, when the target is a dedicated or custom-built device, and the adversary has no opportunity to obtain an identical copy, the challenge becomes more significant. Variations in architecture, firmware optimizations, and even the version of the cryptographic algorithm introduce discrepancies in power consumption, thereby increasing the difficulty for attackers to generalize a pre-trained model.

3.3 Power Analysis of Hardware Implementations

Time can affect leakage in software implementations of AES, and the leakage tends to be less noisy due to the sequential execution of instructions. In contrast to the challenges posed by hardware implementations of AES, this characteristic facilitates deep-learning models in exploring features linked to each subkey through a divide-and-conquer strategy. Traces obtained from hardware implementations often exhibit overlapping features arising from multiple concurrent operations as a result of parallel instruction execution. Consequently, side-channel attacks become intrinsically more difficult, especially within advanced process technology. Here are some advantages of hardware implementations of AES compared to software implementations:

1. Speed and efficiency. Hardware implementations of AES can provide significantly faster encryption and decryption speeds compared to software implementations [123].
2. Lower power consumption. Hardware implementations of AES can be more power-efficient compared to software implementations, especially in resource-constrained devices [124].
3. Physical security. Encryption and decryption operations are performed within dedicated hardware components, rendering it more difficult for attackers to retrieve sensitive data via side-channel attacks or software vulnerabilities [95].

Many existing attacks targeting hardware implementations of AES are founded on two widely recognized datasets: DPA contest V2 [125] and AES_HD [81]. These datasets' traces are obtained from AES implementations on Xilinx Virtex-5 FPGA series. Table 5 summarizes some existing attack results on these two well-known datasets.

Table 5: Summary of existing DLSCAs on AES_HD, DPA_V2 and DPA_V4 datasets

Work	Dataset	Best classifier	Leakage model	Result
[56]	DPA_V2	CNN	ID	≈200 traces (full key)
[81]	DPA_V4	SVM	HW	3 traces
[81]	AES_HD	TA	HW	700 traces
[80]	AES_HD	CNN	ID	≈800 traces to reach $PGE < 50$
[49]	DPA_V2	AE + TA	ID	450 traces
[83]	AES_HD	CNN	ID	≈2.1K traces
[83]	DPA_V4	CNN	ID	≈3 traces
[85]	AES_HD	CNN	ID	1050 traces
[84]	AES_HD	CNN	One bit	831 traces
[85]	DPA_V4	CNN	ID	3 traces
[108]	DPA_V4	GNN	HW	10 traces
[45]	DPA_V4	TransNet	ID	2 traces
[45]	AES_HD	TransNet	ID	≈900 traces
[92]	DPA_V2	CPA	HD	495 traces
[50]	DPA_V4	AMCNNet	ID	1 traces
[50]	AES_HD	AMCNNet	ID	683 traces
[87]	AES_HD	CNN	ID	1974 traces
[86]	DPA_V4	InceptionNet	ID	1 traces

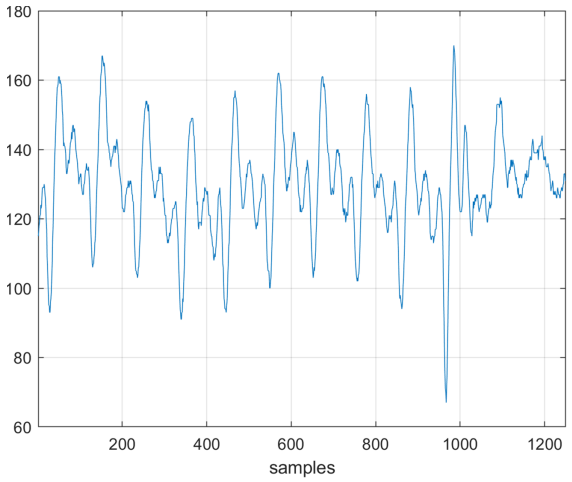
In [81], the random forest approach requires in excess of 5000 traces to retrieve a subkey from the Virtex-5 implementation of AES. Reference [126] investigates the theoretical soundness of CNN in the context of SCAs on hardware implementations of AES. References [56,79,80] showcased effective assaults on Virtex-5 FPGAs through the application of CNNs. In [80], the CNN models trained on data with additional non-task-specific noise in [80] are able to recover a subkey with 25,000 traces for the AES_HD dataset.

Apart from Xilinx Virtex-5 FPGA, a non-profiled attack [127] employs roughly 3.7K traces to compromise a lightweight Artix-7 FPGA implementation of AES. Additionally, reference [128] demonstrates the efficacy of CNN-based side-channel attacks on ASICs. Afterwards, reference [102] introduces a multi-point attack framework named tandem scheme to attack hardware implementations of AES. This approach combines the classification outcomes of CNN models trained at various attack points. In [102], the proposed tandem scheme successfully recovered a subkey from a Xilinx Artix-7 FPGA implementation of AES using 219 traces.

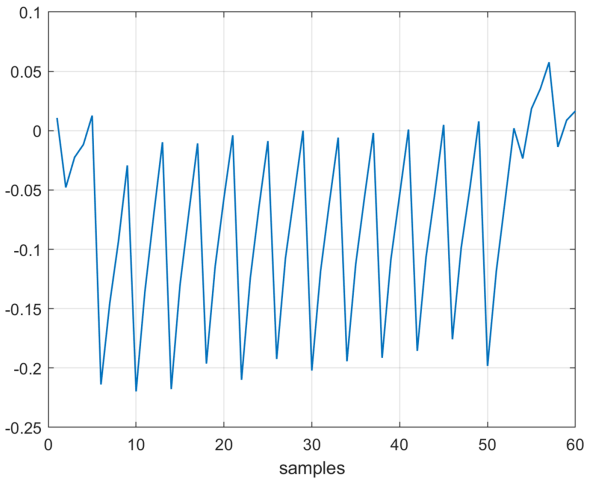
When targeting advanced embedded system implementations of AES with parallel computing and countermeasures, the preprocessing step gains significant importance, especially for degenerating the noise level in the captured traces. Initially, electronic noise inherently exists in cryptographic devices, particularly in hardware implementations. Moreover, noise serves as a fundamental component in many countermeasures designed to mitigate SCA [49]. These noise sources play a crucial role in significantly diminishing the Signal-to-Noise Ratio (SNR) of side-channel leakages, thus increasing the level of complexity in side-channel attacks. For the purpose of reducing the noise level in captured power or EM traces, the SCA community begins to use deep learning techniques, especially autoencoders, to mitigate the effects caused by environmental noise or countermeasures. In [49], the training of the Convolutional Denoising Autoencoder

(CDAE) involves learning a unique non-linear mapping that converts noisy traces into clean ones throughout the profiling phase. In the attack stage, reference [49] first uses the trained CDAE autoencoder to obtain the 'clean' traces. Afterwards, these traces are applied to a trained classifier to extract the secret key. They test their model on two publicly available datasets. For the DPA_V2 dataset, they use the ID of the register value written in the final round as the power model to train the classifier. By utilizing the trained CDAE model for the denoising process, they achieve using 450 traces to recover the key. As for the ASCAD dataset, they achieve to use 160 traces to reach GE less than 2.

Fig. 9 shows plots of example traces obtained from hardware implementations of AES. Fig. 9a shows a power trace obtained from a Xilinx Virtex-5 FPGA of AES-128 (AES_HD dataset) and Fig. 9b is a trace obtained from a Xilinx Artix-7 FPGA implementation of AES-128. From Fig. 6b in the section of software implementation, distinct operations and executions associated with different subkeys are clearly observable. However, when it comes to the Fig. 9b, we can observe that the trace segment representing 10-round operation of AES-128 obtained from an Artix-7 FPGA implementation execute the entire encryption process within several clock cycles. Intuitively, the task of training deep-learning models becomes more challenging for adversaries due to the requirement of dealing with overlapping features. Besides, by comparing the existing DLSCAs results between software and hardware implementations of AES (as shown in the tables above), it becomes evident that hardware implementations of AES consistently exhibit higher resistance against side-channel attacks compared to software implementations. Considering this, our recommendation for designing a cryptographic module with a focus on DLSCA resistance is to employ hardware implementations accompanied by suitable countermeasures. However, it is unfortunate that many cryptographic modules, particularly those utilized in lightweight IoT edge computing devices, continue to rely on unprotected software implementations due to resource limitations [17]. This leaves a significant number of systems vulnerable to the potential compromise of information security through practical, non-invasive, and highly effective attacks. It is imperative to recognize the existing threat landscape and take proactive measures to address these vulnerabilities.



(a) A trace from AES_HD dataset.



(b) A power trace from Artix-7 FPGA implementation of AES-128.

Figure 9: Graphs of traces collected from hardware implementations of AES. The initial figure depicts a power trace collected from a Xilinx Virtex-5 FPGA of AES-128 (AES_HD dataset) and the second plot shows a power trace obtained from a Xilinx Artix-7 FPGA implementation of AES-128

4 Discussion

This section begins with a discussion of the performance of deep learning models used in SCAs. Afterwards, we expand the discussion a bit on the unresolved issues in DLSCAs on AES, followed by an exploration of potential mitigation strategies.

4.1 Technical Taxonomy

The characteristics of the dataset exert a direct influence on the selection of DL models. For sequential side-channel trace datasets (e.g., ASCAD, AES_RD), where traces exhibit temporal or local feature correlations, CNNs are well-suited for extracting local sequential features through their convolutional layers [96], while Transformers demonstrate superior performance in capturing long-range dependencies via self-attention mechanisms [115]. For datasets with high noise levels and low feature dimensions (e.g., early traces in DPA_V2), MLPs are applicable for basic feature fitting; their performance can be further improved through integration with CNNs or Transformers to achieve enhanced robustness. For graph-structured datasets (e.g., datasets where side-channel trace dependencies are modeled as graphs), GNNs are the only models capable of effectively learning the inter-node relationships within the graph [47].

The complexity of leakage patterns dictates the level of "expressiveness" demanded of a DL model. For the ID leakage model, in which leakage exhibits a direct mapping to sensitive data and follows a simplistic pattern, the multilayer perceptron structure of MLPs can adequately fit such linear or weakly nonlinear relationships [88]. For the HD and HW leakage models, in which leakage involves fine-grained patterns related to bit differences or bit weights, the local feature extraction capability of CNNs and the global pattern capture ability of Transformers prove crucial, these models enable the excavation of subtle bit-level patterns corresponding to HD/HW from side-channel traces. In summary, the "complexity level" of the leakage model requires that the DL models possess commensurate feature learning capabilities, ranging from basic fitting to advanced pattern recognition. A technical taxonomy for DL-based side-channel analyses is shown in the Fig. 10.

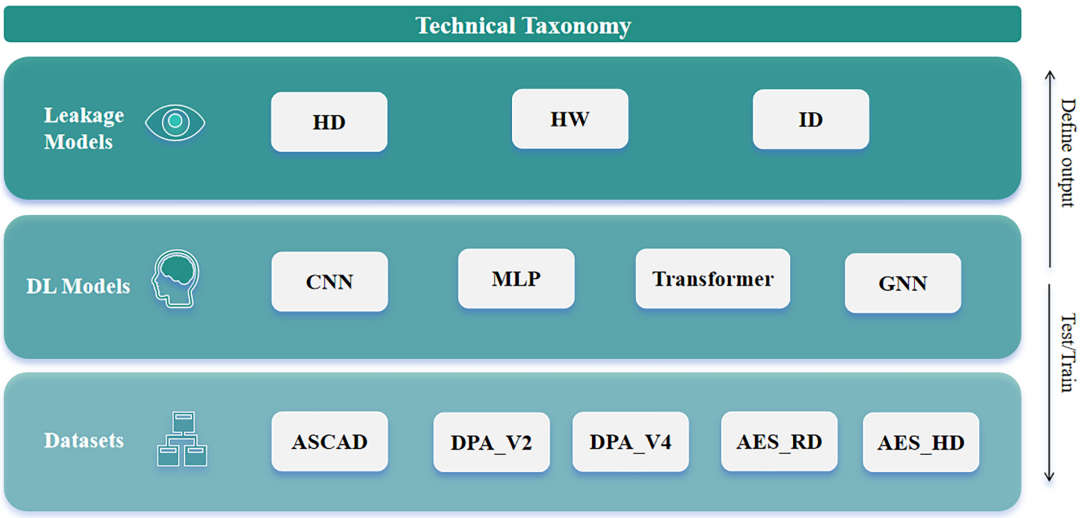


Figure 10: A technical taxonomy for DL-based side-channel analyses, comprising leakage models, DL models, and datasets

4.2 Analysis on Deep Learning Models

Deep learning methods have grown progressively prominent within the domain of SCAs due to their capacity to deal with complex patterns from side-channel measurements. As highlighted in this work, well-trained deep learning models have proven to be significantly more efficient at extracting secret keys from side-channel traces compared to traditional signal processing methods, especially when the leakage in the captured traces is relatively low. In this subsection, we expand on how different deep learning models discussed in this paper are suited for various attack scenarios.

MLPs are among the simplest deep learning models used in SCAs, typically applied when the side-channel traces are already pre-processed or feature-engineered. For example, as described in Section 3, it is feasible to achieve a high single-trace success rate by using a straightforward MLP model when targeting ATXmega128D4 implementations of AES-128. This is because the traces leaked from the target are typically well-synchronized and noise-filtered.

CNNs, by contrast, are highly effective in capturing spatial hierarchies within data, making them well-suited for side-channel traces, especially when dealing with noisy, desynchronized traces or devices implementing countermeasures. CNNs apply convolutional layers to extract local features in the data, making them especially efficient in identifying subtle leakage patterns without the need for manual feature engineering. In side-channel analysis, CNNs can learn to identify both spatial (across multiple samples) and temporal (across successive clock cycles) correlations, which is critical for capturing key-dependent information from traces. For instance, CNNs have been shown to outperform MLPs in cases where traces contain low leakage levels or are noisy, as they are better at identifying the underlying features that correspond to the secret key. A major benefit of CNNs is their capacity to work with raw side-channel data, reducing the reliance on prior data processing steps and improving attack efficiency, especially in large-scale datasets.

AEs, are another deep learning model that has gained attention in SCAs. AEs are designed to learn efficient data representations by encoding and decoding input data, and are particularly useful for anomaly detection, denoising, or feature extraction. In SCAs, AEs can be used to lower the dimensionality of the side-channel traces, removing noise and focusing the model's attention on the most important features. AEs, especially when used for anomaly detection, can help identify unusual or anomalous side-channel traces, which could then be further processed by another classifier (such as an MLP or CNN) to extract the key. The ability of AEs to compress side-channel data while preserving crucial information makes them useful in situations where available traces are noisy or contain outlier data points that might otherwise interfere with the accuracy of the model.

To summarize, the choice of model depends heavily on the particular attack scenario and the features of the side-channel data. Next, we expand on unresolved issues with specific examples in DLSCAs on AES.

4.3 Analysis on Leakage Model

The selection of an appropriate leakage model serves as a pivotal determinant that dictates the successfulness and efficiency of DL-SCAs. This is given that it delineates the core learning objective of the neural network and exerts a direct influence on the final attack accuracy metrics. Among the commonly adopted leakage models in current research, each exhibits unique advantages and inherent limitations—factors that further modulate their applicability scope and performance in specific attack scenarios.

The HD model represents a robust analytical choice that is exclusively deployed in the context of hardware - based cryptographic implementations. The efficacy of this model hinges on the attacker's capacity to mathematically model the bit - flip events occurring within a specific register during the interval between two consecutive cryptographic operations. When such precise architectural knowledge is accessible, the HD

model enables highly accurate characterization of power consumption behaviors. This capability frequently facilitates rapid and successful cryptographic key recovery attacks, thereby demonstrating its effectiveness in targeted hardware-oriented scenarios. Nevertheless, a fundamental limitation of the HD model lies in its inapplicability to softwares-based cryptographic implementations, as well as any scenario where the dynamic changes in the internal state of the cryptographic system cannot be precisely delineated. This constraint ultimately restricts the HD model to a relatively narrow domain of side-channel attack applications.

For software - oriented cryptographic targets, the primary comparative focus lies between the ID model and the HW model. The HW model, which categorizes intermediate cryptographic values into distinct classes according to the count of bits set to logic one, imposes a more generalized leakage assumption compared to the ID model—this generalization enhances its adaptability to the variable execution environments commonly associated with software—based cryptographic implementations. While the HW model demonstrates theoretical validity, this generalization gives rise to a notable practical limitation: severe label imbalance. The distribution of HW values conforms to a binomial distribution, which implies that certain classes exhibit substantially higher occurrence frequencies than others. This label imbalance poses challenges to model training: it may bias the model toward majority classes, impairing its ability to converge to a high-accuracy, robust solution across all cryptographic key hypotheses and undermining the deep-learning side-channel attack's effectiveness.

In contrast, the ID model has evolved into the predominant choice for state-of-the-art (SOTA) DL-SCAs. By treating each distinct intermediate cryptographic value as a unique class label, the ID model completely circumvents the label imbalance issue inherent to alternative leakage models. This approach formulates a classification task for the deep learning network that exhibits greater uniformity in label distribution. This methodological design enables the model to directly learn the nuanced correlation between the exact sensitive data values and their corresponding power consumption signals. Consequently, attacks utilizing the ID model consistently demonstrate faster convergence, superior generalization capability, and higher overall attack accuracy compared to those relying on the HW model, solidifying its position as the preferred leakage model for software implementations.

4.4 Unresolved Issues in DLSCAs

Unresolved issues in DLSCAs on AES often revolve around enhancing deep-learning models' generalization capability and interpretability. At present, There exist studies that employ deep-learning models as tools for segmenting traces across various victims, rather than as an attacking scheme. For instance, reference [116] puts forward an automated trace segmentation approach based on reinforcement learning that applies to many common implementations of public-key algorithms. Through experiments, they proved the transferability of the proposed framework on in excess of 10 datasets. Another unresolved challenge in DLSCA on AES is the lack of model interpretability. Deep learning models, particularly those involving complex architectures like CNNs or AEs, often function as black boxes, offering limited insights into the features they learn or the decision-making processes they follow. For example, while a CNN might effectively retrieve the secret key from power traces with jitter-based countermeasures [29], it remains unclear which specific features of traces contribute the most to the success. This lack of interpretability not only hinders the debugging and fine-tuning of models but also complicates the evaluation of countermeasures.

In the current research landscape, the evaluation indicator system for DLSCA remains unstandardized: due to differences in model architectures, dataset characteristics, and training strategies, core performance indicators like SR and PGE show significant result variability across different deep learning models, while practical indicators such as key recovery time and computational complexity lack unified, clear definitions in

existing literature. Additionally, experimental settings vary substantially across studies, collectively creating significant barriers to objective, quantitative cross-study comparisons of DLSCA performance.

As quantum computing progresses from theoretical constructs toward practical hybrid architectures, its potential influence on side-channel analysis and DLSCA frameworks is becoming increasingly significant. Quantum algorithms, such as Grover's search and variational quantum classifiers, could dramatically accelerate key recovery or leakage modeling tasks when integrated into classical deep learning pipelines. Conversely, quantum hardware itself introduces novel side-channel vectors—arising from decoherence dynamics, qubit control signals, and cryogenic interfaces—that may be exploitable using techniques conceptually similar to classical DLSCAs. This convergence raises new security concerns where quantum and classical leakage paths may interact, amplifying vulnerability in hybrid cryptographic environments. Consequently, future research on DLSCAs should consider quantum-aware threat models and hybrid analysis frameworks capable of evaluating cross-domain leakage between quantum and classical computation layers.

Next, we explore various potential mitigation strategies highlighted in the review.

4.5 Potential Mitigation Strategies

From the review, it is evident that countermeasures are effective in mitigating the impact of SCAs. In this section, we categorize potential mitigation strategies into three different levels: algorithmic, implementation, and physical levels.

Algorithm level. From the review, we can find that it is always more difficult for adversaries to compromise some protected version of AES (such as masking) compared to unprotected AES in the same implementation. For example, Aysu et al. [129] propose a lightweight yet effective countermeasure, which leverages the specific opportunities of Binary Ring-Learning With Errors (B-RLWE) and is based on the randomization of intermediate states and masked threshold decoding. And Cui and Balasch [130] propose extending the RISC-V core with custom instructions to accelerate AES finite field arithmetic, thereby combating SCAs. At the same time, the reasonable configuration of the algorithm can also demonstrate extraordinary anti-side-channel capabilities in certain specific SCAs. For example, Barthe et al. [131] present a general method grounded in the concept of constant-time simulation to against cache-based timing attacks.

Implementation level. Another path to protect IoT edge devices from DLSCAs is to implement a cryptographic algorithm with interference. Techniques introducing noise and randomness have shown great effectiveness in this review to make the side-channel leakage hard to be detected. For example, Hardware implementations, in general, are more resistant to side-channel attacks than software implementations, since they are designed with fixed, streamlined circuits that minimize variations in power consumption, timing, and EM emissions. Unlike software, which relies on sequential execution of instructions that can leak information through observable patterns, hardware circuits can execute operations in parallel, making it harder to correlate side-channel information with secret data [95]. Meanwhile, due to the limitations of software circuits in hardware devices, their functional implementation lags behind that of hardware-based SCA countermeasures. For SCA-related issues that existing hardware cannot address with software algorithms, new approaches emerge in hardware design. For example, Bhandari et al. introduce LiCSPA, a novel countermeasure strategy addressing a critical threat to cryptographic hardware in modern technology nodes [132]. Implementation-level countermeasures that introduce noise are another effective strategy for mitigating the impact of SCAs. For instance, the AES duplication technique proposed in [133] ensures that a duplicated block within the device generates algorithmic noise closely resembling the power profile of the primary block and dependent on its input. This approach adds leakage-like noise to the side-channel traces, effectively complicating the task for adversaries by making it more challenging to distinguish actual leakage

from the noise. For the current mainstream SCA methods based on deep learning, the introduction of noise will make the target features less obvious, thereby reducing the attacker's attack capability.

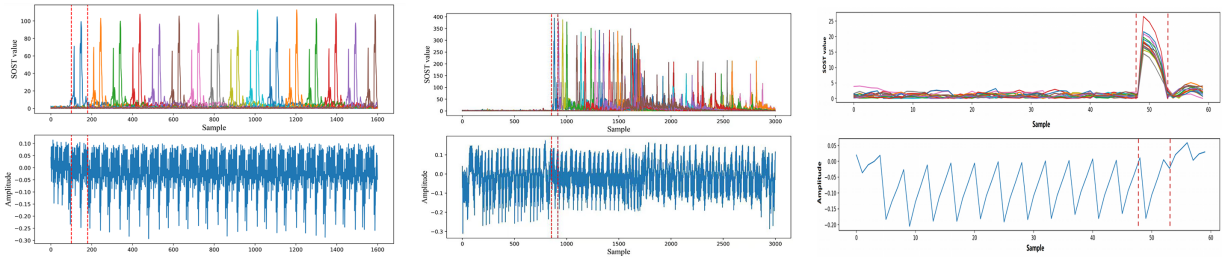
Physical level. The final suggested potential mitigation strategy in this paper is to directly utilize some other physical products to stop or slow down adversaries to capture side-channel measurements. As demonstrated in [134], placing a hidden camera behind a door or in an adjacent room significantly complicates detection via EM radiation. Reference [135] investigates the influence of physical countermeasures on side-channel vulnerability through an experimental setup featuring a PCB-based protective enclosure designed to shield the integrated circuit from electromagnetic and power analysis attacks. The experimental results demonstrate that the side-channel countermeasures at the physical layer will have a negative impact on the attack effect of the attacker.

Effective countermeasures in these areas require continuous research and development to keep pace with the evolving sophistication of attack techniques. From the review, it is evident that for the security-critical applications, conducting multi-level countermeasure that integrates algorithmic, physical, and implementation approaches is essential.

4.6 Impact of Microcontrollers on Leakage Characteristics

The effectiveness of a side-channel attack is profoundly influenced by the underlying hardware architecture executing the cryptographic algorithm. A critical, often overlooked, factor in profiling attacks is the correlation between the microcontroller's data path width and the resulting leakage model.

In 8-bit architectures, operations are inherently byte-oriented. When a 128-bit AES key is processed, each 8-bit subkey is typically manipulated in a separate instruction cycle. This high locality and specificity of leakage make 8-bit MCUs comparatively more vulnerable to SCAs, as the attacker can easily model and target the leakage of individual key bytes. In contrast, 32-bit architectures like the ARM Cortex-M series introduce significant complexity. These processors possess a 32-bit data bus and can perform word-level operations. Consequently, a single instruction might process up to four AES key bytes simultaneously. This parallelism causes the leakage of multiple subkeys to overlap in the time domain, creating a composite leakage signal that is much harder to dissect. The attacker's model, which often assumes leakage from a single small part of the state, becomes less accurate. As illustrated in the Fig. 11a presents the *t*-test results of the XMega dataset, while Fig. 11b depicts the *t*-test results of the STM32 dataset. It can be observed from the results that the side-channel leakage of the XMega dataset is prominent with a relatively sparse distribution; in contrast, the side-channel leakage of the data collected from the STM32 platform exhibits a more compact distribution, rendering it more difficult to conduct SCA.



(a) *t*-test on traces from ATxmega128D4.

(b) *t*-test on traces from STM32F3.

(c) *t*-test on traces from Atrix-7 FPGA.

Figure 11: Graphs presents test results for AES-128 implementations across differenton diverse MCU architectures are presented: the first figure corresponds to ATxmega128D4 microcontroller, the second forto STM32F3 MCU, and the third forto Artix-7 FPGA, with side-channel leakage trace examples proviincluded for each architecture

Furthermore, in an FPGA, the cryptographic algorithm is often fully or partially unrolled and deeply pipelined. Different operations on multiple blocks of data occur concurrently across the chip's fabric. The side-channel signal becomes an aggregate of numerous concurrent switching activities, making the contribution of any specific operation extremely weak and noisy. Therefore, attacking a well-designed FPGA implementation generally requires orders of magnitude more traces and more advanced analysis techniques compared to attacking a software implementation on a sequential MCU.

In summary, the attack complexity escalates from the byte-level serial processing of 8-bit AVR, through the word-level parallelism of 32-bit ARM cores, to the massive spatial and temporal concurrency of FPGAs. Any comprehensive SCA study must account for these architectural differences when selecting attack strategies and setting performance expectations.

5 Conclusions

This paper offers a thorough review of different deep learning methods used in side-channel attacks on AES implementations. We begin with an introduction to deep learning-based side-channel attacks, followed by an extensive comparison of diverse deep learning techniques. Subsequently, we conduct an extensive review of relevant research works, organizing them based on the targeted implementations and utilized side channels. The findings and steps of each approach are reported in detail. In summary, this study highlights the growing concerns and potential risks associated with the use of deep learning in side-channel attacks. Although deep learning offers powerful capabilities in extracting and analyzing side-channel information, it also faces challenges such as the requirement for substantial volumes of training data, vulnerability to adversarial attacks, and the potential for overfitting.

While real-world applications of DLSCAs are rarely reported, their potential to threaten cryptographic system security is well recognized through research and proof-of-concept demonstrations. With the ongoing advancement of the deep learning field, it becomes imperative for researchers and practitioners to develop robust countermeasures and defense strategies to mitigate the risks associated with these attacks. This survey functions as a valuable resource to understand the current state of DLSCAs, providing information on their capabilities, challenges, and potential avenues for future research.

6 Future Works

Future research about DLSCAs on AES in IoT environments should address critical challenges to enhance their effectiveness and adaptability. A key direction involves improving model generalization to account for real-world variability. Current studies often rely on controlled datasets, but IoT devices operate in dynamic conditions with noise and interference. Techniques such as transfer learning, domain adaptation, and the creation of more diverse datasets could improve the robustness of the model in diverse operational environments.

Another crucial issue is the interpretability of deep learning models in SCA tasks. Despite their high performance, these models often serve as "black boxes," offering little understanding of the features they exploit. Research should focus on applying explainability techniques, such as feature attribution or visualizing model outputs, to identify key characteristics in side-channel traces. Developing inherently interpretable architectures tailored for side-channel analysis would also advance this area.

Lastly, the most important future work is the development of countermeasures. Techniques such as running-time anomaly detection, adaptive cryptographic mechanisms, and active interference sources could significantly mitigate risks. Collaboration between academia and industry will be essential to translate these strategies into practical tools to secure IoT systems.

The development of DLSCAs raises ethical concerns, as the same research that strengthens cybersecurity can also be exploited by malicious actors. While exploiting how DLSCAs can pose a threat to cryptographic implementations is essential for evaluating vulnerabilities of physical devices, publicly available shared DLSCA techniques' details may inspire adversaries' malicious activities. However, without exploring advanced attack techniques and case studies, it is difficult to keep up-to-date on the capacity of these malicious actions. An essential approach to balancing ethical considerations with academic requirements in DLSCA research involves restricting the experimental setup to a controlled environment and a predefined scope of targets.

Acknowledgement: The authors would like to acknowledge the support from the Key R&D Program of Hunan Province (Grant No. 2025AQ2024) and Distinguished Young Scientists Fund (Grant No. 24B0446) for providing the research environment. Additionally, we extend our gratitude to the researchers who have made their datasets publicly available, enabling comprehensive comparative analysis in this field.

Funding Statement: This work is supported by the following research grants.
1. The Key R&D Program of Hunan Province (Grant No. 2025AQ2024) of the Department of Science and Technology of Hunan Province.
2. Distinguished Young Scientists Fund (Grant No. 24B0446) of Hunan Education Department.

Author Contributions: Junnian Wang conceptualized the study framework and methodology, conducted the literature analysis and synthesis and wrote the original draft. Xiaoxia Wang performed systematic literature investigation data extraction and validation and contributed to manuscript structuring. Zexin Luo provided critical analysis of research trends and contributed to manuscript review and editing. Qixiang Ouyang supervised project progress and contributed to manuscript review and editing. Chao Zhou conducted comparative analysis of methodologies and contributed to results interpretation and manuscript revision. Huanyu Wang as the corresponding author oversaw project coordination provided overall supervision, provided resources and led the manuscript finalization process. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: There are no data generated in this manuscript.

Ethics Approval: The manuscript does not have any ethical problem.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Admass WS, Munaye YY, Diro A. Cyber security: state of the art, challenges and future directions. Cyber Secur Appl. 2023;2(1):100031. doi:10.1016/j.csa.2023.100031.
2. Liu C, Chakraborty A, Chawla N, Roggel N. Frequency throttling side-channel attack. In: ACM SIGSAC Conference on Computer and Communications Security (CCS); 2022 Nov 7–11; Los Angeles, CA, USA.
3. Tosun T, Savas E. Zero-value filtering for accelerating non-profiled side-channel attack on incomplete NTT based implementations of lattice-based cryptography. IEEE Trans Inf Forensics Secur (TIFS). 2024;19(3):3353–65. doi:10.1109/tifs.2024.3359890.
4. Gao Y, Qiu H, Zhang Z, Wang B, Ma H, Abuadbba A, et al. DeepTheft: stealing DNN model architectures through power side channel. In: IEEE Symposium on Security and Privacy (SP). Piscataway, NJ, USA: IEEE; 2024. p. 3311–26.
5. Batina L, Bhasin S, Jap D, Picek S. CSI NN: reverse engineering of neural network architectures through electromagnetic side channel. In: USENIX Security Symposium (USENIX Security); 2019 Aug 14–16; Santa Clara, CA, USA. p. 515–32.
6. Moradi A, Schneider T. Improved side-channel analysis attacks on Xilinx bitstream encryption of 5, 6, and 7 series. In: International Workshop on Constructive Side-Channel Analysis and Secure Design. Cham, Switzerland: Springer; 2016. p. 71–87. doi:10.1007/978-3-319-43283-0_5.

7. Wang Y, Paccagnella R, Gang Z, Vasquez WR, Kohlbrenner D, Shacham H, et al. GPU. zip: on the side-channel implications of hardware-based graphical data compression. In: 2024 IEEE Symposium on Security and Privacy (SP); 2024 May 19–23; San Francisco, CA, USA. p. 3716–34.
8. Yang Q, Gasti P, Zhou G, Farajidavar A, Balagani KS. On inferring browsing activity on smartphones via USB power analysis side-channel. IEEE Trans Inf Forensics Secur (TIFS). 2016;12(5):1056–66. doi:10.1109/tifs.2016.2639446.
9. Yu Y, Moraitis M, Dubrova E. Why deep learning makes it difficult to keep secrets in FPGAs. In: Workshop in Dynamic and Novel Advances in Machine Learning and Intelligent Cyber Security; 2020 Dec 7; Lexington, MA, USA. p. 1–9. doi:10.1145/3477997.3478001.
10. He D, Wang H, Deng T, Liu J, Wang J. Improving IIoT security: unveiling threats through advanced side-channel analysis. Comput Secur. 2024;148(1):104135. doi:10.1016/j.cose.2024.104135.
11. Liu Y, Wei L, Zhou Z, Zhang K, Xu W, Xu Q. On code execution tracking via power side-channel. In: ACM SIGSAC Conference on Computer and Communications Security (CCS). New York, NY, USA: ACM; 2016. p. 1019–31.
12. Hu J, Wang H, Zheng T, Hu J, Chen Z, Jiang H, et al. Password-stealing without hacking: Wi-Fi enabled practical keystroke eavesdropping. In: ACM SIGSAC Conference on Computer and Communications Security (CCS). New York, NY, USA: ACM; 2023. p. 239–52.
13. Cardaioli M, Conti M, Balagani K, Gasti P. Your pin sounds good! Augmentation of pin guessing strategies via audio leakage. In: European Symposium on Research in Computer Security (ESORICS). Cham, Switzerland: Springer; 2020. p. 720–35.
14. Ni T, Zhang X, Zhao Q. Recovering fingerprints from in-display fingerprint sensors via electromagnetic side channel. In: ACM SIGSAC Conference on Computer and Communications Security (CCS). New York, NY, USA: ACM; 2023. p. 253–67.
15. Camurati G, Poeplau S, Muench M, Hayes T, Francillon A. Screaming channels: when electromagnetic side channels meet radio transceivers. In: ACM SIGSAC Conference on Computer and Communications Security (CCS). New York, NY, USA: ACM; 2018. p. 163–77.
16. Wang R, Wang H, Dubrova E, Brisfors M. Advanced far field EM side-channel attack on AES. In: ACM Cyber-Physical System Security Workshop (CPSS). New York, NY, USA: ACM; 2021. p. 29–39. doi:10.1145/3411504.3421214.
17. Wang H. Amplitude-modulated EM side-channel attack on provably secure masked AES. J Cryptogr Eng. 2024;14(3):537–49. doi:10.1007/s13389-024-00347-3.
18. Kocher P, Jaffe J, Jun B. Differential power analysis. In: Annual International Cryptology Conference. Cham, Switzerland: Springer; 1999. p. 388–97.
19. Kocher PC. Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems. In: Annual International Cryptology Conference. Cham, Switzerland: Springer; 1996. p. 104–13.
20. Nassi B, Vayner O, Iluz E, Nassi D, Jancar J, Genkin D, et al. Optical cryptanalysis: recovering cryptographic keys from power LED light fluctuations. In: ACM SIGSAC Conference on Computer and Communications Security (CCS). New York, NY, USA: ACM; 2023. p. 268–80.
21. Genkin D, Shamir A, Tromer E. Acoustic cryptanalysis. J Cryptol. 2017;30(2):392–443. doi:10.1007/s00145-015-9224-2.
22. Agrawal D, Archambeault B, Rao JR, Rohatgi P. The EM side—channel(s). In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Cham, Switzerland: Springer; 2002. p. 29–45.
23. Bayoudh K. A survey of multimodal hybrid deep learning for computer vision: architectures, applications, trends, and challenges. Inf Fusion. 2024;105:102217. doi:10.1016/j.inffus.2023.102217.
24. Wu Y, Chen Z, Yao X, Chen X, Zhou Z, Xue J. JAN: joint attention networks for automatic ICD coding. IEEE J Biomed Health Inform. 2022;26(10):5235–46. doi:10.1109/jbhi.2022.3189404.
25. Hua H, Li Y, Wang T, Dong N, Li W, Cao J. Edge computing with artificial intelligence: a machine learning perspective. ACM Comput Surv. 2023;55(9):1–35. doi:10.1145/3555802.
26. Golightly L, Modesti P, Garcia R, Chang V. Securing distributed systems: a survey on access control techniques for cloud, blockchain, IoT and SDN. Cyber Secur Appl. 2023;1(7):100015. doi:10.1016/j.csa.2023.100015.

27. Brier E, Clavier C, Olivier F. Correlation power analysis with a leakage model. In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Cham, Switzerland: Springer; 2004. p. 16–29.
28. Chari S, Rao JR, Rohatgi P. Template attacks. In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Cham, Switzerland: Springer; 2002. p. 13–28.
29. Cagli E, Dumas C, Prouff E. Convolutional neural networks with data augmentation against jitter-based countermeasures. In: International Conference on Cryptographic Hardware and Embedded Systems (CHES). Cham, Switzerland: Springer; 2017. p. 45–68.
30. Randolph M, Diehl W. Power side-channel attack analysis: a review of 20 years of study for the layman. Cryptography. 2020;4(2):15. doi:10.3390/cryptography4020015.
31. Gierlichs B, Batina L, Tuyls P, Preneel B. Mutual information analysis: a generic side-channel distinguisher. In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Cham, Switzerland: Springer; 2008. p. 426–42.
32. Goodwill G, Jun B, Jaffe J, Rohatgi P. A testing methodology for side-channel resistance validation. In: NIST Non-Invasive Attack Testing Workshop; 2011 Sep 26–27; Nara, Japan; Vol. 7, p. 115–36.
33. Hettwer B, Gehrer S, Güneysu T. Applications of machine learning techniques in side-channel attacks: a survey. J Cryptogr Eng. 2020;10(2):135–62. doi:10.1007/s13389-019-00212-8.
34. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97. doi:10.1023/a:1022627411411.
35. Murthy SK. Automatic construction of decision trees from data: a multi-disciplinary survey. Data Min Knowl Discov. 1998;2(4):345–89. doi:10.1023/a:1009744630224.
36. Breiman L. Random forests. Mach Learn. 2001;45:5–32.
37. Picek S, Perin G, Mariot L, Wu L, Batina L. SoK: deep learning-based physical side-channel analysis. ACM Comput Surv. 2023;55(11):1–35. doi:10.1145/3569577.
38. Zunaidi MR, Sayakkara A, Scanlon M. Systematic literature review of EM-SCA attacks on encryption. arXiv:2402.10030. 2024.
39. Panoff M, Yu H, Shan H, Jin Y. A review and comparison of AI-enhanced side channel analysis. J Emerg Technol Comput Syst. 2022;18(3):62. doi:10.1145/3517810.
40. Naserelden S, Alias N, Altigani A, Mohamed A, Badreddine S. Advance attacks on AES: a comprehensive review of side channel, fault injection, machine learning and quantum techniques. Edelweiss Appl Sci Technol. 2025;9(4):2471–86. doi:10.55214/25768484.v9i4.6586.
41. Dobias P, Rezaeezade A, Chmielewski L, Malina L, Batina L. SoK: reassessing side-channel vulnerabilities and countermeasures in PQC implementations. Cryptology ePrint Archive; 2025. Paper 2025/1222. [cited 2025 Nov 2]. Available from: https://eprint.iacr.org/2025/1222.
42. Daemen J, Rijmen V. The design of Rijndael. Vol. 2. Cham, Switzerland: Springer; 2002.
43. Wu J. Introduction to convolutional neural networks. Nanjing, China: National Key Lab for Novel Software Technology Nanjing University China; 2017.
44. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Adv Neural Inf Process Syst. 2017;30:6000–10.
45. Hajra S, Saha S, Alam M, Mukhopadhyay D. TransNet: shift invariant transformer network for side channel analysis. In: International Conference on Cryptology in Africa. Cham, Switzerland: Springer; 2022. p. 371–96.
46. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. IEEE Trans Neural Netw. 2009;20(1):61–80. doi:10.1109/tnn.2008.2005605.
47. Abbas S, Ojo S, Bouazzi I, Sampedro GA, Al Hejaili A, Almadhor AS, et al. Securing data from side-channel attacks: a graph neural network-based approach for smartphone-based side channel attack detection. IEEE Access. 2024;12(70):138904–20. doi:10.1109/access.2024.3465662.
48. Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation. In: Parallel distributed processing: explorations in the microstructure of cognition: foundations. Cambridge, MA, USA: MIT Press; 1985. p. 318–62.
49. Yang G, Li H, Ming J, Zhou Y. CDAE: towards empowering denoising in side-channel analysis. In: International Conference on Information and Communications Security. Cham, Switzerland: Springer; 2019. p. 269–86.

50. He P, Zhang Y, Gan H, Ma J, Zhang H. Side-channel attacks based on attention mechanism and multi-scale convolutional neural network. Comput Electr Eng. 2024;119(2):109515. doi:10.1016/j.compeleceng.2024.109515.
51. Zhao Z. Far field electromagnetic side channel analysis of AES [master's thesis]. Stockholm, Sweden: KTH Royal Institute of Technology; 2020.
52. Timon B. Non-profiled deep learning-based side-channel attacks with sensitivity analysis. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2019;2019(2):107–31. doi:10.46586/tches.v2019.i2.107-131.
53. Ji Y, Wang R, Ngo K, Dubrova E, Backlund L. A side-channel attack on a hardware implementation of CRYSTALS-Kyber. In: IEEE European Test Symposium (ETS). Piscataway, NJ, USA: IEEE; 2023. p. 1–5.
54. Wang R, Dubrova E. A side-channel secret key recovery attack on CRYSTALS-kyber using k chosen ciphertexts. In: Codes, cryptology and information security. Cham, Switzerland: Springer; 2023. p. 109–28.
55. Ngo K, Wang R, Dubrova E. Higher-order boolean masking does not prevent side-channel attacks on LWE/LWR-based PKE/KEMs. In: International Symposium on Multiple-Valued Logic (ISMVL). Piscataway, NJ, USA: IEEE; 2023. p. 190–5.
56. Maghrebi H, Portigliatti T, Prouff E. Breaking cryptographic implementations using deep learning techniques. In: International Conference on Security, Privacy, and Applied Cryptography Engineering. Cham, Switzerland: Springer; 2016. p. 3–26.
57. Liu K. Far field EM side-channel attack based on deep learning with automated hyperparameter tuning [master's thesis]. Stockholm, Sweden: KTH Royal Institute of Technology; 2021.
58. Welch BL. The generalization of 'STUDENT'S problem when several different population varlances are involved. Biometrika. 1947;34(1–2):28–35. doi:10.1093/biomet/34.1-2.28.
59. Sim BY, Kang J, Han DG. Key bit-dependent side-channel attacks on protected binary scalar multiplication. Appl Sci. 2018;8(11):2168. doi:10.3390/app8112168.
60. Elaabid MA, Meynard O, Guilley S, Danger JL. Combined side-channel attacks. In: International Workshop on Information Security Applications. Cham, Switzerland: Springer; 2010. p. 175–90.
61. Standaert FX, Malkin TG, Yung M. A unified framework for the analysis of side-channel key recovery attacks. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Cham, Switzerland: Springer; 2009. p. 443–61.
62. Zhang J, Zheng M, Nan J, Hu H, Yu N. A novel evaluation metric for deep learning-based side channel analysis and its extended application to imbalanced data. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2020;2020(3):73–96. doi:10.46586/tches.v2020.i3.73-96.
63. Pahlevanzadeh H, Dofe J, Yu Q. Assessing CPA resistance of AES with different fault tolerance mechanisms. In: Asia and South Pacific Design Automation Conference (ASP-DAC). Piscataway, NJ, USA: IEEE; 2016. p. 661–6.
64. Brisfors M, Forsmark S, Dubrova E. How deep learning helps compromising USIM. In: International Conference on Smart Card Research and Advanced Applications. Cham, Switzerland: Springer; 2020. p. 135–50.
65. Hu F, Shen J, Vijayakumar P. Side-channel attacks based on multi-loss regularized denoising AutoEncoder. IEEE Trans Inf Foren Secur. 2024;19:2051–65. doi:10.1109/tifs.2023.3343947.
66. Mangard S, Oswald E, Popp T. Power analysis attacks: revealing the secrets of smart cards. Vol. 31. Cham, Switzerland: Springer Science & Business Media; 2008.
67. Koeune F, Standaert FX. A tutorial on physical security and side-channel attacks. In: Foundations of security analysis and design III (FOSAD 2005, FOSAD 2004). Berlin/Heidelberg, Germany: Springer; 2004. p. 78–108.
68. Bhasin S, Danger JL, Guilley S, Najm Z. A low-entropy first-degree secure provable masking scheme for resource-constrained devices. In: WESS '13: Proceedings of the Workshop on Embedded Systems Security. New York, NY, USA: ACM; 2013. p. 1–10.
69. Coron JS, Kizhvatov I. An efficient method for random delay generation in embedded software. In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Berlin/Heidelberg, Germany: Springer; 2009. p. 156–70.
70. Bogdanov A, Khovratovich D, Rechberger C. Biclique cryptanalysis of the full AES. In: International Conference on the Theory and Application of Cryptology and Information Security. Berlin/Heidelberg, Germany: Springer; 2011. p. 344–71.

71. Zorba BB, Alkar AZ, Aydos M, Kolukisa-Tarhan A. Software implementation performances of block ciphers: a systematic literature review. In: International Workshop on Big Data and Information Security (IWBIS). Piscataway, NJ, USA: IEEE; 2022. p. 65–74.
72. Dworkin M. Recommendation for block cipher modes of operation: galois/counter mode (GCM) and GMAC. Gaithersburg, MD, USA: NIST Special Publication; 2007.
73. Suykens JA, Vandewalle J. Least squares support vector machine classifiers. Neural Process Lett. 1999;9(3):293–300. doi:10.1023/a:1018628609742.
74. Hospodar G, Gierlichs B, De Mulder E, Verbauwhede I, Vandewalle J. Machine learning in side-channel analysis: a first study. J Cryptographic Eng. 2011;1(4):293. doi:10.1007/s13389-011-0023-x.
75. Martinasek Z, Zeman V. Innovative method of the power analysis. Radioengineering. 2013;22(2):586–94.
76. Martinasek Z, Hajny J, Malina L. Optimization of power analysis using neural network. In: International Conference on Smart Card Research and Advanced Applications. Berlin/Heidelberg, Germany: Springer; 2013. p. 94–107.
77. Martinasek Z, Malina L, Trasy K. Profiling power analysis attack based on multi-layer perceptron network. In: Computational problems in science and engineering. Berlin/Heidelberg, Germany: Springer; 2015. p. 317–39.
78. Wong SC, Gatt A, Stamatescu V, McDonnell MD. Understanding data augmentation for classification: when to warp? In: International Conference on Digital Image Computing: Techniques and Applications (DICTA). Piscataway, NJ, USA: IEEE; 2016. p. 1–6.
79. Picek S, Samiotis IP, Kim J, Heuser A, Bhasin S, Legay A. On the performance of convolutional neural networks for side-channel analysis. In: International Conference on Security, Privacy, and Applied Cryptographic Engineering. Berlin/Heidelberg, Germany: Springer; 2018. p. 157–76.
80. Kim J, Picek S, Heuser A, Bhasin S, Hanjalic A. Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2019;2019(3):148–79. doi:10.46586/tches.v2019.i3.148-179.
81. Picek S, Heuser A, Jovic A, Bhasin S, Regazzoni F. The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2018;2019(1):209–37. doi:10.46586/tches.v2019.i1.209-237.
82. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intelligence Res. 2002;16:321–57. doi:10.1613/jair.953.
83. Jin M, Zheng M, Hu H, Yu N. An enhanced convolutional neural network in side-channel attacks and its visualization. arXiv:2009.08898. 2020.
84. Wang H, Forsmark S, Brisfors M, Dubrova E. Multi-source training deep-learning side-channel attacks. In: IEEE International Symposium on Multiple-Valued Logic (ISMVL). Piscataway, NJ, USA: IEEE; 2020. p. 58–63.
85. Zaid G, Bossuet L, Habrard A, Venelli A. Methodology for efficient CNN architectures in profiling attacks. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2020;2020(1):1–36. doi:10.46586/tches.v2020.i1.1-36.
86. Huang H, Wu J, Tang X, Zhao S, Liu Z, Yu B. Deep learning-based improved side-channel attacks using data denoising and feature fusion. PLoS One. 2025;20(4):e0315340. doi:10.1371/journal.pone.0315340.
87. Ou Y, Wei Y, Rodríguez-Aldama R, Zhang F. A lightweight deep learning model for profiled SCA based on random convolution kernels. Information. 2025;16(5):351. doi:10.3390/info16050351.
88. Zhang L, Xing X, Fan J, Wang Z, Wang S. Multi-label deep learning based side channel attack. In: Asian Hardware Oriented Security and Trust Symposium (AsianHOST). Piscataway, NJ, USA: IEEE; 2019. p. 1–6.
89. Wu L, Picek S. Remove some noise: on pre-processing of side-channel measurements with autoencoders. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2020;2020(4):389–415. doi:10.46586/tches.v2020.i4.389-415.
90. Zaid G, Bossuet L, Carbone M, Habrard A, Venelli A. Conditional variational autoencoder based on stochastic attacks. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2023;2023(2):310–57. doi:10.46586/tches.v2023.i2.310-357.
91. Wu L, Weissbart L, Krček M, Li H, Perin G, Batina L, et al. Label correlation in deep learning-based side-channel analysis. IEEE Trans Inf Foren Secur (TIFS). 2023;18:3849–61. doi:10.1109/tifs.2023.3287728.

92. Yao F, Wei Y, Chen H, Pasalic E. Improving the performance of CPA attacks for ciphers using parallel implementation of S-boxes. IET Inf Secur. 2023;2023(1):6653956. doi:10.1049/2023/6653956.
93. Zhang X, Zhu Y, Hu B, Cao J, Lin Z. A novel power system side channel attack method based on machine learning CNN-transformer. J Phys Conf Ser. 2023;2615(1):012011. doi:10.1088/1742-6596/2615/1/012011.
94. Zhu J, Lu J. Leading degree: a metric for model performance evaluation and hyperparameter tuning in deep learning-based side-channel analysis. IACR Trans Cryptogr Hardw Embed Syst. 2025;2025(2):333–61. doi:10.46586/tches.v2025.i2.333-361.
95. Wang H. Deep learning side-channel attacks on advanced encryption standard [dissertation]. Stockholm, Sweden: KTH Royal Institute of Technology; 2023.
96. Benadjila R, Prouff E, Strullu R, Cagli E, Dumas C. Deep learning for side-channel analysis and introduction to ASCAD database. J Cryptogr Eng. 2020;10(2):163–88. doi:10.1007/s13389-019-00220-8.
97. Wang H, Brisfors M, Forsmark S, Dubrova E. How diversity affects deep-learning side-channel attacks. In: IEEE Nordic Circuits and Systems Conference (NORCAS). Piscataway, NJ, USA: IEEE; 2019. p. 1–7.
98. Wang R, Wang H, Dubrova E. Far field EM side-channel attack on AES using deep learning. In: ACM Workshop on Attacks and Solutions in Hardware Security (ASHES). New York, NY, USA: ACM; 2020. p. 35–44.
99. Rijsdijk J, Wu L, Perin G, Picek S. Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2021;2021(3):677–707. doi:10.46586/tches.v2021.i3.677-707.
100. Wu L, Perin G, Picek S. I choose you: automated hyperparameter tuning for deep learning-based side-channel analysis. IEEE Trans Emerg Top Comput. 2024;12(2):546–57. doi:10.1109/tetc.2022.3218372.
101. Feng T, Gao H, Li X, Liu C. Side-channel attacks on convolutional neural networks based on the hybrid attention mechanism. Discov Appl Sci. 2025;7(5):1–15. doi:10.1007/s42452-025-06854-0.
102. Wang H, Dubrova E. Tandem deep learning side-channel attack on FPGA implementation of AES. Vol. 2. Berlin/Heidelberg, Germany: Springer; 2021. p. 1–12.
103. Bischof H, Pinz A, Kropatsch WG. Visualization methods for neural networks. In: IAPR International Conference on Pattern Recognition. Piscataway, NJ, USA: IEEE; 1992. p. 581–5.
104. Masure L, Dumas C, Prouff E. Gradient visualization for general characterization in profiling attacks. In: International Workshop on Constructive Side-Channel Analysis and Secure Design. Berlin/Heidelberg, Germany: Springer; 2019. p. 145–67.
105. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European Conference on Computer Vision. Berlin/Heidelberg, Germany: Springer; 2014. p. 818–33.
106. Wang H, Feng T, Liu C. Wavelet coefficients based generative adversarial networks for side-channel attack preprocessing. Signal Image Video Process. 2025;19(9):697. doi:10.1007/s11760-025-04272-8.
107. Sutton RS, Barto AG. Reinforcement learning: an introduction. Cambridge, MA, USA: MIT press; 2018.
108. de Bruijn V. Side-channel analysis with graph neural networks [master's thesis]. Delft, The Netherlands: Delft University of Technology; 2021.
109. Das D, Golder A, Danial J, Ghosh S, Raychowdhury A, Sen S. X-DeepSCA: cross-device deep learning side channel attack. In: 2019 56th ACM/IEEE Design Automation Conference (DAC). Piscataway, NJ, USA: IEEE; 2019. p. 1–6.
110. Golder A, Das D, Danial J, Ghosh S, Sen S, Raychowdhury A. Practical approaches toward deep-learning-based cross-device power side-channel attack. IEEE Trans Very Large Scale Integr Syst. 2019;27(12):2720–33. doi:10.1109/tvlsi.2019.2926324.
111. Wang H, Dubrova E. Federated learning in side-channel analysis. In: Information Security and Cryptology-ICISC 2020: 23rd International Conference. Berlin/Heidelberg, Germany: Springer; 2021. p. 257–72.
112. Hu F, Wang H, Wang J. Cross subkey SCA based on small samples. Sci Rep. 2022;12(1):6254.
113. Rezaeezade A, Yap T, Jap D, Bhasin S, Picek S. Breaking the blindfold: deep learning-based blind side-channel analysis. Cryptology ePrint Archive; 2025. Paper 2025/157. [cited 2025 Nov 2]. Available from: https://eprint.iacr.org/2025/157.

114. Ninan M, Nimmo E, Reilly S, Smith C, Sun W, Wang B, et al. A second look at the portability of deep learning side-channel attacks over EM traces. [cited 2025 Nov 2]. Available from: https://par.nsf.gov/biblio/10566521.
115. Liao J, Wang H, Wang J, Tang Y. Switch-T: a novel multi-task deep-learning network for cross-device side-channel attack. J Inf Secur Appl. 2025;93(1):104146. doi:10.1016/j.jisa.2025.104146.
116. Wang Z, Ding Y, Wang A, Zhang Y, Wei C, Sun S, et al. SPA-GPT: general pulse tailor for simple power analysis based on reinforcement learning. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2024;2024(4):40–83. doi:10.46586/tches.v2024.i4.40-83.
117. Chen Y, Wang B, Su C, Li A, Tang Y, Li G. Enhancing model generalization for efficient cross-device side-channel analysis. IEEE Trans Inf Foren Secur. 2025;20(2):10114–29. doi:10.1109/tifs.2025.3611696.
118. Lv H, Zheng Z, Luo T, Wu F, Tang S, Hua L, et al. Data-free evaluation of user contributions in federated learning. In: International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks. Piscataway, NJ, USA: IEEE; 2021. p. 1–8.
119. McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics. London, UK: PMLR; 2017. p. 1273–82.
120. Liu Z, Lv H, Liu X, Ma C, Wu F, Liu L, et al. Similarity and diversity: PCA-based contribution evaluation in federated learning. IEEE Internet Things J. 2025;12(12):20393–405. doi:10.1109/jiot.2025.3546679.
121. Polikar R. Ensemble learning. In: Ensemble machine learning. Berlin/Heidelberg, Germany: Springer; 2012. p. 1–34.
122. Wu L, Won YS, Jap D, Perin G, Bhasin S, Picek S. Ablation analysis for multi-device deep learning-based physical side-channel analysis. IEEE Trans Depend Secure Comput. 2024;21(3):1331–41. doi:10.1109/tdsc.2023.3278857.
123. Priya SSS, Karthigaikumar P, Teja NR. FPGA implementation of AES algorithm for high speed applications. Analog Integr Circuits Signal Process. 2022;112(1):115–25. doi:10.1007/s10470-021-01959-z.
124. Lee U, Kim HK, Lim YJ, Sunwoo MH. Resource-efficient FPGA implementation of advanced encryption standard. In: IEEE International Symposium on Circuits and Systems (ISCAS). Piscataway, NJ, USA: IEEE; 2022. p. 1165–9.
125. TELECOM ParisTech SEN research group T. DPA contest v2. 2010 [Internet]. [cited 2025 Nov 2]. Available from: http://www.DPAcontest.org/v2/.
126. Masure L, Dumas C, Prouff E. A comprehensive study of deep learning for side-channel analysis. IACR Trans Cryptogr Hardw Embed Syst (TCHES). 2020;2020(1):348–75. doi:10.46586/tches.v2020.i1.348-375.
127. Ramezanpour K, Ampadu P, Diehl W. SCAUL: power side-channel analysis with unsupervised learning. IEEE Trans Comput. 2020;69(11):1626–38. doi:10.1109/tc.2020.3013196.
128. Kubota T, Yoshida K, Shiozaki M, Fujino T. Deep learning side-channel attack against hardware implementations of AES. In: Euromicro Conference on Digital System Design. Piscataway, NJ, USA: IEEE; 2019. p. 261–8.
129. Aysu A, Orshansky M, Tiwari M. Binary ring-LWE hardware with power side-channel countermeasures. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway, NJ, USA: IEEE; 2018. p. 1253–8.
130. Cui S, Balasch J. Efficient software masking of AES through instruction set extensions. In: 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway, NJ, USA: IEEE; 2023. p. 1–6.
131. Barthe G, Grégoire B, Laporte V. Secure compilation of side-channel countermeasures: the case of cryptographic "constant-time". In: 2018 IEEE 31st Computer Security Foundations Symposium (CSF). Piscataway, NJ, USA: IEEE; 2018. p. 328–43.
132. Bhandari J, Nabeel M, Mankali L, Sinanoglu O, Karri R, Knechtel J. LiCSPA: lightweight countermeasure against static power side-channel attacks. In: 2025 IEEE International Symposium on Circuits and Systems (ISCAS). Piscataway, NJ, USA: IEEE; 2025. p. 1–5.
133. Moraitis M, Brisfors M, Dubrova E, Lindskog N, Englund H. A side-channel resistant implementation of AES combining clock randomization with duplication. In: IEEE International Symposium on Circuits and Systems (ISCAS). Piscataway, NJ, USA: IEEE; 2023. p. 1–5.

134. Zhang Q, Liu D, Zhang X, Cao Z, Zeng F, Jiang H, et al. Eye of sauron: long-range hidden spy camera detection and positioning with inbuilt memory EM radiation. In: USENIX Security Symposium (USENIX Security). New York, NY, USA: ACM; 2024. p. 109–26.
135. Chen P, Li J, Cheng W, Cheng C. Uncover secrets through the cover: a deep learning-based side-channel attack against Kyber implementations with anti-tampering covers. IEEE Trans Comput. 2025;74(6):2159–67. doi:10.1109/tc.2025.3547610.