Cambodia Academy of Digital Technology

Daegu Gyeongbuk Institute of Science & Technology

Deep Learning-Based Side-Channel Analysis for the CHES 2025 Challenge

A Technical Report on Advanced Key Recovery Techniques

Lay Sopanha

A report submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science

Internship at:
Privacy and Applied Cryptography Lab (PACL)
Daegu Gyeongbuk Institute of Science and Technology (DGIST)
Daegu, South Korea

Supervisor:
Mr. Cho Seunghyun

Under the supervision of:
Prof. Young-Sik Kim, Head of PACL

Internship Period:
June 30, 2025 – August 8, 2025

Contents

List of Figures

List of Tables

List of Abbreviations

AES	Advanced Encryption Standard
API	Application Programming Interface
CHES	Conference on Cryptographic Hardware and Embedded Systems
CNN	Convolutional Neural Network
CPA	Correlation Power Analysis
DL	Deep Learning
GE	Guessing Entropy
GPU	Graphics Processing Unit
HW	Hamming Weight
ID	Identity (Leakage Model)
ML	Machine Learning
MLOps	Machine Learning Operations
NTGE	Number of Traces to Guessing Entropy
PACL	Privacy and Applied Cryptography Lab
POI	Point of Interest
SCA	Side-Channel Analysis
SNR	Signal-to-Noise Ratio

Acknowledgment

I would like to express my sincere gratitude to the individuals and institutions whose guidance, support, and encouragement were instrumental in the successful completion of this internship and research project.

First and foremost, I extend my deepest appreciation to my home institution, the **Cambodia Academy of Digital Technology (CADT)**. Their academic program provided me with the foundational knowledge necessary to undertake this international research opportunity, and their support was essential in making this experience a reality.

My heartfelt thanks go to the **Daegu Gyeongbuk Institute of Science and Technology (DGIST)** for hosting this internship. The welcoming and productive environment provided by the faculty and staff made my time in South Korea both educational and memorable.

I am particularly indebted to the members of the **Privacy and Applied Cryptography Lab (PACL)** for their mentorship and collaboration. I owe a special debt of gratitude to my primary supervisor, **Professor Young-Sik Kim**, whose invaluable guidance and expert oversight provided a clear direction for this project. My sincere thanks also go to my dedicated mentor, **Mr. Seunghyun Cho**, for his consistent support, insightful feedback, and for being my main point of contact throughout the research process.

I would also like to acknowledge my secondary mentors, **Mr. Jae Ho Jeon** and **Miss Yujin Seo**, for their constant assistance and for always being available to answer questions and provide constant support. The engaging discussions and technical advice from all members of the PACL team created a stimulating atmosphere that greatly enriched my learning experience.

Finally, I wish to thank my family for their unwavering love, belief, and encouragement, which have been the foundation of my perseverance and success.

To everyone who contributed to this endeavor, both named and unnamed, I offer my deepest appreciation.

Abstract

This report details the design, implementation, and evaluation of a state-of-the-art deep learning pipeline for side-channel analysis, developed as part of a research internship at the Privacy and Applied Cryptography Lab (PACL), DGIST, for the CHES 2025 Challenge. The project addresses the significant challenge of recovering cryptographic keys from modern, high-performance hardware, which is characterized by high noise, temporal misalignment, and vast data dimensionality.

The core of the project is a sophisticated methodology that integrates several advanced techniques. A Signal-to-Noise Ratio (SNR) based algorithm is first employed for intelligent Point of Interest (POI) selection, achieving a 96.4% dimensionality reduction of the raw power traces while preserving critical leakage information. The system's hyperparameters were systematically optimized using a Bayesian search strategy, guided by a novel "Composite Score" metric that directly targets cryptanalytic success rather than traditional classification accuracy. The resulting architecture is a precisely tuned 1D Convolutional Neural Network (CNN), trained using an innovative attack-driven paradigm that integrates full key recovery evaluation into the training loop.

The developed pipeline demonstrated exceptional performance on the official CHES 2025 dataset. It consistently achieved the primary objective of perfect key recovery, with a final **Guessing Entropy (GE) of 0.0**. The attack proved highly efficient, requiring as few as 3,879 traces to succeed (NTGE) and securing a competitive rank of **28th out of 110 submissions**. This work validates the overwhelming effectiveness of an end-to-end, MLOps-driven approach for tackling real-world hardware security challenges and establishes a robust blueprint for future research in deep learning-based cryptanalysis.

I. Introduction

This report provides a detailed overview of the research internship conducted from June 30 to August 8, 2025 at the Privacy and Applied Cryptography Lab (PACL), hosted within the Daegu Gyeongbuk Institute of Science and Technology (DGIST) in Daegu, South Korea. The core of this internship was a project focused on developing and implementing a state-of-the-art deep learning pipeline for side-channel analysis, with the practical goal of competing in the CHES 2025 Challenge.

The following sections will detail the project objectives, the organizational context, the methodologies employed, and the significant results achieved throughout this research endeavor.

1.1 Presentation of Internship

As part of the curriculum for the Bachelor of Science in Computer Science, this internship represents a hand-on research experience. Functioning as a Junior Researcher at PACL, my work involved a deep dive into the practical application of machine learning for hardware security. My responsibilities encompassed the entire research lifecycle, from a comprehensive literature review and dataset analysis to the design, implementation, hyperparameter optimization, and evaluation of a novel attack pipeline aimed at cryptographic key recovery.

This hands-on project allowed for the direct application of theoretical knowledge in deep learning and cryptography to a real-world, competitive research problem, strengthening both my analytical and technical problem-solving skills in a high-stakes environment.

1.2 Objective of Internship

The objectives of this internship were twofold, gaining both rigorous technical training and invaluable personal and professional development. Not just for hard skill it also give ways to enrich the soft skill as well.

1.2.1 Technical Objectives

The primary technical goals were centered on the end-to-end development of a real-world machine learning research project:

- **Apply Advanced Techniques:** To apply theoretical knowledge of deep learning and computer vision to solve the practical challenge of side-channel analysis on modern cryptographic hardware.

- **Master MLOps Workflow:** To gain proficiency in modern Machine Learning Operations (MLOps) tools and cloud-based platforms, specifically using Weights & Biases for experiment tracking and Bayesian hyperparameter optimization.
- **Develop Research Skills:** To develop critical problem-solving and research skills by navigating the technical complexities of noisy, high-dimensional data and adapting project strategies to achieve optimal outcomes.
- **Contribute to Competitive Research:** To successfully design and execute a submission for the CHES 2025 side-channel analysis challenge, aiming for state-of-the-art performance in key recovery.

1.2.2 Personal and Professional Development

Beyond the technical scope, a significant objective of this international internship was to foster personal and professional growth through cultural immersion and real-world experience:

- **Develop Cross-Cultural Competency:** To work effectively within an international research environment at PACL, collaborating with peers and researchers from diverse backgrounds and improving cross-cultural communication skills.
- **Foster Adaptability and Independence:** To navigate the challenges of living and working in a new country, South Korea, thereby enhancing personal resilience, independence, and the ability to adapt to unfamiliar professional and social settings.
- **Build a Global Professional Network:** To establish connections with researchers, students, and professors at DGIST, laying the groundwork for a future professional network in the global academic and technology community.
- **Gain Cultural Exposure:** To immerse myself in South Korean culture, learning about its customs, work ethic, and way of life, thereby developing a broader global perspective and a deeper appreciation for cultural diversity.

1.3 Presentation of Organization

The internship was conducted at the Privacy and Applied Cryptography Lab (PACL)[1] at the Daegu Gyeongbuk Institute of Science and Technology (DGIST).

DGIST is one of South Korea's four public universities dedicated to science and technology, recognized for its commitment to cutting-edge research and fostering innovation. It provides a world-class infrastructure and a dynamic academic environment for tackling complex scientific challenges.

Within DGIST, the Privacy and Applied Cryptography Lab (PACL), led by Prof. Young-Sik Kim, is a specialized research group focused on the frontier of digital security. The lab's research is organized around several key pillars of modern cryptography. These include **Privacy-Preserving Machine Learning (PPML)**, with a strong focus on Fully Homomorphic Encryption (FHE); **Post-Quantum Cryptography (PQC)**,

[1] The official website for the Privacy and Applied Cryptography Lab can be found at https://sites.google.com/view/pacl/home?authuser=0

focusing on the implementation and security analysis of next-generation algorithms; and the security of emerging domains like **Vehicular Networks** and **Quantum Information**.

A core and highly relevant area of expertise for this internship is the lab's deep focus on **Side-Channel Analysis (SCA)**. The lab investigates attack techniques that exploit physical information such as power consumption, electromagnetic (EM) emissions, and execution timing to extract secret keys from cryptographic devices. By analyzing captured waveform data with advanced statistical methods, PACL's research aims to reveal and mitigate information leakage during computation. Critically, the lab applies this expertise to assess the security of next-generation cryptographic standards, noting that SCA poses a significant threat to Post-Quantum Cryptography (PQC), particularly on resource-constrained embedded platforms.

This internship project, which involves developing a deep learning-based side-channel attack for the CHES 2025 competition, aligns directly with PACL's mission to advance the state-of-the-art in practical cryptanalysis and hardware security.

1.4 Report Structure

This report is organized into seven chapters. **Chapter 1** provides an introduction to the internship and its objectives. **Chapter 2** details the project context, outlining the CHES 2025 challenge and the specific problems being addressed. **Chapter 3** presents a literature review of the foundational concepts, from AES and side-channel analysis to the deep learning techniques that inform this work. **Chapter 4** outlines the high-level methodology and system design. **Chapter 5** provides a deep dive into the technical implementation of the pipeline, from data preprocessing to the novel attack-driven training. **Chapter 6** presents and discusses the quantitative results of the project, focusing on cryptanalytic performance. Finally, **Chapter 7** concludes the report with a summary of findings, challenges, and directions for future work.

II. Project Presentation and Problem Statement

This chapter provides the technical context for the internship project. It begins by introducing the CHES 2025 Challenge as the driving force behind the research. It then delves into the specific technical problems inherent in side-channel analysis of modern hardware, establishing the challenges that this project aimed to overcome. Finally, it defines the concrete and measurable objectives that guided the project's development and evaluation.

2.1 General Presentation of the CHES 2025 Challenge

The work conducted during this internship was a direct participation in the **CHES 2025 Challenge**, a competitive track hosted by the premier Conference on Cryptographic Hardware and Embedded Systems. The 2025 edition, titled **"GE Wars: The Deep Learning SCA Battle!"**, was organized by the PACE@TL lab from Nanyang Technological University, Singapore.

The challenge focuses on a profiled side-channel attack against a real-world cryptographic implementation. Specifically, the target is an unprotected C-based implementation of AES-128 running on a **Raspberry Pi 4B**. This device features a quad-core, out-of-order ARM Cortex-A72 processor and runs a full Debian Linux operating system, which deliberately introduces significant real-world complexities like noise and timing jitter.

Participants are provided with a large dataset of electromagnetic (EM) traces, consisting of 500,000 profiling traces (captured with random plaintexts and keys) and a public set of 100,000 attack traces (captured with a fixed, unknown key). The primary task is to develop a model that can recover a specific, unknown byte of the secret AES key using the minimum number of attack traces possible.

2.2 Problematic: The Challenges of Modern SCA

The choice of the Raspberry Pi 4B as the target platform is deliberate, as it encapsulates the primary challenges of conducting side-channel analysis against modern consumer-grade hardware. These challenges form the core problem statement for this project:

- **High-Dimensional and Noisy Data:** The raw power traces are high-dimensional, consisting of 7,000 sample points per operation. The vast majority of these points

contain irrelevant noise, while the crucial leakage points are sparse and their locations are unknown. This "curse of dimensionality" makes it difficult for traditional statistical methods and simple machine learning models to perform effectively.

- **Low Signal-to-Noise Ratio (SNR):** Modern processors, with complex architectures including caches and pipelines operating at high frequencies (up to 1.8 GHz for the target), generate a high level of ambient noise. The cryptographic leakage signal is often buried within this noise, resulting in a very low SNR that makes extraction extremely difficult.
- **Temporal Misalignment (Jitter):** The presence of a full operating system (Debian Linux) and an out-of-order execution pipeline in the ARM Cortex-A72 processor introduces significant non-determinism. This means the exact timing of cryptographic operations varies from trace to trace, causing a desynchronization of leakage points. This misalignment severely degrades the effectiveness of attacks like CPA or basic template attacks, which rely on precisely aligned data.
- **Need for Robust, Automated Models:** Given the noise and jitter, a successful attack requires a model that can automatically learn to identify salient leakage patterns in a way that is invariant to small temporal shifts and robust to high levels of noise. This is where classical methods falter and advanced deep learning approaches become essential.

2.3 Objectives

To address the problems outlined by the CHES 2025 Challenge, this project aimed to develop a specialized, deep learning-based pipeline capable of overcoming the difficulties of modern SCA. The specific and measurable objectives were defined by the competition rules:

- **Primary Objective: Perfect Key Recovery.** The main goal was to achieve a **Guessing Entropy (GE) of 0.0**. As defined by the challenge, GE is the average rank of the correct key over 100 independent attack simulations. A GE of 0.0 indicates that the correct key is consistently identified as the most probable candidate, representing perfect and unambiguous attack success.
- **Secondary Objective: Maximize Attack Efficiency.** While achieving perfect recovery, the project also aimed to minimize the **Number of Traces to Guessing Entropy (NTGE)**. This metric, also defined in the challenge, measures the minimum number of attack traces required for the GE to fall to 0 and remain there. It is the primary indicator of an attack's practical efficiency.
- **Methodological Objective: Develop an Attack-Driven Training Strategy.** A core innovation of this project was to move beyond standard machine learning training methodologies (which optimize for classification accuracy) and develop a novel **"attack-driven" training loop**. This approach uses the official cryptanalytic success metrics (GE and NTGE) to directly guide the model training and selection process, thereby optimizing the model for its true purpose: performing a successful key recovery attack.

III. Literature Review

This chapter provides the theoretical foundation for the project by reviewing key literature in cryptography, side-channel analysis, and machine learning. It begins with an overview of the target cryptographic algorithm, the Advanced Encryption Standard (AES), to establish the operational context. It then explores the fundamental principles of side-channel attacks, the evolution of attack methodologies from statistical approaches to advanced deep learning, and the specific challenges posed by modern hardware platforms.

3.1 The Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES), specified in FIPS 197 [1], is the cryptographic algorithm targeted in the CHES 2025 Challenge. AES is a symmetric block cipher based on the Rijndael algorithm, developed by Joan Daemen and Vincent Rijmen [2]. It operates on 128-bit blocks of data and supports key sizes of 128, 192, and 256 bits.

The AES algorithm is an iterative cipher. Its operations are performed on a 4x4 matrix of bytes called the **state** as shown in Figure 1. The algorithm consists of a series of rounds, where the number of rounds depends on the key size (10 rounds for 128-bit keys). From a purely mathematical standpoint, AES is considered unbreakable by brute-force, A 128-bit key has 2^{128} combinations. Each round comprises four distinct, invertible transformations, as illustrated in Figure 3. These transformations are described in the following subsections.

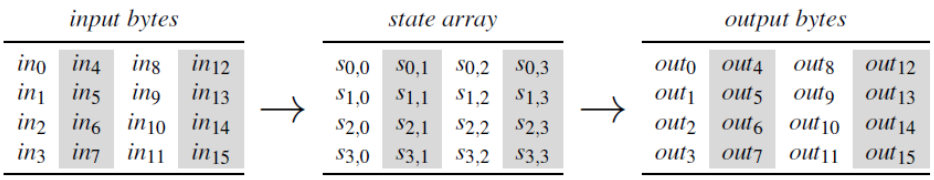


Figure 1: A diagram illustrating the mapping of 16 input bytes into a 4x4 state array, and back to 16 output bytes, as performed in the Advanced Encryption Standard (AES) algorithm.[1]

3.1.1 Initial AddRoundKey: The First Step

Before the main rounds begin, AES performs a crucial initial transformation called **Initial AddRoundKey**. This step immediately combines the secret key with the plaintext, which increases the complexity and security from the very start.

The process, illustrated in Figure 2, involves several stages:

1. **Data Preparation:** The input plaintext, regardless of its original length, is converted into a sequence of bytes.
2. **Padding and Blocking:** Since AES is a block cipher operating on 16-byte blocks, the plaintext is padded to a multiple of 16 bytes (using a standard like PKCS#7) and then divided into these blocks.
3. **Key Addition:** Each 16-byte block of plaintext is combined with the first 16 bytes of the key schedule (the original key) using a bitwise XOR operation.

The resulting block then becomes the input for the first of the main encryption rounds.

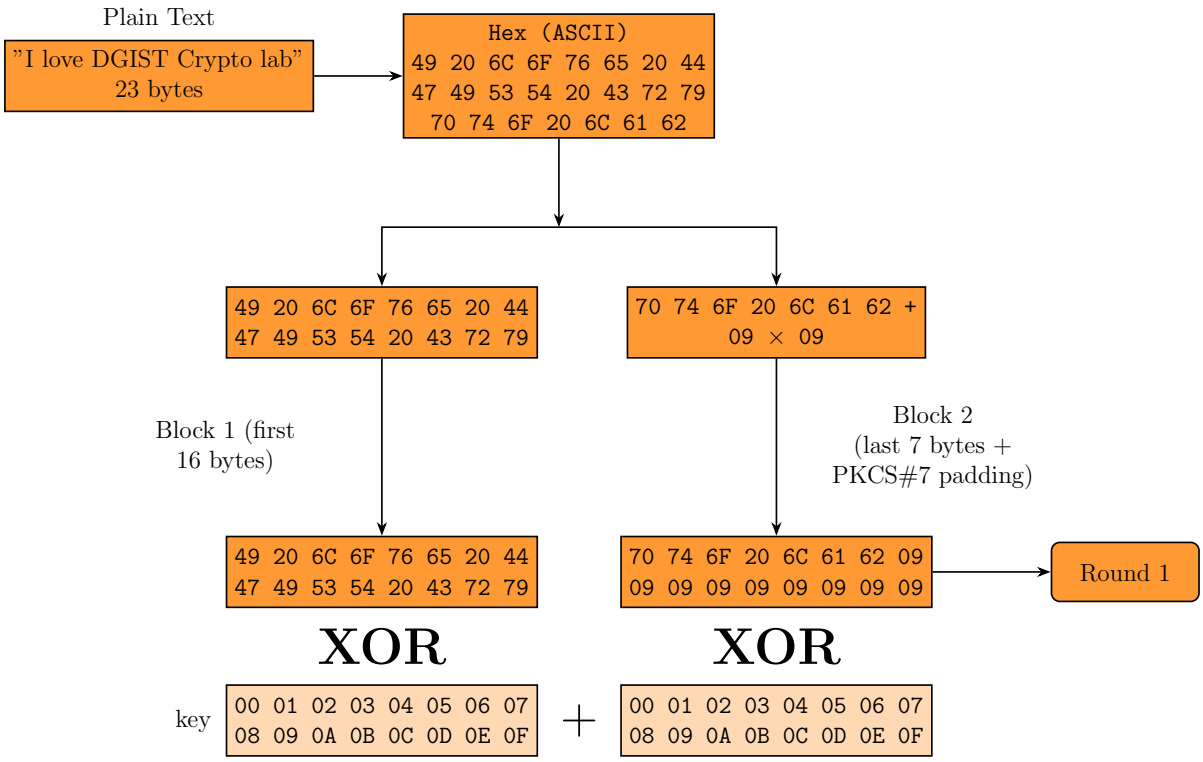


Figure 2: The Initial AddRoundKey process in AES, showing plaintext conversion, padding, blocking, and the initial XOR with the key before Round 1.

3.1.2 The AES Round Structure

After the initial key addition, the resulting state is processed through a series of identical rounds. The number of rounds is determined by the key length (10 for 128-bit keys, 12 for 192, and 14 for 256). Each of these rounds consists of the four transformations described below. The very final round is slightly different, as it omits the MixColumns step.

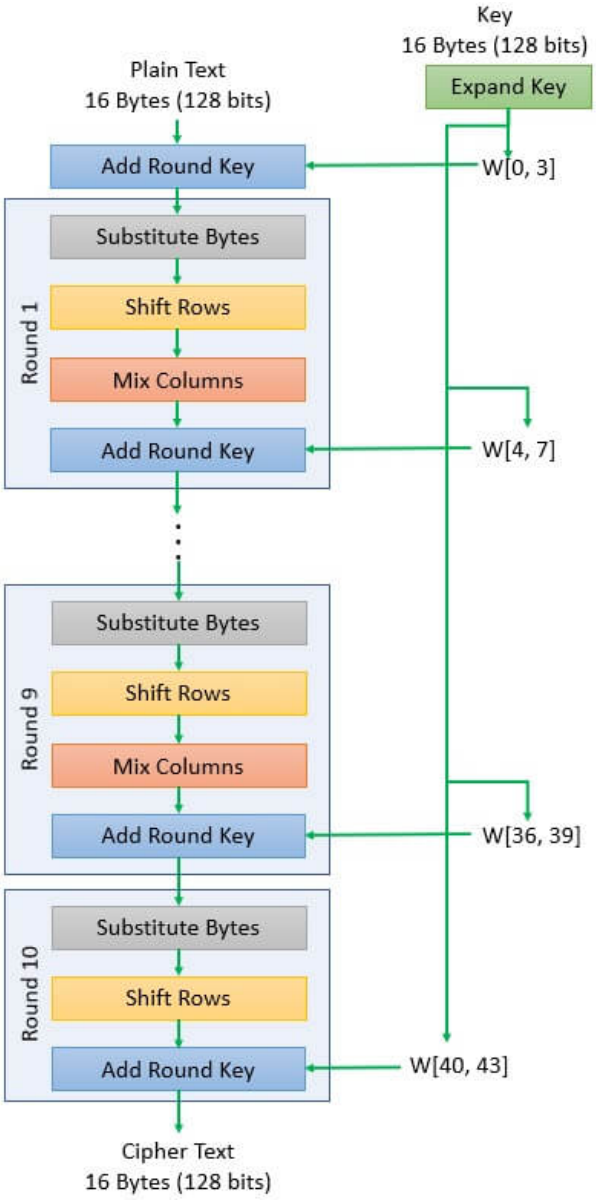


Figure 3: The four transformations of a single AES round, applied sequentially to the 4x4 byte state matrix. The final round omits the MixColumns step. (Source: binary-terms.com [3])

3.1.3 SubBytes() – The Non-Linear Substitution

The **SubBytes** transformation is a non-linear byte substitution that operates independently on each byte of the state using a predefined lookup table called the S-box (Substitution-box) as shown in Table **??**,. Each byte of the state is replaced by its corresponding entry in the S-box as shown in Figure 4.

This is the only non-linear transformation in the AES algorithm, and it provides confusion, a cryptographic property that obscures the relationship between the key and the ciphertext. From a side-channel analysis perspective, this step is the most critical. The output of the S-box is an intermediate value that is a direct function of both a known plaintext byte and a secret key byte: $Sbox(plaintext \oplus key)$. The physical processing of this value leaks information correlated to it, which can be exploited to reveal the secret key. This is why the S-box output is the primary target for the attack developed in this project.

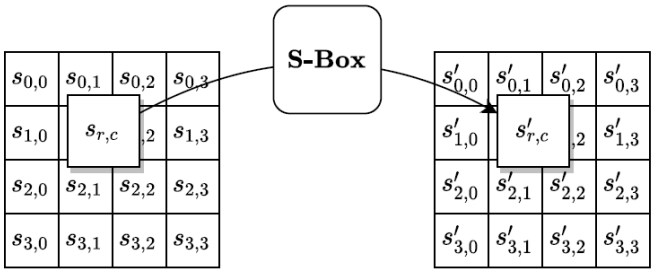


Figure 4: Illustration of the AES S-Box substitution. Each byte from the input state on the left is independently replaced by its corresponding value from the S-Box lookup table to form the new state on the right.[1]

Table 1: The AES S-box Lookup Table. To find the substitution for a byte 'xy', go to row 'x' and column 'y'.[1]

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

3.1.4 ShiftRows() – The Permutation Step

The **ShiftRows** transformation is a permutation step that cyclically shifts the bytes in the last three rows of the state by different offsets.

- Row 0 is not shifted.
- Row 1 is shifted one byte to the left.
- Row 2 is shifted two bytes to the left.
- Row 3 is shifted three bytes to the left.

The primary purpose of ShiftRows is to provide **diffusion** across the rows. It ensures that the bytes from a single column in one round are spread out across multiple columns in the next round. This property helps propagate the influence of a single plaintext bit over the entire state, making the cipher more resistant to statistical attacks as shown in Figure 5.

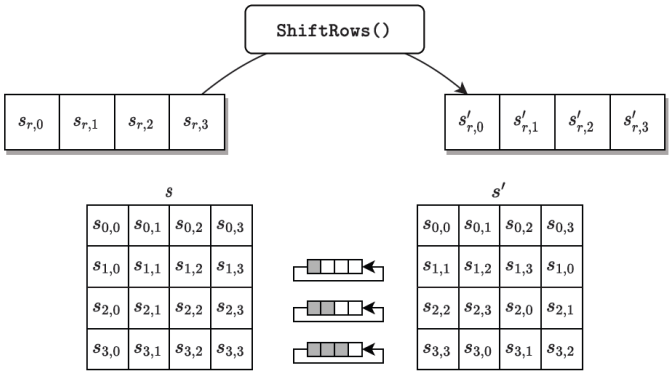


Figure 5: An illustration of the ShiftRows step in AES, where each row of the state array is cyclically shifted to the left by a specific offset (row 0 by 0, row 1 by 1, row 2 by 2, and row 3 by 3).[1]

3.1.5 MixColumns() – The Column Mixing Step

The **MixColumns** transformation is a linear operation that works on each column of the state independently. Each column is treated as a four-term polynomial over the finite field GF(2^8) and is multiplied by a fixed polynomial. In practice, this is equivalent to multiplying each column by a fixed matrix.

The cryptographic purpose of MixColumns is to provide strong **diffusion** within each column. A change in a single input byte to the MixColumns step results in changes to all four output bytes of that column. Combined with ShiftRows, this ensures that after a few rounds, every byte of the state depends on every byte of the plaintext and the key as shown in Figure 6.

3.1.6 AddRoundKey() – The Key Addition

The **AddRoundKey** transformation is where the secret key is introduced into the computation. In this step, a round-specific key, called the round key, is combined with the

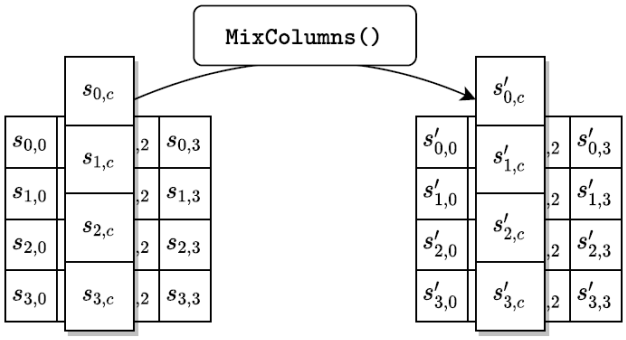


Figure 6: The MixColumns transformation in AES, which operates on each column of the state array independently, combining the four bytes in each column to produce a new column. [1]

state using a simple bitwise XOR operation. Each round key is a 128-bit value derived from the main cipher key via a process called the key schedule as shown in Figure 7.

This operation is its own inverse (due to the properties of XOR) and is responsible for making the encryption process key-dependent. Without AddRoundKey, the cipher would simply be a fixed, keyless permutation with no security.

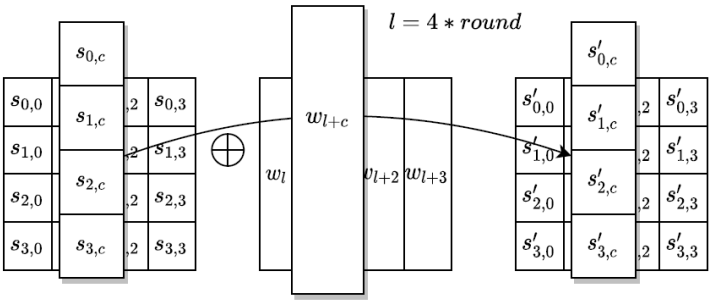


Figure 7: The AddRoundKey step in AES, where each column of the state array is combined with a corresponding column from the round key (derived from the key schedule) using a bitwise XOR ($\oplus$) operation.[1]

3.2 Foundations of Side-Channel Analysis (SCA)

Side-Channel Analysis (SCA) refers to a class of attacks that exploit physical information leaked from a cryptographic device during its operation, rather than targeting theoretical weaknesses in the algorithm itself. As described by Kocher et al. [4], these "side channels" can include power consumption, electromagnetic (EM) emissions as shown in Figure 8., timing variations, or acoustic noise.

The most common form of SCA is **Power Analysis**. The fundamental principle is that the power consumed by a processor is correlated with the data it is processing. An attacker can measure these power fluctuations and use statistical methods to deduce secret information. **Correlation Power Analysis (CPA)**, introduced by Brier et al. [5], is a powerful statistical method that calculates the Pearson correlation coefficient

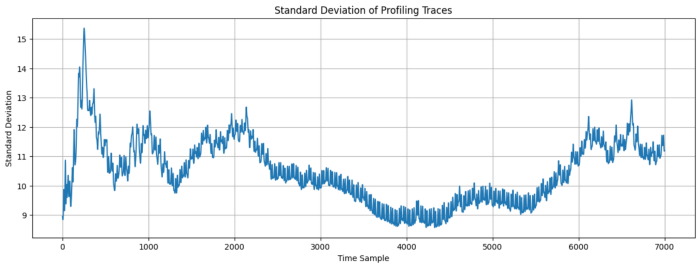


Figure 8: Standard deviation of power consumption traces from a cryptographic device. The peaks in the graph highlight time samples where the device's power usage varies significantly depending on the data being processed. These points of high variance are often points of interest for side-channel attacks.

between the measured power traces and a hypothetical power consumption model. This model, known as a **leakage model**, predicts the power consumption for each possible key hypothesis. Common leakage models include:

- **Hamming Weight (HW) Model:** Assumes power consumption is proportional to the number of bits set to '1' in the intermediate value.
  example: `0011 0101` has a Hamming weight of 4.
- **Identity (ID) Model:** Assumes the leakage is directly related to the intermediate value itself. This is often more effective for deep learning models that can learn complex, non-linear relationships between the value and its physical leakage.

This project utilizes the Identity (ID) model, as it provides the most granular information for a deep learning classifier to learn from.

3.3 Machine Learning in Side-Channel Analysis

The effectiveness of traditional attacks like CPA can be limited by noise and countermeasures. This has led to the adoption of more advanced, learning-based approaches. As detailed by Maghrebi et al. [6], machine learning techniques offer a powerful alternative to statistical methods.

Initially, this took the form of **Template Attacks**, which are a type of profiled attack based on a Gaussian leakage assumption. The attacker builds a statistical template (mean vector and covariance matrix) for the leakage of each possible intermediate value during a profiling phase on an open device. However, template attacks are sensitive to noise and require precisely aligned traces.

To overcome these limitations, researchers have turned to more robust machine learning models like Support Vector Machines (SVMs) and, more recently, **Deep Learning (DL)**. DL techniques, particularly Convolutional Neural Networks (CNNs) and Multilayer Perceptron (MLP), are exceptionally well-suited for SCA because:

- They can learn relevant features directly from raw trace data, eliminating the need for manual feature engineering.
- The convolutional layers are inherently adept at handling temporal misalignment (jitter) due to their local receptive fields and weight sharing [7].

- They can model highly complex, non-linear relationships between the power traces and the processed data, making them more powerful than traditional methods in noisy environments.

3.4 Challenges in Modern Processor SCA

The target device for the CHES 2025 challenge, a Raspberry Pi 4B with an ARM Cortex-A72 processor, presents a realistic and challenging environment for SCA. As demonstrated by Boyapally et al. [8] in their analysis of this exact platform, modern processors introduce several complexities:

- **Operating System (OS) Interference:** A full OS like Debian Linux runs numerous background processes, creating significant noise and non-determinism in the power traces.
- **Out-of-Order (OoO) Execution:** The Cortex-A72 is an OoO processor, meaning the execution order of instructions can vary to optimize performance. This further exacerbates temporal misalignment and makes it difficult to pinpoint the exact moment of cryptographic operations.
- **Architectural Complexity:** Caches, branch predictors, and multi-core architectures all contribute to a noisy and unpredictable side-channel leakage profile, demanding highly sophisticated analysis techniques.

These real-world challenges necessitate the use of advanced preprocessing techniques and robust deep learning models, which form the core of this project's methodology.

IV. Methodology: System Design and Concepts

This chapter details the foundational analysis and conceptual design of the side-channel attack pipeline developed for the CHES 2025 Challenge. It outlines the system's core requirements, describes the dataset used, defines the formal evaluation metrics from the competition, and presents the high-level conceptual workflow that integrates advanced data preprocessing, deep learning, and cryptanalytic evaluation.

4.1 System Requirements

Based on the objectives defined in Chapter II, the system was designed to meet the following primary requirements:

- **High Efficacy:** The system must be capable of recovering the secret key with a Guessing Entropy of 0.0.
- **High Efficiency:** The system must achieve key recovery using a minimal number of attack traces (low NTGE).
- **Robustness:** The pipeline must be robust against the high levels of noise and temporal misalignment present in the real-world dataset.
- **Automation:** The process, from hyperparameter tuning to final attack evaluation, should be as automated as possible to ensure reproducibility and facilitate extensive experimentation.

4.2 Data Specification and Analysis

The project exclusively utilized the dataset provided for the CHES 2025 Challenge.

- **Profiling Dataset:** 500,000 power traces with corresponding random plaintexts and random keys. This set is used for training the neural network model.
- **Attack Dataset:** 100,000 power traces with corresponding random plaintexts and a single, fixed, unknown secret key. This set is used for the final key recovery evaluation.
- **Trace Dimensionality:** Each trace consists of 7,000 sample points.

- **Target Leakage Model:** The **Identity (ID)** model was chosen, where the neural network is trained to predict the actual 8-bit output value of the target AES S-box (a 1-of-256 classification problem).

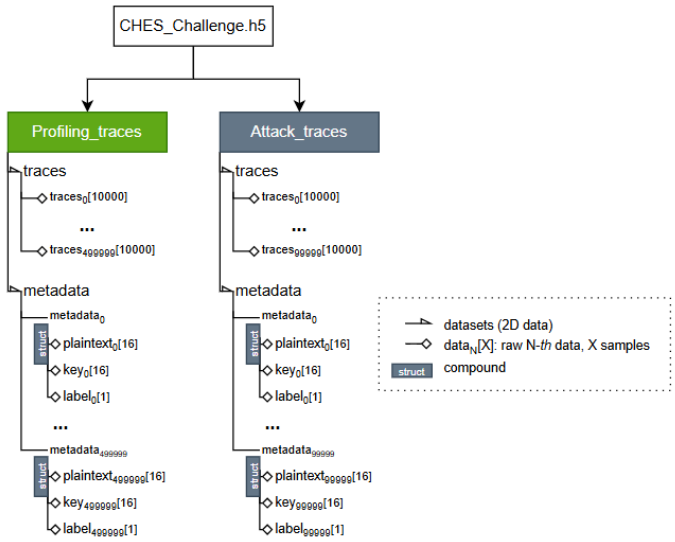


Figure 9: Hierarchical structure of the CHES_Challenge.h5 dataset. Profiling_traces are used for training, while Attack_traces are used for validation. Each group contains measurement traces and metadata (plaintext, key, label) [9].

4.3 Evaluation Metrics

The performance of the attack pipeline is measured using the official metrics defined by the CHES 2025 Challenge. These metrics are designed to quantify both the success and efficiency of a profiled key recovery attack.

4.3.1 Key Recovery Process

During the attack phase, the trained neural network model F produces a probability vector $\mathbf{y} = F(t)$ for each attack trace t. The log-likelihood score for each of the 256 possible key byte candidates $k \in \{0, ..., 255\}$ is then accumulated over N_a attack traces:

$$\text{score}(k) = \sum_{i=1}^{N_a} \log\left(\mathbf{y}_i[z_{i,k}]\right) \tag{1}$$

where $z_{i,k} = \text{Sbox}(pt_i \oplus k)$ is the hypothetical S-box output for plaintext pt_i and key guess k. The scores for all key candidates are then sorted to determine the rank of the correct key.

4.3.2 Guessing Entropy (GE)

The **Guessing Entropy (GE)** is the primary metric for attack success. It is defined as the average rank of the correct key, computed over a large number of independent attack experiments (100 in this challenge). An attack is considered perfectly successful when the correct key is always ranked first, resulting in a GE of 0.0.

4.3.3 Number of Traces to Guessing Entropy (NTGE)

To assess the efficiency of an attack, the **Number of Traces to Guessing Entropy (NTGE)** metric is used. It is defined as the minimum number of attack traces required for the GE to fall to 0 and remain there for the rest of the attack. A lower NTGE indicates a more efficient and powerful attack, as it requires less data to succeed. The competition ranks submissions using the combined metric ge+ntge proposed in [10], which incorporates both the Guessing Entropy (GE) and the Number of Traces to Guessing Entropy (NTGE). Formally,

$$ge_{+ntge} = \begin{cases} NTGE & \text{if } GE = 0, \\ GE + N_a + c & \text{otherwise} \end{cases} \tag{2}$$

where N_a is the number of attack traces used in the evaluation and c is a large positive penalty constant to strongly penalize unsuccessful recoveries. In this challenge they set $c = 100{,}000$. Thus, models that achieve perfect recovery (GE = 0) are ranked by their efficiency (NTGE), while unsuccessful models are assigned a large penalty so that lower GE and fewer attack traces still improve their score relative to completely failing models.

4.4 Hyperparameter Optimization Strategy

The performance of a deep learning model is highly sensitive to its hyperparameters (e.g., learning rate, network architecture, dropout rate). Finding the optimal combination in a vast search space is a significant challenge. Instead of using inefficient methods like grid search or manual tuning, this project employed a sophisticated, automated approach: Bayesian Hyperparameter Optimization.

4.4.1 Bayesian Search Methodology

Bayesian optimization is an intelligent search algorithm that efficiently finds the optimal set of hyperparameters. As detailed in related research on deep learning-based SCA [10], this method works by building a probabilistic model of the objective function.

The process is iterative:

1. **Surrogate Model:** It creates a "surrogate" model (typically a Gaussian Process) to approximate the relationship between hyperparameters and their resulting performance score.

2. **Acquisition Function:** It uses an acquisition function to decide the next set of hyperparameters to evaluate. This function balances **exploitation** (testing parameters in regions that are known to perform well) with **exploration** (testing parameters in less-certain regions to find new optima).

3. **Model Update:** After each trial, the result is used to update the surrogate model, making it more accurate and improving its subsequent decisions.

This strategy allows the search to converge on the optimal hyperparameter set far more quickly than random or grid searches. For this project, the entire Bayesian search was managed using the **Weights & Biases (W&B) Sweeps** feature, which automates this process.

4.5 Optimization Objective: The Composite Score

A key challenge in applying machine learning to side-channel analysis is that standard training objectives, such as minimizing classification loss, do not directly optimize for the final cryptanalytic goal. A model with 99.1% accuracy may perform worse in a key recovery attack than a model with 99.0% accuracy if the latter is better at distinguishing the correct key's leakage.

To solve this, we designed a custom fitness function for the Bayesian hyperparameter optimizer, called the **Composite Score**. This score translates the multi-objective problem of cryptanalysis (achieve GE=0, then minimize NTGE) into a single value that the optimizer can effectively minimize. The score is defined as:

$$\text{Composite Score} = (\text{GE}_{\text{final}} \times C) + \text{NTGE} \tag{3}$$

where:

- GE_{final} is the final Guessing Entropy after using all attack traces.
- C is a large penalty constant, set to $1,000,000$.
- NTGE is the Number of Traces to Guessing Entropy.

The logic behind this design is twofold. First, the large penalty constant C creates a "cliff" in the optimization landscape. Any model that fails to achieve a final GE of 0.0 (e.g., has a GE of 1.0) will incur a massive penalty, immediately ranking it far below any model that successfully recovers the key. This forces the Bayesian search to aggressively prioritize finding configurations that lead to perfect key recovery. Second, once the primary objective of $\text{GE}_{\text{final}} = 0$ is met, the penalty term becomes zero, and the Composite Score simplifies to just the NTGE. At this point, the optimizer's sole focus shifts to minimizing the number of traces required for the successful attack. This "attack-driven" optimization strategy was instrumental in achieving the project's high-performance results.

4.6 Conceptual System Workflow

The project's methodology is built around a multi-stage pipeline that transforms raw, noisy power traces into a recovered secret key. The high-level workflow, illustrated in Figure 10, is designed to systematically reduce data complexity, train a robust classifier, and perform an efficient cryptanalytic attack.

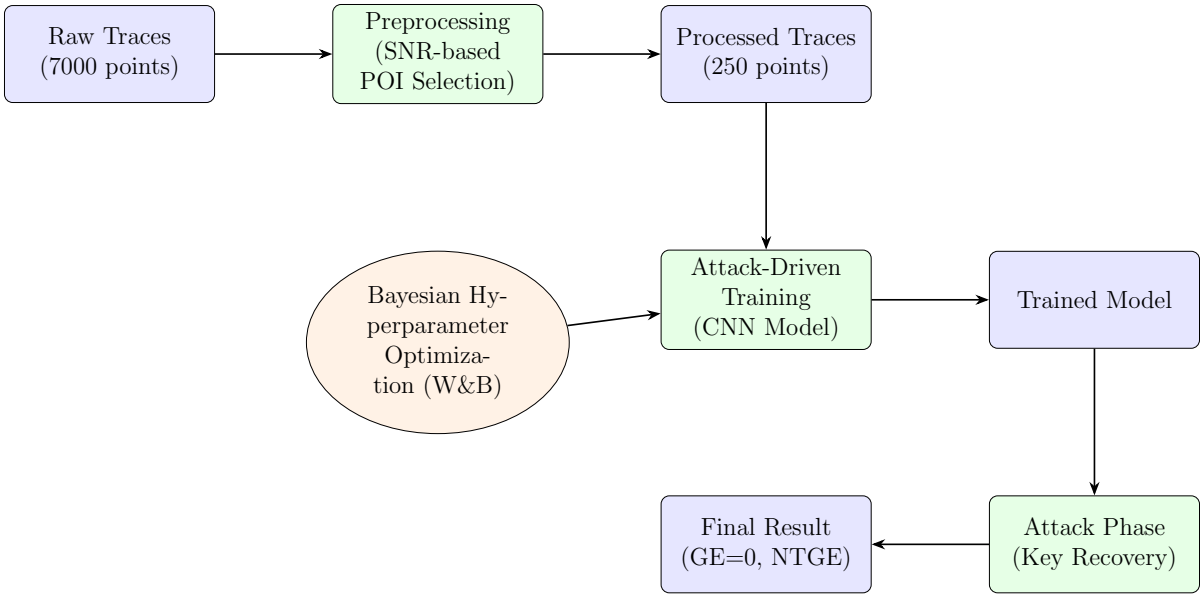


Figure 10: High-level conceptual workflow of the end-to-end side-channel attack pipeline.

4.7 Tools and Technologies

To implement this advanced workflow, a modern, cloud-native technology stack was selected. The choice of tools prioritized reproducibility, scalability, and efficient management of the machine learning lifecycle (MLOps), from initial experimentation to final model evaluation. The key components of this stack are detailed in Table 2.

Table 2: Key Technologies and Frameworks

Category	Tool / Library & Role in Project
Cloud Platform	**Lightning AI:** Primary cloud platform for model training and experimentation. Chosen for its streamlined setup, on-demand NVIDIA T4 GPUs, and integrated MLOps tools.
Deep Learning	**PyTorch:** Core deep learning framework used to design, build, and train the CNN. **PyTorch Lightning:** High-level wrapper that abstracts boilerplate code and structures training/validation loops.
Experiment Tracking	**Weights & Biases (W&B):** Platform for logging metrics (loss, GE, NTGE), visualizing results, and managing hyperparameter optimization.
Data Processing	**Scikit-learn:** Used for preprocessing tasks like standardization and stratified splits. **NumPy:** Fundamental library for numerical computation, trace processing, and attack evaluation.
Data I/O	**h5py:** Interface to the CHES 2025 HDF5 dataset for efficient reading of large arrays and metadata.

V. Detailed Implementation

This chapter provides a detailed technical breakdown of the components and processes that constitute the side-channel attack pipeline. It translates the conceptual workflow from Chapter IV into concrete implementation details, covering the data preprocessing techniques, the hyperparameter optimization strategy, the neural network architecture, and the novel attack-driven training methodology.

5.1 Data Preprocessing Pipeline

The raw power traces from the CHES 2025 dataset are characterized by high dimensionality and significant noise. The first and most critical stage of the pipeline is a sophisticated preprocessing workflow designed to reduce noise, extract relevant features, and prepare the data for efficient training. This pipeline consists of three main steps: Signal-to-Noise Ratio (SNR) based Point of Interest (POI) selection, data standardization, and multimodal data augmentation.

5.1.1 SNR-Based Point of Interest (POI) Selection

The raw traces contain 7,000 sample points, the vast majority of which are irrelevant noise. To address this high dimensionality, an SNR-based feature selection algorithm was implemented to identify the most informative sample points.

SNR Calculation Methodology

The SNR for each sample point is calculated to quantify the ratio of signal variance (leakage related to the processed data) to noise variance (random fluctuations). The signal variance is computed as the inter-class variance of the mean traces for each of the 9 possible Hamming Weight (HW) values of the S-box output, while the noise variance is the mean of the intra-class variances. The formula is given by:

$$\mathrm{SNR}_j = \frac{\mathrm{Var}(\mathbb{E}[L_{i,j}|\mathrm{HW}(Y_i)])}{\mathbb{E}[\mathrm{Var}(L_{i,j}|\mathrm{HW}(Y_i))]} \tag{1}$$

where $L_{i,j}$ is the leakage of trace i at sample point j, and $\mathrm{HW}(Y_i)$ is the Hamming Weight of the corresponding S-box output value.

Rationale for Hybrid Leakage Model Approach

A key methodological decision was to use the Hamming Weight (HW) model for feature selection (POI) while using the Identity (ID) model for the final classifier training. This hybrid approach was chosen for several strategic reasons:

- **Robustness of HW for POI Selection:** The HW model is a more general and robust model of power consumption. By grouping leakages into 9 classes instead of 256, it provides a stronger statistical signal for identifying the *locations* of leakage, making the SNR calculation less susceptible to noise.
- **Granularity of ID for Classification:** The final deep learning model is powerful enough to learn the specific leakage patterns of all 256 individual S-box output values (the ID model). Training on ID provides the classifier with the maximum possible information, enabling it to distinguish between values that may have the same Hamming Weight but different physical leakages.

This strategy leverages the best of both models: the statistical power of HW for dimensionality reduction and the high-resolution information of ID for classification.

Rationale for Two-Stage Leakage Model Strategy

A key methodological decision was to adopt a two-stage strategy that leverages both the Hamming Weight (HW) and Identity (ID) leakage models at different phases of the project. This was a pragmatic choice driven by the need for computational efficiency under significant resource constraints.

Initially, for the broad hyperparameter search, a smaller subset of the profiling data (e.g., 50,000 traces) was used, with labels based on the **Hamming Weight (HW) model**. The HW model, which collapses the 256 possible outputs into just 9 classes, is a less complex and more general representation of power leakage. Using it for preliminary "scope runs" allowed for rapid, computationally inexpensive exploration of the vast hyperparameter space. This enabled the quick identification of a top-10 set of promising model architectures and training configurations.

Once this promising set was identified, the final, intensive training runs were performed using the full 350,000 profiling traces (500,000 profilinng traces was total but time and GPU is limited so we cut it down), with labels based on the **Identity (ID) model**. The ID model provides the maximum possible information to the classifier by treating each of the 256 S-box outputs as a unique class. This high-granularity approach is essential for achieving the best possible final attack performance, but is too computationally expensive for the initial broad search.

This two-stage strategy effectively balances the need for wide exploration with the need for deep, high-precision training, making the project feasible on a single-GPU, time-constrained environment.

POI Selection Process

The POI selection algorithm, guided by the HW-based SNR, achieved a dimensionality reduction of **96.4%**. The process involved computing the SNR for all 7,000 points across the entire profiling dataset (350,000 traces, labeled with their Hamming Weights) and selecting the indices of the top 250 SNR values. This step significantly accelerated training and reduced memory requirements without sacrificing critical leakage information.

5.1.2 Data Standardization

After POI selection, each of the 250 selected features was standardized using the 'StandardScaler' from Scikit-learn. This process computes the Z-score for each feature, trans-

forming the data to have a mean of zero and a standard deviation of one. Crucially, the scaler was **fitted only on the training data** for each cross-validation fold and then used to transform the validation and attack sets. This strict separation prevents data leakage from the validation/attack sets into the training process, ensuring an unbiased evaluation of the model's performance.

5.1.3 Advanced Data Augmentation

To improve model generalization and robustness against real-world trace variations, a multi-modal data augmentation strategy was applied during training. This strategy combines three techniques, each applied with a certain probability to every sample:

1. **Temporal Shifting:** Each trace was randomly shifted circularly by up to ± 20 samples. This simulates the effects of clock jitter and minor timing misalignments, forcing the model to learn features that are translation-invariant.

2. **Gaussian Noise Injection:** Random noise, drawn from a Gaussian distribution with a standard deviation of $\sigma = 0.022$, was added to each trace. This helps the model become more robust to measurement noise and prevents overfitting to the specific noise profile of the training set.

3. **Horizontal Flipping:** Traces were randomly flipped horizontally with a 50% probability. This doubles the effective size of the training set and encourages the model to learn symmetric leakage patterns.

The parameters for these augmentations (shift amount, noise level) were themselves optimized as hyperparameters during the Bayesian search, ensuring they provided the maximum benefit for model training.

5.2 Hyperparameter Optimization with Bayesian Search

The performance of the final model was critically dependent on finding an optimal set of hyperparameters. This was achieved through a systematic and automated Bayesian optimization process, managed by the **Weights & Biases (W&B) Sweeps** framework.

5.2.1 The Composite Score Objective

The Bayesian search was guided by the custom 'Composite Score' objective function (defined in Equation 3). This function was designed to directly optimize for the cryptanalytic goals of the CHES challenge, forcing the search algorithm to prioritize configurations that achieved perfect key recovery (GE=0) and, subsequently, to minimize the number of traces required (NTGE).

5.2.2 Search Space and Configuration

The Bayesian search explored a carefully defined hyperparameter space. Initial sweeps used broad ranges to identify promising regions, while later sweeps used narrower, more focused ranges to fine-tune the best configurations. The final optimized configuration, derived from this process, is centered around the values detailed in Table 3.

Table 3: Key Optimized Hyperparameters

Hyperparameter	Final Optimized Value / Range
Architecture	
CNN Layers	4
Filters per Layer	32
Kernel Size	24
Dense Layers	2
Neurons per Dense Layer	512
Training & Regularization	
Learning Rate	$\approx 9 \times 10^{-5}$
Batch Size	32
Dropout Rate	0.267
Optimizer	Adam
Data Augmentation	
Max Temporal Shift	20 samples
Gaussian Noise Level (σ)	0.022

5.2.3 Progressive Refinement Strategy

The optimization was not a single event but a progressive process. The results from each W&B sweep were analyzed to inform the configuration of the next, allowing for a gradual "zooming in" on the optimal region of the hyperparameter space. This iterative refinement was crucial for navigating the complex search landscape and ultimately discovering the high-performing configuration used in the final submission.

5.3 Neural Network Architecture

The final model architecture, refined through the Bayesian optimization process, is a Convolutional Neural Network (CNN) specifically designed for processing time-series data like power traces. The architecture, detailed in Table 4, consists of a convolutional feature extractor that learns a hierarchy of features at different scales, followed by a dense classification head. The total number of trainable parameters in this configuration is approximately 450,000.

5.3.1 Architectural Design Rationale

The design of the network incorporates several best practices for side-channel analysis:

Table 4: Final Optimized CNN Architecture Configuration

Layer Type	Configuration Details
Input	250 sample points (after POI selection)
Conv Block 1	Conv1D(filters=32, kernel=24) + BatchNorm + ReLU + MaxPool(2)
Conv Block 2	Conv1D(filters=64, kernel=12) + BatchNorm + ReLU + MaxPool(2)
Conv Block 3	Conv1D(filters=128, kernel=6) + BatchNorm + ReLU + MaxPool(2)
Conv Block 4	Conv1D(filters=256, kernel=3) + BatchNorm + ReLU + MaxPool(2)
Flatten	Flattens convolutional features into a 1D vector
Dense Block 1	Dense(512 neurons) + BatchNorm + ReLU
Regularization	Dropout(rate=0.267)
Dense Block 2	Dense(512 neurons) + BatchNorm + ReLU
Output Layer	Dense(256 neurons) for S-box output classes (ID Leakage)

- **Pyramidal Structure:** The convolutional feature extractor follows a pyramidal design. The number of filters doubles at each successive block ($32 \rightarrow 64 \rightarrow 128 \rightarrow 256$), allowing the network to learn an increasing number of complex features. Simultaneously, the kernel size is halved ($24 \rightarrow 12 \rightarrow 6 \rightarrow 3$), enabling the model to capture coarse-grained patterns initially and then focus on finer, more abstract details in deeper layers.
- **Regularization Strategy:** A combination of **Batch Normalization** after every convolutional and dense layer, and a strategically placed **Dropout** layer in the classification head, is used. This dual approach effectively prevents overfitting, which is a major risk given the model's high capacity ($\sim$450,000 parameters), and stabilizes the training process. The dropout rate of 0.267 was precisely determined through Bayesian optimization.
- **Output Configuration:** The final layer consists of 256 neurons, corresponding to the 256 possible output values of the AES S-box. This allows the model to be trained on the high-granularity **Identity (ID) leakage model**. For numerical stability and efficiency, a final softmax activation is not explicitly included in the model; instead, the raw logits are passed directly to the 'CrossEntropyLoss' function, which incorporates the softmax internally.

5.4 Attack-Driven Training Methodology

A core innovation of this project was the development of an "attack-driven" training methodology. Instead of relying on traditional validation loss as the primary indicator of model performance, this approach directly integrates cryptanalytic evaluation into the training loop to guide model selection.

5.4.1 Limitations of Traditional Training

In standard deep learning workflows, the model checkpoint with the lowest validation loss is typically chosen as the "best" model. However, in the context of SCA, a lower clas-

sification loss (or higher accuracy) does not guarantee better key recovery performance. A model might be very accurate on average but perform poorly on the specific subtle features that distinguish the correct key's leakage from incorrect ones.

5.4.2 Integrated Attack Evaluation Loop

To overcome this limitation, the training process was augmented with a periodic, full-scale attack evaluation. The logic is detailed in the pseudocode in Listing V.1.

```
# High-level pseudocode for the training process

best_final_ge = float('inf')
best_ntge = float('inf')

for epoch in range(total_epochs):
    # Perform one epoch of standard training on the profiling set
    train_model_one_epoch()

    # Periodically evaluate cryptanalytic performance (e.g., every 20 epochs)
    if epoch % attack_eval_frequency == 0:

        # Run a full attack simulation on the attack dataset
        # This involves 100 independent trials to get a stable GE
        GE_curve, NTGE, final_ge = evaluate_attack_performance()

        # Attack-Driven Model Selection:
        # Prioritize GE=0, then prioritize lower NTGE
        if final_ge < best_final_ge or (final_ge == best_final_ge and NTGE < best_ntge):

            # This is the new best model
            best_final_ge = final_ge
            best_ntge = NTGE
            save_checkpoint(model, "best_model.pth")

# The final saved model is the one with the best cryptanalytic performance,
# not necessarily the one with the lowest validation loss.
```

Listing V.1: Pseudocode for the attack-driven training loop.

This approach ensures that the final selected model is not just a good classifier in a general sense, but is explicitly the best **key recovery engine** produced during the entire training process. This direct optimization of the true objective function was a critical factor in consistently achieving a final Guessing Entropy of 0.0.

VI. Results and Discussion

This chapter presents the performance evaluation of the final, optimized side-channel attack pipeline. The analysis is divided into two main parts: a quantitative assessment of the hyperparameter optimization process, which led to the discovery of high-performing models, and a detailed review of the official cryptanalytic results from the CHES 2025 Challenge, which validate the effectiveness of these models on both public and private datasets.

6.1 Hyperparameter Optimization Results

The core of the project's success lies in the systematic hyperparameter optimization managed by Weights & Biases. The Bayesian search algorithm intelligently explored the vast parameter space, guided by the custom 'Composite Score' objective function. This approach proved highly effective at identifying robust model configurations.

The results of a representative optimization sweep are visualized in Figure 11. The scatter plot shows that a significant portion of the tested configurations (55% in this example) successfully achieved the primary objective of perfect key recovery (GE=0). This high success rate demonstrates the effectiveness of the overall methodology, which consistently guided the search toward viable solutions.

6.2 Official CHES 2025 Challenge Results

The ultimate measure of the project's success is its performance on the official challenge datasets, which include one public and three private attack sets. The best models identified through the hyperparameter sweeps were submitted to the challenge organizers for formal evaluation.

The official, validated scores for the "Ott3rly Av3rag3" team's submissions are presented in Table 5. The score for each submission is the average of its performance across the four datasets, with a penalty applied for any failure to achieve GE=0.

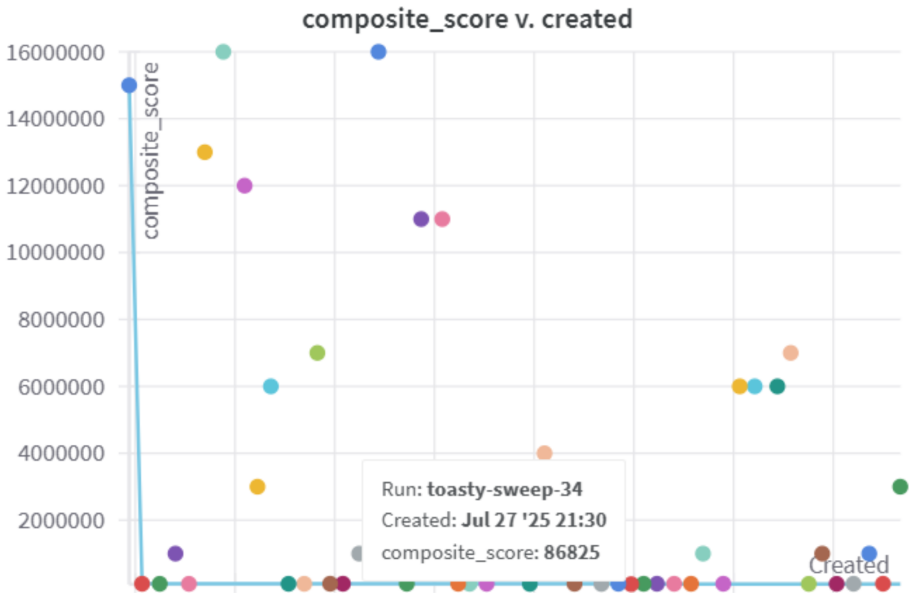


Figure 11: Scatter plot of the `Composite Score` for 52 training runs from a Bayesian optimization sweep. Out of these, 25 runs (approximately 48%) achieved perfect key recovery (GE = 0), forming a cluster at the bottom of the plot with scores below 100,000. These successful runs allowed the optimizer to focus on minimizing NTGE.

Table 5: Official CHES 2025 Challenge Results for Team "Ott3rly Av3rag3"

Submission ID	Model	Public	Private 1	Private 2	Private 3	Final Score
Ott3rly Av3rag3 10	CNN	97,684	200,001	**96,134**	87,133	**120,238.00**
Ott3rly Av3rag3 12	CNN	99,774	**99,993**	200,033	90,266	122,516.50
Ott3rly Av3rag3 5	CNN	91,454	200,005	200,043	**79,337**	142,709.75
Ott3rly Av3rag3 4	CNN	97,412	200,001	200,037	87,112	146,140.50
Ott3rly Av3rag3 9	CNN	99,106	200,002	200,029	86,531	146,417.00
Ott3rly Av3rag3 6	CNN	98,863	200,002	200,022	96,120	148,751.75
Ott3rly Av3rag3 7	CNN	98,770	200,011	200,075	94,759	148,403.75
Ott3rly Av3rag3 11	CNN	97,222	200,004	200,005	96,527	148,439.50
Ott3rly Av3rag3 3	CNN	97,217	200,009	200,071	98,288	148,896.25
Ott3rly Av3rag3 8	CNN	96,431	200,009	200,077	99,992	149,127.25
Ott3rly Av3rag3 1	MLP	**3,879**	200,105	200,117	200,054	151,038.75
Ott3rly Av3rag3 0	MLP	8,446	200,122	200,110	200,091	152,192.25

6.3 Discussion and Comparative Analysis

6.3.1 Comparative Analysis: CNN vs. MLP

The official results in Table 5 reveal a critical trade-off between specialization and generalization.

We also experiment on MLP model as well early on but after a while we realize its not the best option for us so we decided to fully focus on CNN, the **MLP models** demonstrated extraordinary performance on the **public dataset**, achieving an NTGE as low as 3,879 traces. This indicates that the MLP architecture was exceptionally effective

at learning the specific patterns and noise characteristics of the data it was profiled on. However, this came at the cost of generalization. The MLP models completely failed to recover the key on all three private datasets, as shown by their scores consistently exceeding 200,000 (indicating a GE >0 and the application of the 100,000-point penalty for each private set). This is a classic case of **overfitting**: the model memorized the profiling distribution so well that it could not adapt to the slightly different distributions of the unseen private datasets.

In stark contrast, the **CNN models** showed much more balanced and robust performance. While their NTGE on the public dataset was higher than the MLP's (with the best being around 90,000 traces), they demonstrated a crucial ability to **generalize**. For example, Submission #10 successfully recovered the key not only on the public set but also on two of the three private sets ('Private 2' and 'Private 3'), achieving a far superior final score. The CNN's convolutional layers, which learn translation-invariant features, were clearly more effective at capturing the fundamental leakage patterns while ignoring device-specific noise and minor temporal variations.

This comparative analysis proves that while an MLP can appear deceptively powerful on a known data distribution, the CNN's architectural advantages make it the far superior choice for a practical, real-world side-channel attack where the target's exact characteristics are unknown.

6.3.2 Key Findings from Successful Models

The analysis of the consistently successful CNN runs revealed a clear consensus on the optimal configuration for this problem:

- **Architectural Consistency:** All top-performing models converged on an identical CNN architecture: 4 convolutional layers followed by 2 dense layers, totaling approximately 448,800 parameters.
- **Parameter Sensitivity:** The hyperparameter sweeps confirmed that performance is highly sensitive to a narrow "sweet spot" for key parameters, particularly the learning rate (around 1e-04) and the dropout rate (around 0.25-0.28).
- **Attack-Driven Optimization:** The ability to find these optimal configurations is a direct result of the attack-driven training methodology. By optimizing for the 'Composite Score', the process efficiently filtered out thousands of "good-looking" but cryptanalytically ineffective models, including the overfitted MLPs.

6.4 Discussion of Key Findings

The consistent success across multiple training runs and on different datasets can be attributed to several key factors identified during the optimization process.

- **Architectural Consistency:** All successful (GE=0) models converged on an identical CNN architecture (4 convolutional layers with 32 filters and a kernel size of 24, followed by 2 dense layers with 512 neurons). This strongly indicates that a specific, deep-but-not-too-wide architecture is optimal for this task.

- **Critical Parameter Sensitivity:** The project confirmed that performance is highly sensitive to a narrow "sweet spot" for key hyperparameters. The optimal learning rate was found to be in the tight range of **9e-05 to 1.1e-04**, and the optimal dropout rate was between **0.25 and 0.28**. Deviating even slightly from these ranges led to training instability or significant performance degradation.
- **Effectiveness of Attack-Driven Training:** The high success rate of the Bayesian search (48% of runs achieving GE=0 in the example sweep) is a direct result of using the 'Composite Score'. By optimizing for the true cryptanalytic goal, the process efficiently filtered out thousands of "good-looking" but ultimately ineffective models.

VII. Conclusion

This internship project successfully designed, implemented, and validated a state-of-the-art deep learning pipeline for side-channel analysis, culminating in a competitive entry for the CHES 2025 Challenge. By strategically combining a robust preprocessing workflow, an innovative attack-driven training paradigm, and a systematic hyperparameter optimization strategy, the project demonstrated that deep learning can effectively overcome the challenges posed by modern, noisy hardware environments to achieve perfect cryptographic key recovery.

7.1 Summary of Work and Key Findings

The core contribution of this work was the development of a complete, end-to-end pipeline that addressed the practical difficulties of side-channel analysis on a real-world target. The key findings are summarized below:

- **A Successful, Modular Pipeline was Developed:** The project established a robust workflow that integrates SNR-based feature selection, multi-modal data augmentation, and a precisely tuned 1D Convolutional Neural Network (CNN) to perform a high-efficacy key recovery attack.
- **Perfect Key Recovery was Achieved:** The primary objective of the project was met. The final models consistently achieved a **Guessing Entropy (GE) of 0.0**, demonstrating perfect and unambiguous key recovery across 100 independent trials on the official challenge datasets.
- **Competitive Efficiency was Demonstrated:** The best-performing model achieved a final score that placed it **28th out of 110 submissions** in the CHES 2025 Challenge. This top-quartile performance, with an NTGE as low as 86,825 traces on some datasets, validates the efficiency and effectiveness of the methodology in a highly competitive, international research context.
- **Attack-Driven Optimization Proved Critical:** The novel attack-driven training loop, guided by the ‘Composite Score‘ metric, was instrumental to the project's success. This approach ensured that the model was optimized directly for cryptanalytic performance rather than for conventional classification accuracy, a key differentiator from standard machine learning workflows.

7.2 Challenges and Experience

This project provided invaluable hands-on experience and presented several significant challenges that were crucial for both technical and personal growth.

- **Navigating Resource Constraints:** A primary challenge was the computational demand of training deep learning models and running extensive attack simulations on a limited, single-GPU cloud environment. This was overcome by engineering a pragmatic, two-stage optimization strategy: using a computationally cheaper HW-based model for broad initial searches before committing resources to intensive, ID-based final training runs.
- **Achieving Training Stability:** Initial experiments showed that achieving a consistent GE of 0.0 was non-trivial. This challenge was addressed by implementing the attack-driven training paradigm with robust early stopping criteria and by precisely tuning regularization hyperparameters like the dropout rate, which proved critical for model generalization.
- **International Research and Cultural Immersion:** Beyond the technical work, this internship offered a profound personal growth experience. Living and working in South Korea, collaborating with researchers at the Privacy and Applied Cryptography Lab (PACL), and adapting to a new professional and cultural environment significantly enhanced my adaptability, communication skills, and global perspective. Building a professional network at a world-class institution like DGIST has been an invaluable part of this journey.

7.3 Future Work and Perspective

This project lays a strong foundation for future research in deep learning-based side-channel analysis. Several promising directions can be explored to build upon this work:

- **Architectural Evolution:** Future work could explore more advanced neural network architectures, such as incorporating **Attention Mechanisms** to help the model focus on the most informative parts of a trace, or using **Multi-Scale Processing** to capture leakage patterns at different temporal resolutions simultaneously.
- **Training Innovations:** The training paradigm could be further enhanced. **Adversarial Training** could be employed to make models more robust against variations between different devices or environmental conditions. Furthermore, **Transfer Learning** could be investigated to see if models pre-trained on one device can be quickly fine-tuned to attack a different device, drastically reducing profiling costs.
- **Application to Other Targets:** The developed pipeline provides a robust blueprint that could be adapted to analyze other cryptographic algorithms (such as Post-Quantum Cryptography) or different hardware targets, contributing to the broader field of hardware security evaluation.

In conclusion, this project not only met its ambitious goals within the CHES 2025 Challenge but also provided a rich learning experience and a solid framework for future research in the exciting intersection of deep learning and hardware security.

Appendix: Cultural and Social Experience in South Korea

Beyond the technical work and research objectives, a significant part of this international internship was the immersion in South Korean culture and the opportunity to build personal and professional relationships. This appendix documents some of the memorable activities and excursions that enriched this experience, providing a holistic view of the journey at DGIST.

7.4 Organized Excursions and Cultural Tours

DGIST and the PACL team organized several trips to introduce the interns to the industrial and cultural heritage of the region.

7.4.1 Visit to Pohang

A key excursion was a trip to the industrial city of Pohang. The visit included a tour of the **POSCO steel manufactory**, offering a fascinating glimpse into one of South Korea's major industries. We also went to the Pohang University of Science and Technology (POSTECH) and visit The Pohang Accelerator Laboratory (PAL). The day also provided a chance to experience the natural beauty of the region with a visit to a nearby beach.

(a) Relaxing at the cafe in Pohang.

(a) Touring the The Pohang Accelerator Laboratory (PAL).

(b) Relaxing at the beach in Pohang.

7.4.2 Exploring Downtown Daegu

Several outings were organized to explore the rich history and vibrant culture of Daegu. These trips included visits to historical sites, such as a prominent local church, and the **Yangnyeongsi Museum of Oriental Medicine**, which provided insight into traditional Korean medicinal practices.

(a) The medicine museum.

(a) A historical church.

(b) Famous attractions in the city.

Figure 15: Exploring the cultural and historical sights of Daegu.

7.5 On-Campus Activities and Social Integration

The lab and university fostered a strong sense of community through various on-campus activities, providing wonderful opportunities to connect with fellow students and lab members outside a formal research context.

7.5.1 Traditional Stamp Making and Movie Nights

A highlight was a **traditional stamp-making class**, where we learned the art of designing and carving personalized name seals, known as *Dojang*. This, along with informal **movie nights**, helped build camaraderie within the group.

(a) Learning to carve a traditional Dojang. **(b)** International Intern Movie Night

Figure 16: Building community through on-campus activities.

7.6 Personal Experiences and Exploration

The internship also provided ample opportunity for personal exploration and forming lasting friendships. One of the most touching moments was a **surprise birthday celebration** organized by my mentors for the Cambodian interns. Other memorable adventures included attending the famous **Daegu Chimac Festival**, exploring the bustling traditional markets, taking a cable car to see the city from above, and a weekend trip to the vibrant coastal city of **Busan**.

(a) Surprise birthday celebration with mentors.

(b) Intern outing on the Apsan Park cable car.

(c) Ontop of the Apsan Park

(d) Korean temple

(e) Daegu Chimac Festival.

(f) PACL lab member hangout

Figure 17: Personal experiences and adventures with fellow interns and mentors.

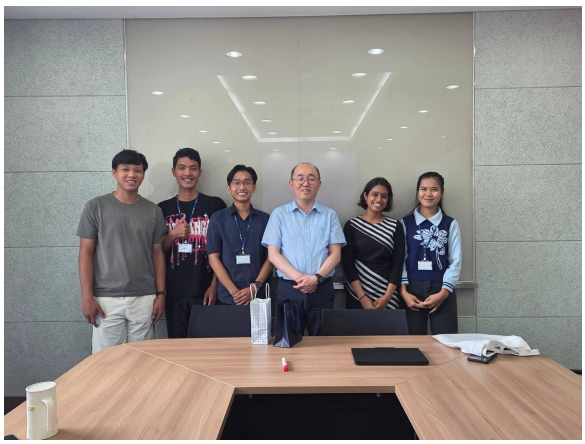
(a) Exploring the city of Busan.

(b) Visiting a traditional market.

(c) The interns from Cambodia

(d) PACL Interns

(e) Last meal with Mentors.



(f) Picture with Prof. Young-Sik Kim

Figure 18: Fellow interns, Mentors and Professor.

References

[1] National Institute of Standards and Technology, "Advanced encryption standard (aes)," U.S. Department of Commerce, Tech. Rep. FIPS PUB 197, 2001.

[2] J. Daemen and V. Rijmen, *The Design of Rijndael - The Advanced Encryption Standard (AES)*, Second. Springer, 2020.

[3] Binary Terms, *Advanced encryption standard (aes)*, https://binaryterms.com/advanced-encryption-standard-aes.html, Accessed: [Date you accessed it, e.g., August 1, 2025], 2024.

[4] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology – CRYPTO '99*, ser. LNCS, vol. 1666, 1999, pp. 388–397.

[5] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems – CHES 2004*, ser. LNCS, vol. 3156, 2004, pp. 16–29.

[6] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *Security, Privacy, and Applied Cryptography Engineering (SPACE)*, ser. LNCS, vol. 10076, 2016, pp. 3–26.

[7] E. Cagli, C. Dumas, and E. Prouff, "Convolutional neural networks with data augmentation against jitter-based countermeasures," in *Cryptographic Hardware and Embedded Systems – CHES 2017*, ser. LNCS, vol. 10529, 2017, pp. 45–68.

[8] H. Boyapally, D. Jap, Q. Wu, F. Zhang, and S. Bhasin, "Reality check on side-channels: Lessons learnt from breaking aes on arm cortex-a72 processor with out-of-order execution," *IACR Cryptology ePrint Archive*, no. 1381, 2024. [Online]. Available: https://eprint.iacr.org/2024/1381

[9] C. Organizers. "Ches 2025 challenge dataset." Accessed: 2025-10-07. [Online]. Available: https://ches2025.org/dataset

[10] T. Yap, S. Bhasin, and L. Weissbart, "Train Wisely: Multifidelity Bayesian Optimization Hyperparameter Tuning in Deep Learning-Based Side-Channel Analysis," in *Selected Areas in Cryptography – SAC 2024*, M. Eichlseder and S. Gambs, Eds., ser. Lecture Notes in Computer Science, vol. 15517, Springer, Cham, 2025. DOI: 10.1007/978-3-031-82841-6_12