Understanding Deep Learning RF Wireless Side-Channel Attacks: Perspectives on Noisy Scenarios

Bijia Cao
School of Physics and Electronics
Hunan University of Science and Technology
Xiangtan, Hunan, China
22020804001@mail.hnust.edu.cn

Yuanyuan Chen
School of Computer Science and Engineering
Hunan University of Science and Technology
Xiangtan, Hunan, China
2205010822@mail.hnust.edu.cn

Xiaoxia Wang
School of Physics and Electronics
Hunan University of Science and Technology
Xiangtan, Hunan, China
614064484@qq.com

Qixiang Ouyang
School of Physics and Electronics
Hunan University of Science and Technology
Xiangtan, Hunan, China
965275069@qq.com

Huanyu Wang*
School of Computer Science and Engineering
Hunan University of Science and Technology
Xiangtan, Hunan, China
huanyu@hnust.edu.cn

Junnian Wang
School of Physics and Electronics
Hunan University of Science and Technology
Xiangtan, Hunan, China
jnwang@hnust.edu.cn

Abstract

Recently developed Radio Frequency Wireless Side-Channel Attacks (RF-WSCAs) have become a significant threat to the widespread deployment of IoT edge devices as they eliminate the need for physical proximity to the target. By analyzing side-channel emissions from mixed-signal chips (unintentionally coupled with radio transmission signals), attackers can compromise cryptographic implementations over long distances. Existing work demonstrates the feasibility of extracting AES keys from Bluetooth devices at a distance of 15-metres using CNN models trained on traces. However, they may overestimate the threat of RF WSCAs by ignoring the effects caused by ambient noise and interference. In this study, we conduct a detailed examination of the extent to which environmental noise and interference can affect the attack efficiency of RF-WSCAs. We first proposed a 20 input CNN architecture that achieved similar attack results to the SOTA model. Then, we conducted a comprehensive analysis of RF-WSCAs from the perspective of noisy scenarios.

CCS Concepts

• **Security and privacy** → Security in hardware; Hardware attacks and countermeasures.

Keywords

RF Wireless Side-Channel Attacks, Noisy Scenarios, CNN, AES, Bluetooth Device

*Corresponding author.

CNSCT 2025, Zhengzhou, China

ACM ISBN 979-8-4007-1262-3/2025/01
https://doi.org/10.1145/3723890.3723921

ACM Reference Format:
Bijia Cao, Yuanyuan Chen, Xiaoxia Wang, Qixiang Ouyang, Huanyu Wang, and Junnian Wang. 2025. Understanding Deep Learning RF Wireless Side-Channel Attacks: Perspectives on Noisy Scenarios. In *2025 4th International Conference on Cryptography, Network Security and Communication Technology (CNSCT 2025), January 17–19, 2025, Zhengzhou, China.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3723890.3723921

1 Introduction

The widespread deployment of IoT edge devices makes them easy and attractive targets for adversaries to carry out malicious activities. Among the various threats, Side Channel Attacks (SCAs) are proven to be a potent way to compromise the security achieved by cryptographic algorithms. In the last decade, deep-learning techniques have become an ally of SCAs [9, 12]. Compared with traditional signal processing methods, well-trained deep learning models can improve the efficiency of attacks by several orders of magnitude [21].

Currently, two commonly used side channels in academia and industry are power consumption [7] and electromagnetic (EM) emission [15]. However, in the most commonly utilized SCAs, physical proximity to the victim is required, which may make the attack less practical in numerous scenarios. In 2018, [4] discovered a new type of side-channel called Radio Frequency (RF) wireless EM emissions or screaming channels. By exploring the coupling effect between the EM emission of the CPU and the radio carrier signals on the mixed-signal chip, the adversary has the ability to avoid direct physical access to the victim. In [4], by capturing 52K traces at a distance of 1 meter, the researchers are able to recover the secret key for the Bluetooth implementation of AES and obtain each of these traces by averaging 500 measurements of the identical encryption process. In the subsequent template attack [5], the secret key is retrieved from 5K traces captured at a distance of 15 m and each trace is obtained by averaging the identical encryption process's 1k measurements, while the key enumeration is still as high as 223. Then, using advanced deep-learning techniques, [22] succeeds in

extracting the AES key from 10K traces at a distance of 15 m from the victim device without repetitions. In 2021, [21] further uses "clean" traces captured via a coaxial cable with a controlled amount of additive noise to construct a neural network. Next, Zhang et al [10] employed a wireless repeater between the adversary and the victim to amplify the captured signals. In 2023, their follow-up work [11] proposes a multispectral capture method to obtain traces with an enhanced signal-to-noise ratio (SNR).

However, traces captured at long distances typically encounters a large amount of interference and noise, which makes it difficult for deep learning models to recognize and leads to a marked reduction in the efficiency of the attack. In this paper, we explore the extent to which noisy environments reduce the efficiency of attacks. Finally, our contribution is summarized below.

1. We propose a profiling-efficient CNN model for RF-WSCAs by focusing on the leakage-dominant segments of traces. Our experiments are performed by targeting nRF52 SoC implementations of AES-128. We show that by cutting the training traces from 110 sample points to 20, the proposed model achieves a comparable effectiveness of the attack to the state-of-the-art (SOTA) model with an 81.8% reduction in training resources.

2. we comprehensively analyze how noise can affect RF-WSCAs. We simulate five different noise levels for traces captured at the distance of 15 meters to the victim with two measuring scenarios.

2 Background

This section first outlines how EM emissions from the CPU coupled with the RF signals as side channels. Then, we present the concept of Deep Learning Side Channel Attack (DLSCAs).

2.1 RF Wireless EM Side Channel

Mixed-signal circuits, also known as Radio Frequency Integrated Circuits (RFICs), combine analogue and digital components in a single integrated circuit. However, this combination if implemented on the same silicon substrate introduces challenges. For example, while executing cryptographic algorithms, the digital component of the chip can generate both power and EM side channels based on running operations and data. The CPU core within the digital segment generates square wave noise due to the continuous switching of the clock signal. This square wave can modulate the side channel leakage. In the analog circuit, this modulated signal can accidentally mix with the Voltage-Controlled Oscillator 's (VCO) baseband signal [2]. Subsequently, the modulated signal is sent through an on-chip antenna. At the receiving side, an attacker can use a Software-Defined Radios (SDRs) to capture the transmitted leakage [20].

2.2 Deep Learning Side-Channel Attack

SCA techniques have undergone considerable development over the last two decades, as illustrated in [17]. Correlation Power Analysis (CPA) [1], initially one of the primary methods, has progressed along with technological advances including sophisticated methods such as deep-learning approaches. Deep learning (DL) techniques have gained immense popularity due to their exceptional ability to identify complex patterns and accurately classify from a wide range of datasets. Typically, compared to traditional signal processing methods, well-trained deep learning models can significantly improve the efficiency of an attack by several orders of magnitude [6, 8, 18]. Furthermore, various neural networks, including CNNs, are increasingly used by adversaries to overcome countermeasures against SCAs [3, 14]. With some exceptions [19], deep-learning techniques are primarily utilized for SCAs that is analyzed and usually consists of two phases: analysis and attack.

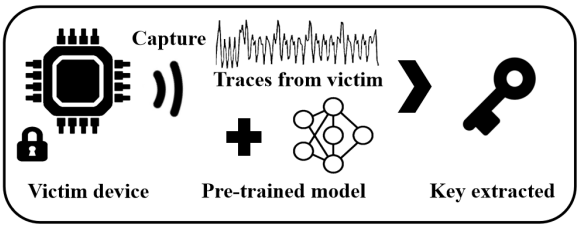

Figure 1: The overview of the profiling stage in DLSCAs.

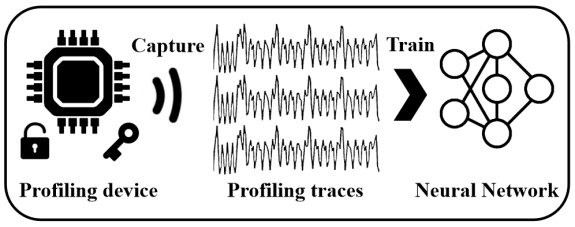

Figure 2: The overview of the attack stage in DLSCAs.

During the profiling stage, it is often assumed that adversaries have complete control one, or possibly more profiling devices that are the same to the victim's and are running the same cryptographic algorithm. This enables a significant amount of side-channel traces and related information can be collected by the attacker. An identified model undergoes training in deep learning, associating values of key-dependent with the traces by creating a leakage profile (as shown in Figure 1).

In the realm of SCAs, it is common that during the attack stage to presume that adversaries can acquire a limited quantity of side-channel traces and employ a pre-trained model to obtain the secret key. In RF-WSCAs, the requirement for physical access is waived as the attacker can obtain traces remotely (as shown in Figure 2).

3 Dataset

In the context of RF-WSCAs, the nRF52_Chip dataset is currently the only publicly available resource. The dataset is separated into two sections for profiling and testing. The profiling part comprises 200K traces collected from five nRF52832 DK implementations of Tiny-AES using a coaxial cable, with every trace representing the mean of the identical encryption operation's 100 measurements. In the following, we call these five nRF52 devices as *D1 – D5*. This portion of the dataset is primarily used for profiling purposes. Table 1 and Table 2 show the details of the analysis set and the initial test set, respectively.

Table 1: An overview of the profiling set. Each trace is captured via a coaxial cable with 100 repetitions.

Distance	Profiling device				
	D1	*D2*	*D3*	*D4*	*D5*
cable	40K	40K	40K	40K	40K

Table 2: Summary of the testing set. Each trace is captured at 15 m distance.

Distance	Victim device				
	D1	*D2*	*D3*	*D4*	*D5*
15m	5K	5K	5K	5K	5K

The testing part consists of 25K traces captured from a location that is 15 meters away from other five victim nRF52832 devices, utilizing an Ettus N210 Software Defined Radio (SDR) featuring a TP-Link Parabolic Antenna. In the following, we call these five nRF52 devices as *D6 – D10*. All traces in this part are captured in an office corridor environment. All traces are preprocessed using the *min-max scaling* technique, normalizing amplitudes to the range [0,1] for consistency and compatibility [13].

At the transmitter side in the dataset [21], they employ a Nordic Semiconductor nRF52832 development kit as the Bluetooth 5-enabled victim device. The SoC runs Tiny-AES encryption periodically at a data rate of 2 *Mbps*, broadcasting the ciphertexts via its on-chip antenna.

At the receiver side in [21], an SDR featuring a TL-ANT2424B grid parabolic antenna that offers a 24 *dBi* gain is utilized for capturing signals. 2.528 *GHz* is set to its center frequency, which is the sum of the center frequencies of the Bluetooth channels ($f_{\text{Bluetooth}} = 2.4GHz$) and the clock frequency of the victim's CPU ($f_{clock} = 64MHz$).

3.1 Evaluation Metric

In this paper, we use one of the most widely utilized evaluation metric, called Guessing Entropy (GE), to measure the attack efficiency. GE offers information about the degree to which the secret key is disclosed given a specific number of traces, which means the degree of indeterminacy or unpredictability linked to the conjecture of the secret key's exposure via side channels within the realm of side-channel attacks. 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 [16].

The quantity of subkey candidates that surpasses the actual probability of $\mathcal{K}_j$ within a set of traces $\mathcal{T}$ and its corresponding known information $\mathcal{X}$.

$$Rank\left(\mathcal{K}_j, \mathcal{T}\right) = \left|\left\{\mathcal{K}_j^{'} \in \mathcal{K} : Pr\left[\mathcal{K}_j|\mathcal{X}, \mathcal{T}\right] < Pr\left[\mathcal{K}_j^{'}|\mathcal{X}, \mathcal{T}\right]\right\}\right| \quad (1)$$

Generally, we classify the d trace sets $\mathcal{T}_1, \ldots, \mathcal{T}_d$ by using the trained model $\mathcal{M}_j$ and calculate the PGE value of the correct subkey by averaging the rank results in each set of tests.

$$PGE\left(\mathcal{K}_j\right) = \frac{\sum_{n=1}^{d} Rank\left(\mathcal{K}_j, \mathcal{T}_n\right)}{d} \quad (2)$$

As shown by the above formula (2), regarding each set of trace $\mathcal{T}_n$, the *rank* of $\mathcal{K}_j$ is denoted as $Rank(\mathcal{K}_j, \mathcal{T}_n)$.

4 Methodology

Existing deep learning-based RF-WSCA works [21, 22] on nRF52 SoC implementations of AES-128 typically train neural networks in trace segments containing all 16 SBox operations, although the model focuses exclusively on recovering the first subkey of AES. This approach may result in excessive training effort due to the inclusion of unrelated patterns from the remaining 15 SBox operations, which do not contribute directly to the extraction of the targeted subkey. We address this issue by isolating the relevant segment of the traces corresponding to the first SBox operation according to the leakage detection result, thereby reducing the computational overhead. Figure 3 shows an example of how leakage detection allocates PoIs for side-channel traces. The upper picture Figure 3 illustrates the leakage detection results for 16 subkeys in the last round of nRF52 implementations of AES-128. To obtain leakage results, we use a total of 5K profiling traces and these traces are separated into two sets based on the HW of the corresponding label: $HW(label) > \beta$ and $HW(label) < \beta$. In this case, the label is determined to the last round AES encryption 's the first SBox output value. The bottom picture Figure 3 shows an example trace that represents 16 SBox operations in the last round. The dashed red line illustrates the PoI allocation for the first SubByte operation in the AES-128 's first round. We use the tagged trace segment to train the model.

To build the profile between the selected 20-point trace segment and the key-dependent intermediate label, we use Convolutional Neural Network (CNN) in our experiments with the layer structures shown in Table 3. We quantify the network's classification error by using the categorical cross-entropy loss. For accomplishing the reduction of the categorical cross-entropy loss to a minimum, we compute the gradient of the categorical cross-entropy loss regarding scores, propagate it backwards via the network to adjust the network's internal parameters using the optimization algorithm of RMSprop, using a learning rate set to 0.00001 and without any decay in its value. The maximum duration of the training process 200 epochs, using a batch size of 256. Based on the leakage detection result presented in Figure 3, the model's input size has been configured to 20. We use the output of the first SBox.

Afterwards, we test the trained model on 10 different testing sets, originated from the testing set of the nRF52_Chip dataset. To simulate different noise levels of the attacking environment, we artificially add Gaussian noise with different signal strength levels to the testing traces captured 15-meter away from the victim devices. Next, we compare the results and analyze to what extent the environment can affect model's performance.

5 Experimental Setup

In this section, we mainly Concentrate on how we generate the 10 testing sets to simulate traces captured remotely under different

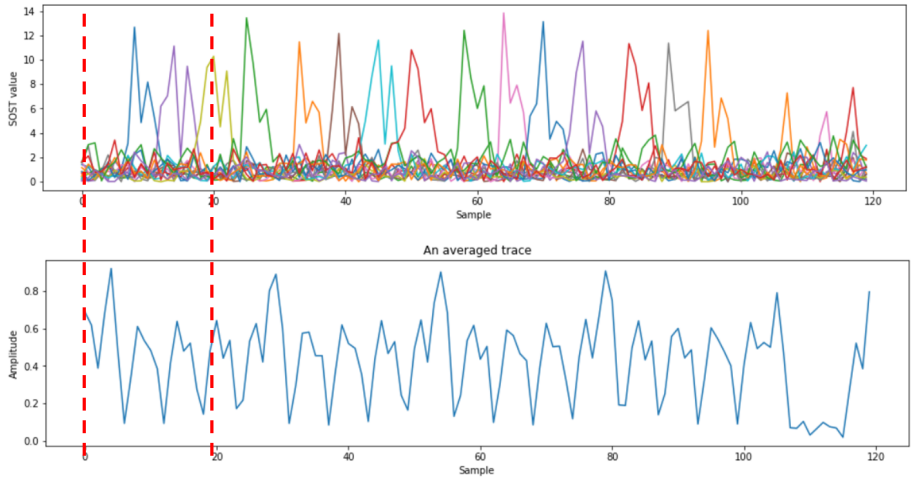

Figure 3: Locating specific trace segment of first SBox operation in the last round of AES. The top plot reveals the leakage detection results of 16 SBox operations in the last round of AES. The bottom plot is an example profiling trace with 110 sample points. The dashed red line illustrates the 20-point segment used by the proposed model.

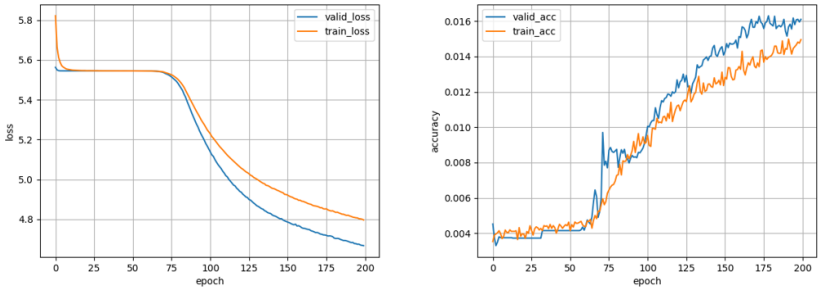

Figure 4: Training vs. validation loss and accuracy of the proposed *CNN* model on profiling traces.

Table 3: The proposed *CNN* architecture.

Layer (Type)	Output Shape	Parameter
Conv 1 (Conv1D)	(None, 20, 16)	64
Batch Norm1	(None, 20, 16)	64
Average Pooling 1	(None, 19, 16)	0
Conv 2 (Conv1D)	(None, 19, 16)	784
Batch Norm1	(None, 19, 16)	64
Average Pooling 2	(None, 18, 8)	0
Flatten	(None, 288)	0
Dropout1	(None, 288)	0
Dense	(None, 256)	73984
Dropout2	(None, 256)	0
Output (Dense)	(None, 256)	65792
Total Parameters: 130,752		

environment. In the nRF52_Chip dataset, there are two original testing sets which represent two scenarios:

1. Traces are captured from the victim nRF52 devices with every trace is the mean of the identical encryption 's 100 measurements (100 repetitions) at a 15-meter distance. This set is for the scenario where adversaries may have the capacity to measure the same encryption.

Table 4: Summary of the derived 10 testing sets $S1 - S10$. We add the artificially generated Gaussian noise trace with a fixed standard deviation σ to the original testing set of the nRF52_Chip dataset. Each trace is captured at 15 m distance.

	S1	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*	*S8*	*S9*	*S10*
#repetition	100	100	100	100	100	1	1	1	1	1
σ	0	0.01	0.06	0.08	0.11	0	0.01	0.06	0.08	0.11

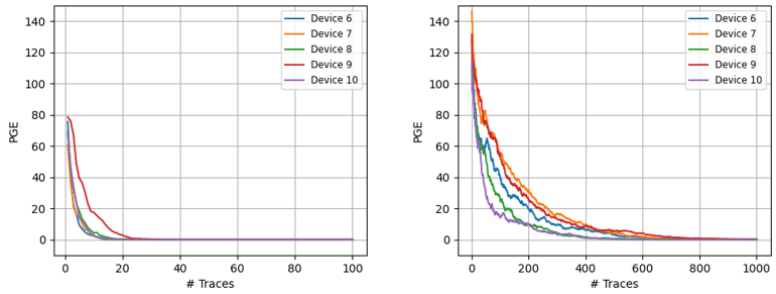

Figure 5: The ranking results of the model attacking *D6-D10* devices (100 repetitions vs 1 repetition).

Table 5: Average trace count needed to recover the first subkey by using the 20-input CNN at 15 m distance (for 50 tests). Each trace represents the average of N measurements using identical encryption.

	D6	*D7*	*D8*	*D9*	*D10*	Average
N=100	10	9	11	18	10	11.6
N=1	448	401	256	487	171	352.6

2. Traces are captured from the victim nRF52 devices without repeating encryption many times (1 repetitions) at a 15-meter distance. This set is for the scenario where adversaries have no access to victims.

In this section, we further derive 10 testing datasets (called $S1 - S10$) from the two described parts. We define five different noise levels and generate the corresponding Gaussian noise traces. Afterwards, we add the artificially generated Gaussian noise trace to the two testing sets, using a mean $\mu = 0$ and standard deviation $\sigma = 0,\ 0.01,\ 0.06,\ 0.08,\ 0.11$, corresponding to five noise levels (NO Noise, Noise 1-4), to derive 10 testing datasets, to derive 10 testing datasets. The details of these 10 testing sets are listed in Table 4.

6 Experimental result

In this section, we first train the model on profiling traces as shown in Table 1, with 20% of traces randomly set aside for validation. Figure 4 compares the accuracy and loss of the model throughout the training and validation phases. The training process was completed using an NVIDIA GeForce RTX 3070 graphics card. Afterwards, we test the trained model on traces captured at 15 m distance. As shown in Table 5, when traces are with 100 repetitions, the model requires 11.6 traces to compromise the implementation. For the case of 1 repetition, the model is feasible to extract the secret key by using 352.6 traces. Figure 5 shows the detailed ranking results of the trained model when attacking devices *D6-D10*, evaluated under the conditions of 100 and 1 trace repetitions, respectively. The figure clearly demonstrates that with 100 repetitions, the model can successfully recover each device's key with a minimal number of traces. Without repetitions, the model can still recover the keys for each device with a few hundred traces. By comparing the SOTA results in [21], our model can achieve a comparable attack result, but can save 81.8% training cost during the training stage. This is mainly due to the focusing strategy during the leakage detection process. Each result in the table is the average of 50 tests.

6.1 Effect of Noise

We set five noise levels according to the standard deviation σ and used the trained model to test on 10 test sets. This includes traces at five noise levels with 100 repetitions and a test set with only 1 repetition Table 6 and Table 7 show how many traces are needed for each device to successfully recover keys under different noise levels when traces are repeated 100 times and once, respectively. As can be seen from the tables, whether it is 100 repetitions or 1 repetition, different noise intensities have varying degrees of effect on the attack results. The model requires an exponentially growing number of traces as the noise level increases. Figure 6 shows the average rank of recovering five target devices. It highlights that in high-noise environments, even if an adversary can measure the same encryption scenario, they may not improve attack efficiency by averaging repeated measurements.

Additionally, we calculate the average number of traces required to successfully attack devices *D6* to *D10* at different noise levels. And based on *No Noise*, the reduction in attack efficiency of the model in different noise intensity environments is calculated. Figure

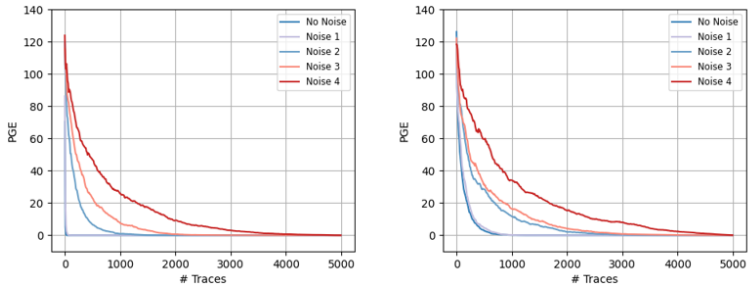

Figure 6: The average rank attack results for devices *D6-D10* across different noise levels.

Table 6: The number of traces (100 repetitions) for successfully recovering the key on *D6-D10* devices at different noise levels.

	No Noise	*Noise 1*	*Noise 2*	*Noise 3*	*Noise 4*
D6(S1)	10	21	835	1251	2090
D7 (S2)	9	23	575	1126	4707
D8 (S3)	11	21	473	1121	2177
D9 (S4)	18	40	606	1221	4258
D10 (S5)	10	16	413	814	1241

Table 7: The number of traces (1 repetitions) for successfully recovering the key on *D6-D10* devices at different noise levels.

	No Noise	*Noise 1*	*Noise 2*	*Noise 3*	*Noise 4*
D6(S6)	448	451	2814	3175	4681
D7 (S7)	401	531	1448	1960	3866
D8 (S8)	256	324	1032	1745	3878
D9 (S9)	487	585	1767	1984	4767
D10 (S10)	171	204	392	1107	1501

7 shows that within the current noise level range, the advantage in the average number of traces required to recover the key with 100 repetitions is greater with 100 repetitions of traces than with one repetition, but this advantage is gradually decreasing. This confirms that when the environmental noise intensity reaches a certain level, the averaged repeated measurements on the target device becomes meaningless.

For different noise environments, the reduction in attack efficiency is shown in Table 8. When traces are with 100 repetitions or just 1 repetition, successfully recovering the key requires an exponentially increasing number of traces as the noise level rises. As shown, the average number of traces required for the *S1-S5* test set increases faster than the *S6-S10* test set. This demonstrates the fragility of data sets with repeated measurements and averages in high-noise environments.

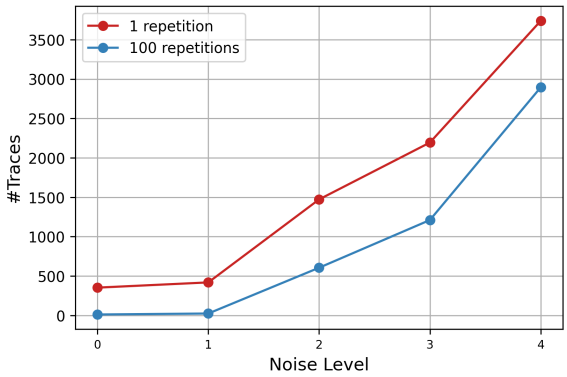

Figure 7: Trend of average traces growth for attacking devices *D6-D10* across different noise levels.

7 Conclusion

In this paper, we first train a CNN model with 20 inputs, reducing training costs by 81.8%. We then conduct a comprehensive analysis of how noise affects test sets in two different scenarios (100 repetitions and 1 repetition). As the intensity of the noise increases, the number of traces required for attack implementation, especially in the 100 repetitions scenario, increases significantly. This indicates that when noise reaches a certain level, the advantage in trace count required for attacks disappears. Therefore, reducing noise in the attack environment is essential, especially in the 100 repetitions scenario.

Table 8: Using the average number of traces required to successfully attack devices *D6-D10* in a noise-free scenario as the baseline to calculate the rate of increase in the number of traces under different noise levels.

	Noise 1	*Noise 2*	*Noise 3*	*Noise 4*
N=100	108.6%	5116.4%	10333.3%	24831.0%
N=1	18.8%	317.8%	522.4%	961.0%

References

[1] Brier, E. *et al.* 2004. Correlation power analysis with a leakage model. *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)* (2004), 16–29.

[2] Bronckers, S. *et al.* 2010. *Substrate noise coupling in analog/RF circuits*. Artech House.

[3] Cagli, E. *et al.* 2017. Convolutional neural networks with data augmentation against jitter-based countermeasures: Profiling attacks without pre-processing. *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)* (2017), 45–68.

[4] Camurati, G. *et al.* 2018. Screaming channels: When electromagnetic side channels meet radio transceivers. *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)* (2018), 163–177.

[5] Camurati, G. *et al.* 2020. Understanding screaming channels: From a detailed analysis to improved attacks. *IACR transactions on Cryptographic Hardware and Embedded Systems (TCHES)*. (2020), 358–401. DOI:https://doi.org/10.46586/tches.v2020.i3.358-401.

[6] Das, D. *et al.* 2019. X-DeepSCA: Cross-device deep learning side channel attack. *ACM/IEEE Design Automation Conference (DAC)* (2019), 1–6.

[7] Gao, Y. *et al.* 2024. Deeptheft: Stealing dnn model architectures through power side channel. *2024 IEEE Symposium on Security and Privacy (SP)* (2024), 3311–3326.

[8] Hajra, S. *et al.* 2023. On the instability of softmax attention-based deep learning models in side-channel analysis. *IEEE Transactions on Information Forensics and Security (TIFS)*. (2023).

[9] He, D. *et al.* 2024. Improving IIoT security: Unveiling threats through advanced side-channel analysis. *Computers & Security*. (2024), 104135.

[10] Hong-Yi, Z. *et al.* 2022. Wireless side-channel analysis method based on repeater. *Journal of Cryptologic Research*. 9, 1 (2022), 175–188.

[11] Hong-Yi, Z. *et al.* 2023. Wireless side-channel analysis method based on spectral addition. *Journal of Cryptologic Research*. 10, 4 (2023), 862–878.

[12] Hu, J. *et al.* 2023. Password-stealing without hacking: Wi-fi enabled practical keystroke eavesdropping. *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)* (2023), 239–252.

[13] Juszczak, P. *et al.* 2002. Feature scaling in support vector data description. *Proc. asci* (2002), 95–102.

[14] Kim, J. *et al.* 2019. Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*. (2019), 148–179. DOI:https://doi.org/10.46586/tches.v2019.i3.148-179.

[15] Ninan, M. *et al.* 2024. A second look at the portability of deep learning side-channel attacks over EM traces. *Proceedings of the International Symposium on Research in Attacks, Intrusions and Defenses (RAID)* (2024), 630–643.

[16] Pahlevanzadeh, H. *et al.* 2016. Assessing CPA resistance of AES with different fault tolerance mechanisms. *2016 21st Asia and South Pacific design automation conference (ASP-DAC)* (2016), 661–666.

[17] Picek, S. *et al.* 2023. SoK: Deep learning-based physical side-channel analysis. *ACM Computing Surveys*. 55, 11 (2023), 1–35. DOI:https://doi.org/10.1145/3569577.

[18] Staib, M. and Moradi, A. 2023. Deep learning side-channel collision attack. *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*. (2023), 422–444. DOI:https://doi.org/10.46586/tches.v2023.i3.422-444.

[19] Timon, B. 2019. Non-profiled deep learning-based side-channel attacks with sensitivity analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*. (2019), 107–131. DOI:https://doi.org/10.46586/tches.v2019.i2.107-131.

[20] Wang, H. 2024. Amplitude-modulated EM side-channel attack on provably secure masked AES. *Journal of Cryptographic Engineering*. 14, (2024), 537–549. DOI:https://doi.org/10.1007/s13389-024-00347-3.

[21] Wang, R. *et al.* 2021. Advanced far field EM side-channel attack on AES. *Proceedings of the 7th ACM on Cyber-Physical System Security Workshop (CPSS)* (2021), 29–39.

[22] Wang, R. *et al.* 2020. Far field EM side-channel attack on AES using deep learning. *Proceedings of the 4th ACM Workshop on Attacks and Solutions in Hardware Security (ASHES)* (2020), 35–44.