CL-SCA: Leveraging Contrastive Learning for Profiled Side-Channel Analysis

Annv Liu, An Wang, Shaofei Sun, Congming Wei, Yaoling Ding, Yongjuan Wang, and Liehuang Zhu *Member, IEEE*

***Abstract*—Side-channel analysis based on machine learning, especially neural networks, has gained significant attention in recent years. However, many existing methods still suffer from certain limitations. Despite the inherent capability of neural networks to extract features, there remains a risk of extracting irrelevant information. The heavy reliance on profiled traces makes it challenging to adapt to remote attack scenarios with limited profiled traces. Besides, attack traces also contain critical information that can be used in the training process to assist model learning. In this paper, we propose a side-channel analysis approach based on contrastive learning named CL-SCA to address these issues. We also leverage a stochastic data augmentation technique to assist model to effectively filter out irrelevant information from the profiled traces. Through experiments of different datasets from different platforms, we demonstrate that CL-SCA significantly outperforms various conventional machine learning side-channel analysis techniques. Moreover, by incorporating attack traces into the training process using our approach, known as CL-SCA+, it becomes possible to achieve even greater enhancements. This extension can further improve the effectiveness of key recovery, which is fully verified through experiments on different datasets.**

***Index Terms*—Side-channel analysis, Contrastive learning, Neural networks, Data augmentation.**

I. Introduction

THE widespread use of Internet of Things (IoT) technology [1] has raised significant concerns about the safety of smart devices. For IoT smart devices, side-channel analysis (SCA) has become one of the most well-known threats. This technique is first proposed in 1996 [2], which has been extensively studied over the past three decades.

Cryptographic devices may unintentionally leak physical information during the execution of cryptographic algorithms, such as power consumption [3], electromagnetic emissions [4], and time deviation [5]. SCA exploits the physical information to recover the key of the cryptographic algorithm. By combining physical observations with the hypothesis of relevant intermediate values or manipulated operations, it is possible to recover the intermediate state of the device. Subsequently, an attack can be launched against the target device to recover the key.

Profiled side-channel analysis is regarded as one of the most threatening techniques of SCA [6], [7]. It assumes that the attacker has a profiled device identical to the target device, and can construct the template of profiled device's leakage traces to recover the key of the target device. Template attack is a traditional profiled side-channel analysis technology [8]. It uses the mean vectors and a covariance matrix to build a template, and then uses the traces of the target device to match it to recover the key. However, it should be noted that template attack is primarily suitable for handling low-dimensional data. This technique heavily relies on preprocessing to reduce the dimensionality of traces.

In 2011, Hospodar *et al.* noted that profiled side-channel analysis can be seen as a classification problem [9]. This observation paved the way for utilizing machine learning techniques to address this issue. Several machine learning methods, including support vector machine (SVM) [9]–[13], random forest (RF) [14], rotation forest [15]–[17], decision tree [15], [16], naive Bayes [13], [15], [18] and so on have been applied in the SCA community to recover the key. These methods have demonstrated higher efficiency and improved performance compared to traditional approaches. Importantly, machine learning methods have shown promise in overcoming the limitations of traditional template attacks, which struggle to handle high-dimensional data [19]. To improve key recovery effectiveness, these techniques often incorporate various feature extraction methods, including principal component analysis (PCA) [20] and linear discriminant analysis (LDA) [21], to extract informative features.

As a type of machine learning technology, neural networks have been extensively studied in the field of SCA due to their ability to accurately identify critical features in the leakage traces. This mitigates the need for feature extraction technologies, as neural networks themselves can achieve good results. Numerous studies have demonstrated that neural networks can construct more effective models compared to traditional machine learning approaches in SCA [22]. Multilayer perceptron (MLP) stands out as an effective method for optimizing power attack performance, surpassing conventional techniques like differential power attack (DPA) [23] and simple power attack [24]. Weissbart *et al.* noted that MLP has a much simpler structure, enabling easier hyperparameter tuning and contributing to the explainability of neural network inner working [25]. Convolutional neural networks (CNN) has also gained widespread application in SCA [26]–[29]. It has been proved that CNN is effective in

Annv Liu, An Wang, Congming Wei, Yaoling Ding and Liehuang Zhu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China. (Email:{Annvliu, wanganl, weicm, dyl19, liehuangz}@bit.edu.cn).

Shaofei Sun is with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081; and also with Henan Key Laboratory of Network Cryptography Technology, Zhengzhou 450001, China. (Email:sfsun@bit.edu.cn).

Yongjuan Wang is with Institute of Cyberspace Security, Information Engineering University, Zhengzhou 450001, China. (Email:pinkywyj@163.com).

(Corresponding author: Shaofei Sun).

Manuscript received April 19, 2021; revised August 16, 2021.

attacking unaligned traces or cryptographic algorithms with random delay countermeasures [26]. Furthermore, the classical long short term memory architecture has been employed to encode traces implemented by the AES on FPGA, resulting in a nearly tenfold improvement in DPA efficiency [30].

Although neural networks have been widely applied to SCA and have achieved significant results, current technologies still have some limitations. Firstly, for remote attack scenarios, the number of profiled traces is limited due to collection difficulties. The effectiveness of SCA methods based on neural network will be affected. Secondly, previous researches have mainly focused on the profiled device and rarely utilized the traces from the target device for training, leading to a potential loss of valuable information.

To address these issues, we have applied the contrastive learning technique to SCA. The use of this technique has become widespread in various fields, including computer graphics and natural language processing. Furthermore, established learning frameworks, such as SimCLR [31], [32] and MoCo [33], [34] have emerged to facilitate its adoption. Contrastive learning divides the classification task into two stages: pretrain and subsequent fine-tuning. This approach enables the extraction of features in unsupervised scenarios, followed by leveraging a small amount of labeled data to achieve robust classification results. In this paper, we apply the idea to SCA and propose a novel SCA approach based on contrastive learning named CL-SCA. To the best of our knowledge, this is the first time that a contrastive approach has been used in profiled side-channel attacks.

The main contributions are summarized as follows:

- We propose a side-channel analysis approach based on contrastive learning named CL-SCA. This approach extracts features in an unsupervised setting, which improves the effectiveness of key recovery. The experimental results show that CL-SCA can recover the secret key successfully with a great advantage in different datasets.
- We introduce a stochastic data augmentation technique specifically tailored to bolster CL-SCA's feature extraction capabilities. By effectively mitigating interference from irrelevant information, this technique enables CL-SCA to concentrate exclusively on extracting critical leakage features imperative for successful key recovery.
- In addition to the profiled traces, we also incorporate the target device's traces into the training process of CL-SCA, which helps to learn more useful information. By assimilating the leakage information beforehand, CL-SCA gains a profound understanding of the patterns within the attack trace set, ultimately achieving a substantial improvement in the effectiveness of key recovery.

The organization of this paper is as follows: Section II introduces the preliminary knowledge of profiled side-channel analysis, convolutional neural network and contrastive learning. Section III shows the stochastic data augmentation technique and CL-SCA in detail, and further proposes CL-SCA+. Section IV describes the experimental results, and Section V presents the conclusion.

II. Preliminaries

A. Profiled Side-Channel Analysis

Profiled side-channel analysis leverages the inherent dependence between a cryptographic device's leakage traces and the data it processes. This technique involves two main stages: a profiled stage and an analysis stage. During the profiling stage, the model for the secret value dependent form of the cryptographic device is constructed with leakage traces. Then, the secret value of the target device can be revealed by comparing the traces collected from the target device with the model prediction during the analysis stage.

The attacker can control a profiled device that is similar to the target device during the profiling stage. Using this device, the attacker can select the plaintext p and the secret key k to execute the cryptographic algorithm and collects the corresponding trace t. There is an association between t and the intermediate value y used during the execution of the cryptographic algorithm. For example, in the case of the AES algorithm, the output of the Sbox is usually used as the intermediate value y, and y can be expressed as Eq.(1).

$$y = Sbox(p \oplus k) \tag{1}$$

During the profiling stage of profiled side-channel analysis, the model $\varphi(\cdot)$ is established to map the trace to different intermediate values of the cryptographic algorithm. Using the model, the probability of each trace corresponding to different intermediate values y can be calculated, as shown in Eq. (2).

$$\begin{aligned} \varphi(t) &= \Pr[Y = y \mid t] \\ &= \Pr[Y = Sbox(p \oplus k) \mid t] \end{aligned} \tag{2}$$

During the analysis stage, the attacker collects the traces of the target device with different plaintexts. Then the attacker matches each trace with the model $\varphi(\cdot)$ to obtain the probability of different intermediate values $\Pr[Y = y \mid t]$ for different key candidates $k \in \mathcal{K}$, where $\mathcal{K}$ is the set of all possible key candidates $\mathcal{K} = \{0, 1, \cdots, 255\}$. By calculating the result of Eq.(3), the profiled side-channel analysis can identify the key corresponding to the highest log-likelihood estimation score, denoted as k_r. The recovered key k_r is considered as the key of the target device.

$$L(k) = \sum_{i=1}^{N} \log \Pr(t_i; k) \tag{3}$$

where N denotes the number of collected traces from the target device. If the recovered key k_r is same as the correct key k^*, then the analysis is considered to be successful.

B. Convolutional Neural Network

Profiled SCA can be viewed as a classification problem, which makes it easier to employ deep learning approaches. Among these techniques, CNN has emerged as a widely adopted solution within the SCA community for accurately classifying leakage traces. Fig. 1 shows an example of trace classification using a CNN. Similar to profiled side-channel analysis, CNN also calculates the probabilities of each trace corresponding to different intermediate values.

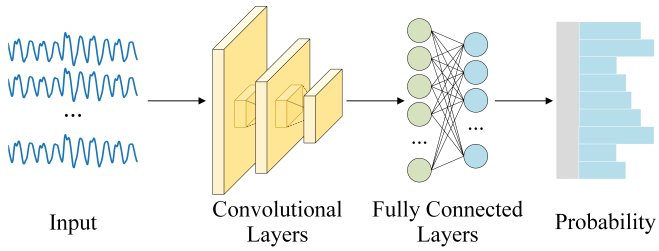

Fig. 1. The schema of CNN.

CNN is typically composed of three main types of layers: convolutional layers, pooling layers, and fully connected layers. The convolutional layer applies filters to the input trace to generate feature maps. Each filter convolves with a small input patch, producing a single value placed in the corresponding position of the output feature map. The convolutional layer produces multiple feature maps representing different filters, which are then passed to the next layer for further processing.

The pooling layer is typically applied after one or more convolutional layers. Its primary purpose is to reduce the spatial dimensions of the feature maps produced by the convolutional layers, while also preserving the most important information. There are various types of pooling operations, but for this paper, we will only focus on average pooling. The output of this layer is the average value of a specific area in the feature map.

Fully connected layers process the entire input volume, unlike convolutional and pooling layers which perform local operations on specific regions of the input data. The input to the fully connected layer usually consists of a flattened feature map that has been produced by one or more preceding convolutional and pooling layers. The flattened feature map is a one-dimensional array of values that represent the features detected in the input trace. The fully connected layer then applies a series of matrix operations to these features to generate a set of output values that correspond to the predicted class probabilities.

C. Contrastive Learning

Self-supervised learning is a type of neural network technology that can learn the representations of data without the need for human-labeled annotations. The main advantage of self-supervised learning lies in its ability to derive meaningful representations from large quantities of unlabeled data, which can then be fine-tuned on smaller labeled datasets for downstream tasks.

Contrastive learning is a type of self-supervised learning that involves training an encoder $f(\cdot)$ to learn representations of data by contrasting positive and negative examples. The goal of contrastive learning is to learn a representation space where similar samples are mapped close together and dissimilar samples are mapped far apart, even if Eq.(4) holds true. Fig. 2 illustrates this process.

$$score(f(x), f(x^+)) >> score(f(x), f(x^-)) \tag{4}$$

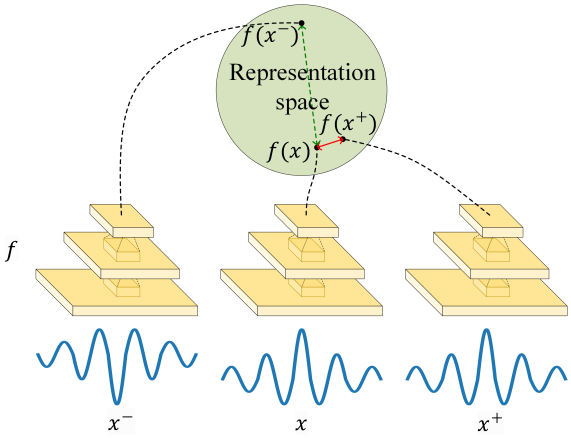

Fig. 2. The process of contrastive learning.

Here, x^+ represents a positive sample that shares similarities with the sample x, and the pair consisting of x^+ and x is regarded as a positive pair. x^- represents a sample that is dissimilar to x, and it forms a negative pair with x. The $score$ is a measure of similarity between representations, and the function $f(\cdot)$ transforms the samples into their corresponding feature vectors.

In this paper, we use a stochastic data augmentation technique to augment each trace into two different traces. The augmented traces have the same intermediate value, and are therefore considered as a positive pair. The traces augmented from different original traces are considered negative samples.

The contrastive loss function is also an important component. We use the normalized temperature-scaled cross entropy loss as loss function, which is named *NT-Xent*. *NT-Xent* takes a pair of traces and their labels as input, and computes a loss that encourages the network to learn representations that separate the positive and negative pairs. The loss function penalizes the distance between the representations of positive pairs, and rewards the distance between the representations of negative pairs.

Overall, contrastive learning is a powerful technique for unsupervised learning, which is applied to side-channel analysis in this paper. By learning useful representations in an unsupervised manner, we prove that contrastive learning can help improve the performance of supervised side-channel analysis based on machine learning.

III. Proposed Side-Channel Analysis Approach Based on Contrastive Learning

In this section, we provide a detailed introduction to our proposed side-channel analysis approach. The process of our approach is illustrated in Fig. 3, which includes two parts: pretraining and fine-tuning. In the pretraining part, each trace undergoes augmentation twice using stochastic data augmentation technique. The objective of this part is to train a feature extraction encoder that encourages similarity between the two augmented traces. In the fine-tuning part, a neural network for classification is built based on the trained encoder. By training on the profiled set, the network can classify the traces in the

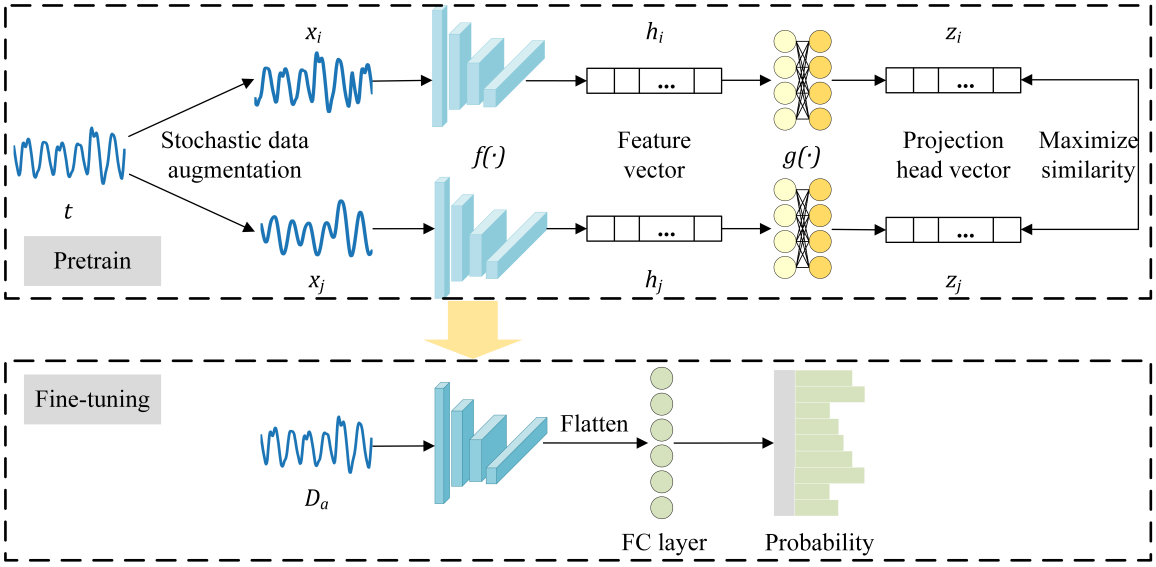

Fig. 3. The basic flow of CL-SCA.

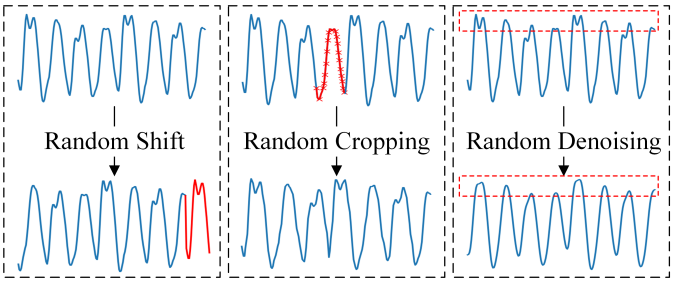

Fig. 4. The illustrations of stochastic data augmentation techniques

attack set and recover the key. The specific details of each part are described in the following subsections.

A. Stochastic Data Augmentation

Data augmentation has been shown to be an effective method for improving the capabilities of neural networks [26]. The first step of our proposed approach is also to perform stochastic data augmentation on each trace in the input dataset. Our goal is to generate two distinct augmented traces from a common original trace t as a positive pair, denoted as x_i and x_j. We aim to create variations in the augmented traces while preserving the critical features for accurate classification. By doing so, the encoder can effectively extract the commonalities between the positive pairs during the pretraining part, so that it can identify the essential features for classification and discard irrelevant information.

To achieve this goal, we employ randomness in our data augmentation approach. The occurrence of augmentation can be more unpredictable, and so that we can generate different augmented traces. As shown in Fig. 4, our stochastic data augmentation technique includes three types of data augmentation: random shift, random cropping, and random denoising. The procedure for each data augmentation technique is as follows:

1) Random Shift (RS): Given the maximum range for left or right shifts L_s, the size of each shift is random. The specific length of movement l_s satisfies the Eq. (5), where l_s is a positive number for a left shift, and a negative number for a right shift.

$$l_s = random(-L_s, L_s) \tag{5}$$

2) Random Cropping (RC): Cropping occurs with a probability of 50%. If cropping occurs, the position of the cropping p_c is random and determined by Eq. (6). L_c in the Eq. denotes the size of the cropping, and L denotes the length of the original trace. After cropping, it is essential to stretch the trace. This involves uniformly filling the cropped trace to ensure that its length remains consistent with its original length.

$$p_c = random(0, L - L_c) \tag{6}$$

3) Random Denoising (RD): Given the weighting factor W, denoising occurs randomly. The calculation process is shown in Eq. (7). The notation t_i^j in this equation represents the j-th point of the original trace t_i, while x_i^j represents the j-th point of the trace x_i after denoising.

$$x_i^j = \frac{u_i^j + W \times u_i^{j+1}}{W+1}, u_i^j = \frac{t_i^j + W \times t_i^{j-1}}{W+1} \tag{7}$$

B. SCA Approach based on Contrastive Learning

This section presents the proposed approach for CL-SCA, which consists of two main parts: pretraining and fine-tuning. In the pretraining part, we leverage stochastic data augmentation techniques to effectively capture the valuable features from the traces. To achieve this, we train the encoder $f(\cdot)$ on N_p traces obtained from the profiled device in an unsupervised setting. It is important to note that these traces and their corresponding labels can form the profiled set D_p. Our encoder $f(\cdot)$ is trained using mini-batch gradient descent. The following is a detailed description of CL-SCA for training:

(1) Perform stochastic data augmentation: Perform two rounds of stochastic data augmentation on each trace t_i from mini-batch to obtain two augmented traces x_i and x_j.

(2) Encode with the encoder $f(\cdot)$: Use the encoder $f(\cdot)$ to encode the augmented traces x_i and x_j as feature vectors h_i and h_j, respectively.

(3) Perform the projection head $g(\cdot)$: Encode the feature vectors h_i and h_j into the projection head output vectors z_i and z_j using the projection head function $g(\cdot)$.

(4) Calculate the loss and train the encoder $f(\cdot)$: Calculate the similarity between z_i and z_j as loss function. Then apply backward training to the encoder $f(\cdot)$ to adjust its weights with the objective of maximizing this similarity.

The stochastic data augmentation technique used in step (1) has been discussed in the previous. By maximizing the similarity between the feature vectors for the positive pair, the encoder is encouraged to focus on the critical leakage features and ignore irrelevant features for classification, which can improve the accuracy and robustness of fine-tuning network.

The projection head employed in step (3) is based on a neural network. Specifically, the projection head is implemented as an MLP with one hidden layer and ReLU activation function. In the fine-tuning part, however, only the feature vectors before the projection head are used for training the model. But the introduction of projection heads in pretraining part still improves the effectiveness of fine-tuning. This is because maximizing the similarity of positive sample pairs during the pretraining part makes the relevant feature vectors of positive sample pairs being trained to be invariant. If the projection head is not utilized, maximizing the similarity will inadvertently erase information that is beneficial for fine-tuning, resulting in a decrease in performance.

In step (4), in order to calculate the similarity for a positive pair, we use $\|\cdot\|$ to represent ℓ_2 regularization, and define the cosine similarity of two vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as:

$$sim(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{(\alpha)^T \beta}{\|\alpha\| \|\beta\|}. \tag{8}$$

We define the size of the mini-batch as n. After data augmentation, there are n positive pairs, i.e. $2n$ augmented traces. For a positive pair, we treat $2(n-1)$ augmented traces within a mini-batch as negative examples. Utilizing cosine similarity $sim(\boldsymbol{\alpha}, \boldsymbol{\beta})$ introduced earlier, we define a loss function for the relevant vectors (z_i, z_j) of a positive pair as:

$$\ell(z_i, z_j) = -\log \left(\frac{\exp\left(\frac{sim(z_i, z_j)}{\tau}\right)}{\sum_{k=1}^{2n} \mathbb{I}_{[k \neq i]} \exp\left(\frac{sim(z_i, z_k)}{\tau}\right)} \right), \tag{9}$$

where $\mathbb{I}_{[k \neq i]}$ is an indicator function evaluating to 1 iff $k \neq i$. τ represents a temperature parameter which is set to 0.07 in our work. The numerator in the log calculates the similarity of positive pair (z_i, z_j), while the denominator calculates the similarity of negative examples of z_i. By minimizing this loss, the encoder can extract common critical features that are still useful for classification after data augmentation. The final loss function used during pretraining, which is termed as *NT-Xent* in previous work, is the average loss of all positive pairs in a mini-batch:

$$\mathcal{L} = \frac{1}{2n} \sum_{i=1}^{n} [\ell(z_i, z_j) + \ell(z_j, z_i)]. \tag{10}$$

After training multiple epochs in the previous steps, the encoder $f(\cdot)$ can unsupervisedly extract critical features from the profiled set.

In the fine-tuning part, we utilize the encoder from the pretraining part for profiled side-channel analysis. We add a fully connected layer to the trained encoder $f(\cdot)$ to construct the neural network. The process of fine-tuning part is to train this required neural network using the labeled profiled trace set D_p and evaluate its performance on N_a traces from the target device, which constitute the attack trace set D_a.

C. CL-SCA+

While CL-SCA is effective in extracting crucial features, there is still a potential waste of information. Specifically, traces in the attack set also contain useful information that have not been utilized. To address this problem, we propose incorporating the attack trace set into the training process and thereby introducing the CL-SCA+.

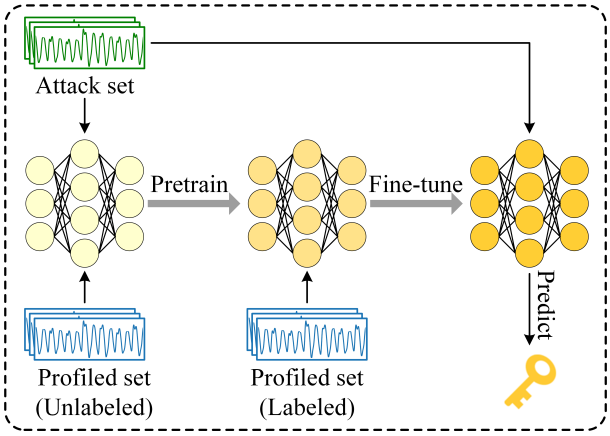

Fig. 5. The process of CL-SCA+.

We observe that the pretraining part of CL-SCA is unsupervised. The pretraining only augments the traces of the profiled set, and then trains the encoder to find the commonalities of the positive pairs. This process does not require any labels of the profiled set. Therefore, we can extend the original input dataset to D, which is composed of the profiled trace set D_p and the attack trace set D_a without any labels. Through this, we can enable the model to learn the useful information of the attack trace set during the pretraining part, thereby improving the effectiveness of key recovery.

Fig. 5 illustrates the CL-SCA+ process. This process begins by utilizing unlabeled profiled trace set and attack trace set to pretrain an encoder. This encoder is responsible for extracting the valuable features. Subsequently, a neural network is constructed based on this pretrained encoder and fine-tuned using a labeled profiled trace set. Finally, the fine-tuned neural network is employed to classify the traces present in the attack trace set, ultimately leading to the key recovery.

TABLE I
HYPERPARAMETERS

Network Architecture	
Convolution Blocks	5
Filters	{64, 128, 256, 512, 512}
Kernel Size	11
Activation Function	ReLU
Pooling Layer	Average
Fully Connected Layers	2
Neurons	4096
Petrain Hyperparameters	
Optimizer	Adam
Weight Decay	0.0001
Learning Rate	0.00001
Loss Function	*NT-Xent*
τ in *NT-Xent*	0.07
Epochs	200
Projection Head Output	128
Fine-tuning Hyperparameters	
Optimizer	Adam
Learning Rate	0.00001
Loss Function	Cross Entropy Loss

IV. Experiments

This section describes the conducted experiments to evaluate the effectiveness of the proposed approach. Two distinct datasets were used for this purpose: the ASCAD dataset and the AES-SAKURA dataset. We conduct the experiments using the PyTorch library in Python on a workstation equipped with an Nvidia GTX 3090 GPU and 128GB of RAM. The hyperparameters used in all experiments are presented in Table I. Our approach is capable of improving the performance of all side-channel analysis methods based on the neural network, and is applicable to various network models. We only showcase the network presented in Table I for the sake of brevity.

A. Experiments on ASCAD Dataset

The ANSSI SCA Database (ASCAD) [27] is a benchmark proposed by Benadjila *et al.* in 2018 for evaluating the performance of side-channel analysis method based on neural networks. This dataset captures electromagnetic traces from the protected AES algorithm executed on the 8-bit AVR architecture ATMega8515 development board. Each trace consists of 700 data points and is associated with a fixed key.

Considering the remote attack scenarios with limited profiled traces, we use a total of 20,000 traces of the ASCAD dataset to apply CL-SCA. Of these, 10,000 traces were labeled with the output of the Sbox and used as the profiled set, while the remaining 10,000 traces were unlabeled and used to form the attack trace set.

In the pretraining part, we utilize all 10,000 profiled traces in D_p without labels to train the encoder. The stochastic data augmentation technique is employed to augment the traces. We fix the hyperparameters of refered data augmentations at $W = 1$, $L_s = 5$ and $L_c = 5$. Due to the fact that most point of the trace is related to the label, and the length of the trace is very short, we keep parameter settings of data augmentation low to avoid losing useful information.

Furthermore, a batch size of 500 is used during pretraining. We employ cosine annealing for the learning rate, which gradually decrease the learning rate over time in a cyclical fashion. Linear warmup is also used for the first 10 epochs to gradually increase the learning rate to its maximum value.

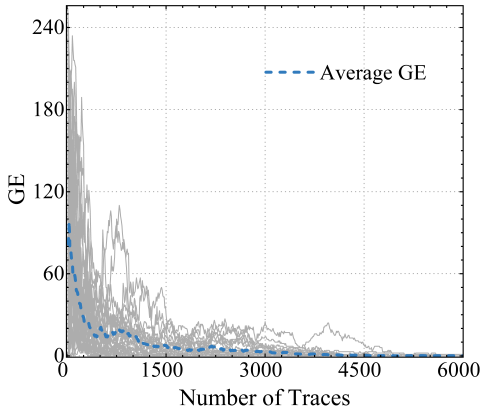

Fig. 6. GE of CL-SCA on ASCAD.

In the fine-tuning part, we concatenate a fully connected layer for classification with the trained encoder. The classification output dimension is set to 256, corresponding to the 256 possible outputs of one byte of Sbox. Notably, we don't freeze the trainable parameters of the encoder. In other words, the encoder and the added fully connected layer is trained simultaneously in the fine-tuning part. We train the neural network with the profiled set and evaluate it with the attack trace set.

We employ the data augmentation techniques in the pretraining part, and assess the performance of the model with the attack trace set during the fine-tuning part. The evaluation metric employed in our study is guessing entropy (GE). GE is a widely used evaluation criterion in SCA field for evaluating the effectiveness of different analysis methods. It quantifies the ranking of the correct key among all candidate keys. Therefore, a lower GE indicates a more accurate key recovery. The results are depicted in Fig. 6. Throughout the experiments, we maintain a fixed number of epochs at 40 to achieve optimal performance. To obtain robust results, we shuffle and calculate the GE curve of the attack set 100 times. Fig. 6 displays these 100 GE curves as gray lines, while the average curve of these 100 GE curves is depicted as a blue dashed line. CL-SCA achieved a GE equal 0 and successfully recovered the correct key using 3,860 traces.

The selection of data augmentation impacts the effectiveness of CL-SCA. Hence, we evaluate the effect of different data augmentation combination on the performance of our approach. We use the CNN as a baseline, which has same model architecture as CL-SCA. We conduct 10 experiments for both CL-SCA and the CNN and calculated the average performance across these 10 experiments. Fig. 7 and Table II shows a comparison of the performance of CL-SCA with different data augmentation techniques on the ASCAD dataset. "None" represents the experimental result of CNN in Fig. 7.

When using only one type of data augmentation, CL-SCA performs best with the RC. It requires about 3627 traces to recover the secret key. CL-SCA with RD follows as the second

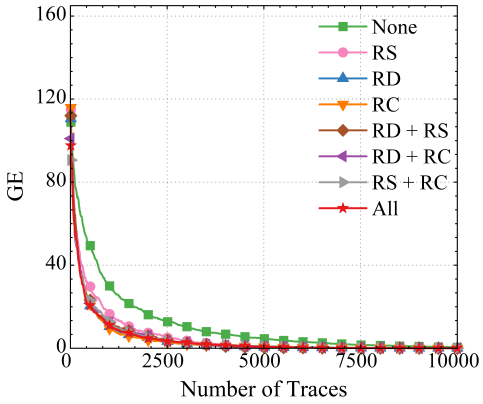

Fig. 7. The results of different data augmentation combinations on ASCAD.

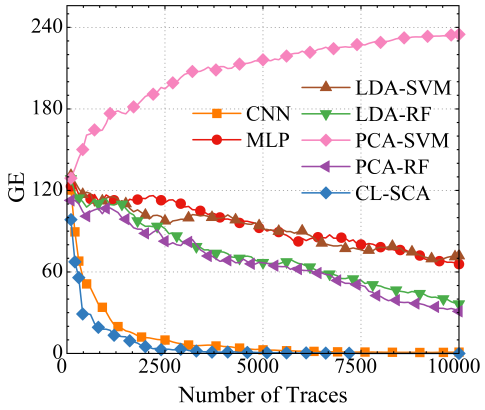

Fig. 8. Comparison of results obtained using different methods on ASCAD.

TABLE II
THE PERFORMANCE OF STOCHASTIC DATA AUGMENTATION COMBINATIONS

Number of Types	Types	Required Traces
One type	**RC**	**3627**
	RD	3923
	RS	4579
Two types	**RD + RS**	**3733**
	RD + RC	4126
	RS + RC	4607
Three types	**RD + RS + RC**	**3888**

most effective technique, while the improvements achieved through RS are more limited. But overall, all of them are better than CNN. This is because not all 700 points in a trace are relevant to the key, and random cropping can help remove some irrelevant points. CL-SCA with RS enables the model to avoid focusing solely on specific fixed samples within the traces, and instead consider the relationships between different samples. CL-SCA with RD improves the signal-to-noise ratio (SNR), resulting in better performance.

When using two types of data augmentation, the best results are achieved when RD and RS are combined. It uses about 3733 traces when the secret key is recovered. However, compared to using only RC, introducing RS or RD leads to a slight decrease in the effectiveness of the experiment. Nevertheless, the extent of decrease is relatively limited. The performance still outperforms CNN.

When using a combination of all three data augmentation techniques, approximately 3888 traces are required when the secret key is recovered successfully. CL-SCA clearly outperforms the CNN model by a significant margin, although this experiment not achieving the top performance among the conducted experiments.

All experiments about stochastic data augmentation emphasize the importance of carefully selecting and utilizing suitable data augmentation tailored to the specific characteristics of the dataset. Notably, incorporating RS as a data augmentation technique proves particularly beneficial for the ASCAD dataset. However, it is important to note that simply increasing the number of data augmentation types does not guarantee improved performance. Overall, our findings underscore the significance of thoughtful augmentation strategy selection for achieving better performance.

After confirming the data augmentation used for the ASCAD dataset, we choose RC as the best data augmentation, and compare our results with various machine learning methods, including CNN, MLP, SVM, and RF. The MLP uses a network structure with three hidden layers. As SVM and RF are more suitable for low-dimensional data, we first use LDA and PCA to reduce the dimensionality of the traces before conducting experiments. Similarly, the network structure of CNN is the same as CL-SCA. The primary difference between CNN and CL-SCA lies in the training process. The side-channel analysis using CNN starts from a randomly initialized model, whereas CL-SCA requires unsupervised pretraining. In addition, to ensure a fair evaluation, all experiments based on neural networks use the same optimizer, learning rate, and other relevant parameters.

The experimental results are shown in Fig. 8, which displays the GE curves obtained from different methods for key recovery. CL-SCA achieves the best performance and can recover the key using about 3,610 attack traces. CNN also performs well, but still needs 7,200 attack traces to recover the key. The other methods, MLP, RF, and SVM cannot recover the key within 10,000 attack traces. Compared to the CNN, CL-SCA reduces the number of required attack traces by 3,590. These results mean that CL-SCA is superior to other machine learning methods, and utilizing our proposed method can effectively reduce the number of traces required for key recovery using CNN by 49.86%.

B. Experiments on AES-SAKURA Dataset

To demonstrate the effectiveness of CL-SCA, we also conduct experiments on the AES-SAKURA dataset. The AES-SAKURA dataset are electromagnetic traces of the AES-128 algorithm implemented on the SAKURA-G platform. The dataset leaks the Hamming distance between the ciphertext and the input of the last round Sbox. We collect a total of 44,200 traces with a length of 500 points (only containing the last round of AES) for the approach, including 35,000 labeled traces and another 9,200 unlabeled traces.

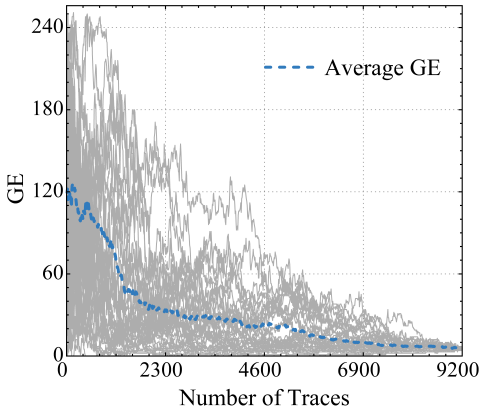

Fig. 9. GE of CL-SCA on AES SAKURA.

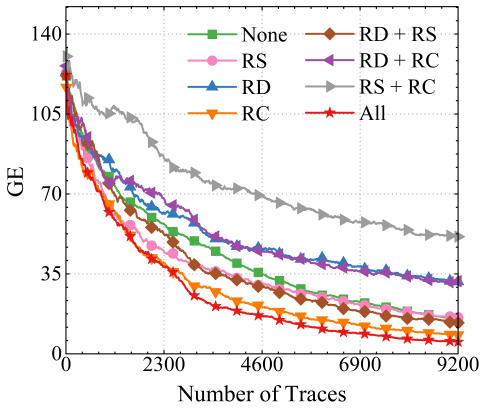

Fig. 10. The results of different data augmentation combinations on AES-SAKURA.

We first conduct CL-SCA on AES-SAKURA using all data augmentation techniques in the pretraining part. All the hyperparameters settings of the experiment are identical to ASCAD, except for the epoch. Based on experiments, we have discovered that setting the epoch to 30 is a more suitable choice to mitigate overfitting.

We present the GE results of the fine-tuning part of CL-SCA in Fig. 9. Due to the hardware parallel simultaneous operation of 16 Sboxes in the same round, the SNR of the traces is very low. Therefore, analyzing this dataset is more challenging compared to ASCAD. Nonetheless, CL-SCA also achieves excellent results. Attacking with 9,200 traces can reduce the GE to 6, and the key can be recovered through key enumeration.

We also evaluate the effectiveness of different data augmentations on this dataset, and present the results in the Fig. 10. It can be observed that for a single type of data augmentation, CL-SCA with RD cannot improve the performance of CL-SCA compared to the CNN experiment. On the contrary, CL-SCA with RC greatly improves the performance, because the SNR is higher at the peak position in the trace of hardware implementation, while the SNR gradually decreases at other positions. CL-SCA with RC can remove low-SNR points with high probability, thereby improving the model's performance.

When using two types of data augmentation, CL-SCA combining RS with RD exhibits the most favorable results. However, it is observed that the introduction of RD has minimal impact on the performance improvement achieved through RS alone. Conversely, when RC is combined with other data augmentation methods, the experimental results are notably poorer, indicating that increasing the number of data augmentation types does not necessarily improve the performance of CL-SCA. When three types of data augmentation are used, the results of multiple experiments are further improved. Its performance is significantly better than CNN and the other data augmentation experiments.

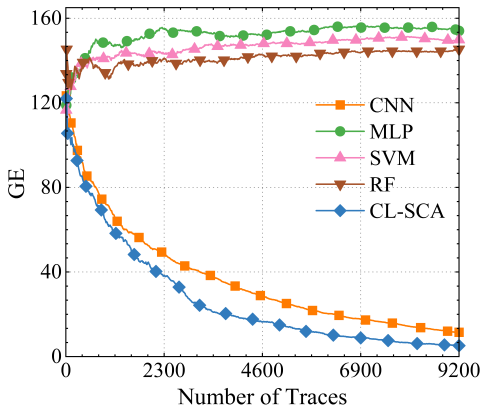

Fig. 11. Comparison of results obtained using different methods on AES-SAKURA.

These experiments once again prove that the selection of data augmentation techniques based on the specific characteristics of the trace dataset can lead to significant improvements. Additionally, increasing the number of data augmentation techniques does not necessarily result in improved performance. Whereas, we would like to emphasize that the three data augmentation combinations used in our experiments produce excellent results on both ASCAD and AES-SAKURA datasets, surpassing the performance of CNN. Therefore, we recommend utilizing all three types of data augmentation when applying CL-SCA to achieve superior generalization.

After selecting combination of all data augmentation types as the most effective data augmentation techniques for the AES-SAKURA dataset, we compare the performance of CL-SCA with several other methods, including CNN, MLP, SVM, and RF. The parameters of side-channel analysis based on neural network are same as ASCAD. The GE results of these experiment are shown in Fig. 11. It can be seen that only the GE of CNN and CL-SCA exhibits a consistent downward trend, with CL-SCA exhibiting the best performance. In contrast, the GE of MLP, RF, and SVM does not demonstrate a decreasing pattern and fails to reach a satisfactory level even with 9,200 traces for attack, remaining outside the top 50. These results highlight the significant advantages of CL-SCA across diverse platforms and implementations. It is evident that CL-SCA outperforms these alternative methods in terms of reducing the GE, further emphasizing its superiority.

TABLE III
EXPERIMENT SETUP

	CNN	CL-SCA	CL-SCA+
Pretrain with stochastic data augmentation	×	✓	✓
Pretrain set	×	Profiled Traces	Profiled Traces and Attack Traces
Fine-tuning/ Regular training	✓	✓	✓
Train set	Profiled Traces	Profiled Traces	Profiled Traces

C. Experiments for CL-SCA+

Although the previous subsection demonstrates the advantages of CL-SCA, this approach still has limitations as it does not utilize information from the attack trace set, despite the presence of useful information in the attack trace set. To address this issue, we utilize the useful information in the attack trace set during the pretraining part of CL-SCA and propose an enhanced approach called CL-SCA+. This approach enables the model to learn some attack trace set's information in an unsupervised manner. In order to evaluate the effectiveness of the enhanced approach, we will conduct experimental verification on different datasets.

We compare the performance of three different methods: CNN, CL-SCA and CL-SCA+. The experimental setup is shown in Table III. To ensure fairness, we use identical hyperparameters, including network architecture and optimizer, as listed in Table I. We set the epoch for the fine-tuning part and the CNN method to 40, in order to prevent overfitting and achieve optimal key recovery. In addition, as recommended before, we use the combination of all data augmentation techniques instead of selecting the data augmentation according to the characteristic of the datasets.

We conduct the experiments on ASCAD dataset, and the results are shown in Fig. 12. Shadows of different colors represent the GE range of different methods, while different dark curves represent the average GE of these methods. It can be observed that the CL-SCA has significant advantages over the CNN. Several CNN experiments show that 10,000 attack traces are not sufficient to reduce GE to 0, whereas all experiments of the CL-SCA can reduce GE to 0. This suggests that simple training is not enough, and using contrastive learning and stochastic data augmentation for pretraining can help the model focus on previously overlooked information, significantly improving its effectiveness.

The results of CL-SCA and CL-SCA+ exhibit a close proximity, with an insignificant difference between them as shown in Fig. 12. Therefore, we analyze the results from all experiments of these two approaches in more detail. We calculate the distribution of the number of required attack traces separately, as shown in Fig. 13. The horizontal axis represents the range of the number of traces required to reduce GE to 0, while the vertical axis represents the number of experiments. In the case of successful secret key recovery in CL-SCA+, we have observed that the maximum number of experiments occurs when the number of attack traces falls within the range of (3000, 4000]. In contrast, when it comes to CL-SCA, the

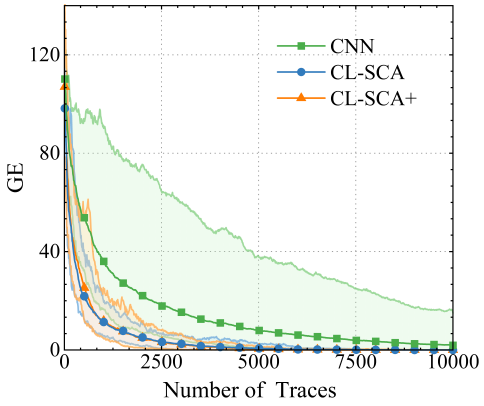

Fig. 12. The results of CL-SCA and CL-SCA+ on ASCAD.

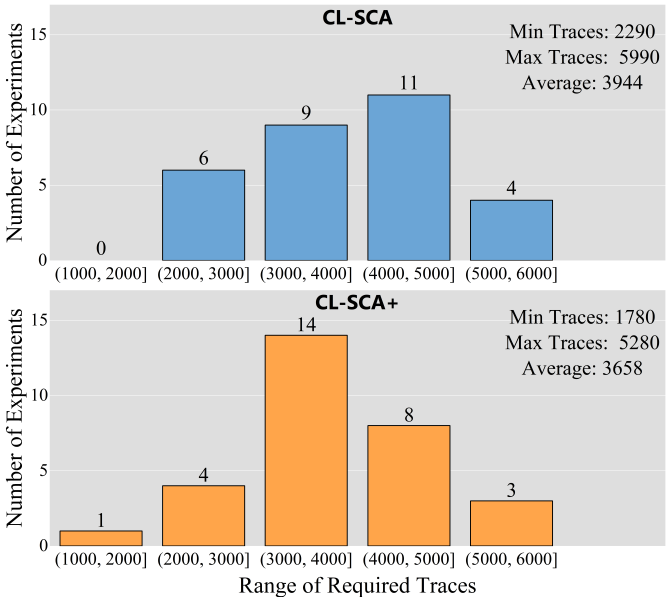

Fig. 13. Distribution of multiple experimental results of CL-SCA and CL-SCA+ on ASCAD.

range of (4000, 5000] attack traces witnessed the highest number of experiments. When processing the same amount of traces, it is more likely for CL-SCA + to successfully recover the key compared to CL-SCA. We also calculate the average number of traces required for all experiments and find that CL-SCA requires 3944 attack traces when the secret key is successfully recovered, while CL-SCA+ requires 3658 traces. Furthermore, CL-SCA+ demonstrates superior performance in both the best and worst experimental results. This indicates that utilizing the attack trace set in the pretraining part can help the model learns more useful information and improves key recovery performance.

We also conduct the experiments on AES-SAKURA dataset using the configurations shown in Table III. The results are presented in Fig. 14. It is evident that both CL-SCA methods are significantly superior to CNN. Unlike the results on the ASCAD dataset, the difference between CL-SCA+ and CL-SCA is more pronounced in AES-SAKURA dataset. The results of improved CL-SCA+ are more stable, and its GE

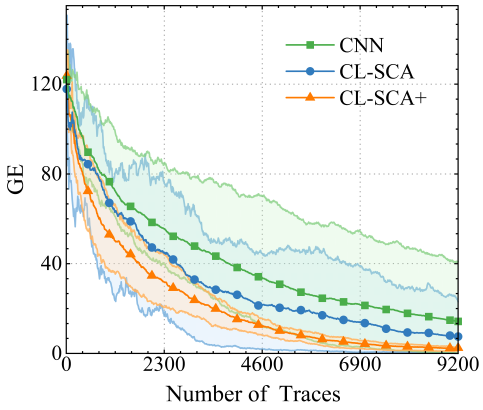

Fig. 14. The results of CL-SCA and CL-SCA+ on AES-SAKURA.

is lower. These experimental results once again highlight the benefits that contrastive learning and stochastic data augmentation bring to side-channel analysis, and the utilization of attack trace sets can also improve the effectiveness of key recovery.

V. Conclusion

In this paper, we apply contrastive learning to the community of side-channel analysis for the first time and propose CL-SCA, which can extract useful information in unsupervised setting. We also utilize stochastic data augmentation technology to assist the model to ignore irrelevant information and focus on critical features in the traces. We prove that when using appropriate data augmentation techniques, which are tailored to the characteristics of the dataset, CL-SCA can perform even better. We also demonstrate that the performance of CL-SCA outperforms traditional machine learning methods through different experiments .

By introducing the attack trace set into the training process, we propose an extension approach CL-SCA+. The improved approach can unsupervisedly extract the useful information in the attack trace set, making key recovery easier. We have conducted a detailed analysis of the experimental results of CL-SCA and CL-SCA+ on different datasets to verify this point.

In the future, we intend to apply the concept of contrastive learning to various research findings in SCA methods based on neural network and investigate its ability for enhancing these methods. In addition, we recognize the need for further exploration into the impact of different components within CL-SCA, such as the projection head and network structure, on experimental results.

Acknowledgments

This work was supported in part by National Key R&D Plan of China (2022YFB3103800), the National Natural Science Foundation of China (61872040, 62002021, 62302036), Beijing Natural Science Foundation (4202070), Henan Key Laboratory of Network Cryptography Technology (LNCT2022-A24).

References

[1] K. L. Lueth, "State of the iot 2020," https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time, 2020, accessed July 19, 2023.

[2] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, ser. Lecture Notes in Computer Science, N. Koblitz, Ed., vol. 1109. Springer, 1996, pp. 104–113. [Online]. Available: https://doi.org/10.1007/3-540-68697-5_9

[3] S. Zhdanov and E. Maro, "Power analysis side-channel attacks on symmetric block cipher magma." in *13th International Conference on Security of Information and Networks*, ser. SIN 2020. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3433174.3433601

[4] L. Batina, S. Bhasin, D. Jap, and S. Picek, "CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 515–532. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/batina

[5] J. Jancar, M. Fourné, D. D. A. Braga, M. Sabt, P. Schwabe, G. Barthe, P.-A. Fouque, and Y. Acar, ""they're not that hard to mitigate": What cryptographic library developers think about timing attacks," in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 632–649.

[6] L. Lerman and O. Markowitch, "Efficient profiled attacks on masking schemes," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1445–1454, 2019.

[7] S. Paguada, L. Batina, I. Buhan, and I. Armendariz, "Playing with blocks: Toward re-usable deep learning models for side-channel profiled attacks," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2835–2847, 2022.

[8] M. O. Choudary and M. G. Kuhn, "Efficient, portable template attacks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 2, pp. 490–501, 2017.

[9] G. Hospodar, E. Mulder, B. Gierlichs, I. Verbauwhede, and J. Vandewalle, "Least squares support vector machines for side-channel analysis," *Center for Advanced Security Research Darmstadt*, pp. 99–104, 2011.

[10] S. Picek, A. Heuser, A. Jovic, and L. Batina, "A systematic evaluation of profiling through focused feature selection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 12, pp. 2802–2815, 2019.

[11] S. Ghandali, S. Ghandali, and S. Tehranipoor, "Deep k-tsvm: A novel profiled power side-channel attack on aes-128," *IEEE Access*, vol. 9, pp. 136 448–136 458, 2021.

[12] T. Bartkewitz, "Leakage prototype learning for profiled differential side-channel cryptanalysis," *IEEE Trans. Computers*, vol. 65, no. 6, pp. 1761–1774, 2016. [Online]. Available: https://doi.org/10.1109/TC.2015.2455974

[13] H. D. Tsague and B. Twala, "Reverse engineering smart card malware using side channel analysis with machine learning techniques," in *2016 IEEE International Conference on Big Data (IEEE BigData 2016), Washington DC, USA, December 5-8, 2016*, J. Joshi, G. Karypis, L. Liu, X. Hu, R. Ak, Y. Xia, W. Xu, A. Sato, S. Rachuri, L. H. Ungar, P. S. Yu, R. Govindaraju, and T. Suzumura, Eds. IEEE Computer Society, 2016, pp. 3713–3721. [Online]. Available: https://doi.org/10.1109/BigData.2016.7841039

[14] A. Levina, D. Sleptsova, and O. Zaitsev, "Side-channel attacks and machine learning approach," in *2016 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT)*, 2016, pp. 181–186.

[15] S. Picek, A. Heuser, A. Jovic, and A. Legay, "Climbing down the hierarchy: Hierarchical classification for machine learning side-channel attacks," in *Progress in Cryptology - AFRICACRYPT 2017 - 9th International Conference on Cryptology in Africa, Dakar, Senegal, May 24-26, 2017, Proceedings*, ser. Lecture Notes in Computer Science, M. Joye and A. Nitaj, Eds., vol. 10239, 2017, pp. 61–78. [Online]. Available: https://doi.org/10.1007/978-3-319-57339-7_4

[16] A. Heuser, S. Picek, S. Guilley, and N. Mentens, "Lightweight ciphers and their side-channel resilience," *IEEE Trans. Computers*, vol. 69, no. 10, pp. 1434–1448, 2020. [Online]. Available: https://doi.org/10.1109/TC.2017.2757921

[17] L. Lerman, S. F. Medeiros, G. Bontempi, and O. Markowitch, "A machine learning approach against a masked AES," in *Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised*

Selected Papers, ser. Lecture Notes in Computer Science, A. Francillon and P. Rohatgi, Eds., vol. 8419. Springer, 2013, pp. 61–75. [Online]. Available: https://doi.org/10.1007/978-3-319-08302-5_5

[18] S. Picek, A. Heuser, and S. Guilley, "Template attack versus bayes classifier," *J. Cryptogr. Eng.*, vol. 7, no. 4, pp. 343–351, 2017. [Online]. Available: https://doi.org/10.1007/s13389-017-0172-7

[19] L. Lerman, G. Bontempi, and O. Markowitch, "Side channel attack: an approach based on machine learning," *Center for Advanced Security Research Darmstadt*, vol. 29, 2011.

[20] Y. Tanaka, R. Ueno, K. Xagawa, A. Ito, J. Takahashi, and N. Homma, "Multiple-valued plaintext-checking side-channel attacks on post-quantum kems," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 473–503, 2023.

[21] E. Cagli, C. Dumas, and E. Prouff, "Enhancing dimensionality reduction methods for side-channel attacks," in *Smart Card Research and Advanced Applications: 14th International Conference, CARDIS 2015, Bochum, Germany, November 4-6, 2015. Revised Selected Papers 14*. Springer, 2016, pp. 15–33.

[22] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*, ser. Lecture Notes in Computer Science, C. Carlet, M. A. Hasan, and V. Saraswat, Eds., vol. 10076. Springer, 2016, pp. 3–26. [Online]. Available: https://doi.org/10.1007/978-3-319-49445-6_1

[23] J. Chen, J.-S. Ng, K.-S. Chong, Z. Lin, and B.-H. Gwee, "A novel normalized variance-based differential power analysis against masking countermeasures," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3767–3779, 2021.

[24] C. Luo, Y. Fei, and D. Kaeli, "Effective simple-power analysis attacks of elliptic curve cryptography on embedded systems," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–7.

[25] L. Weissbart, "On the performance of multilayer perceptron in profiling side-channel analysis," *Cryptology ePrint Archive*, 2019.

[26] E. Cagli, C. Dumas, and E. Prouff, "Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing," in *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, ser. Lecture Notes in Computer Science, W. Fischer and N. Homma, Eds., vol. 10529. Springer, 2017, pp. 45–68. [Online]. Available: https://doi.org/10.1007/978-3-319-66787-4_3

[27] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, and C. Dumas, "Deep learning for side-channel analysis and introduction to ASCAD database," *J. Cryptogr. Eng.*, vol. 10, no. 2, pp. 163–188, 2020. [Online]. Available: https://doi.org/10.1007/s13389-019-00220-8

[28] G. Yang, H. Li, J. Ming, and Y. Zhou, "Convolutional neural network based side-channel attacks in time-frequency representations," in *Smart Card Research and Advanced Applications, 17th International Conference, CARDIS 2018, Montpellier, France, November 12-14, 2018, Revised Selected Papers*, ser. Lecture Notes in Computer Science, B. Bilgin and J. Fischer, Eds., vol. 11389. Springer, 2018, pp. 1–17. [Online]. Available: https://doi.org/10.1007/978-3-030-15462-2_1

[29] J. Kim, S. Picek, A. Heuser, S. Bhasin, and A. Hanjalic, "Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2019, no. 3, pp. 148–179, 2019. [Online]. Available: https://doi.org/10.13154/tches.v2019.i3.148-179

[30] K. Ramezanpour, P. Ampadu, and W. Diehl, "SCAUL: power side-channel analysis with unsupervised learning," *IEEE Trans. Computers*, vol. 69, no. 11, pp. 1626–1638, 2020. [Online]. Available: https://doi.org/10.1109/TC.2020.3013196

[31] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[32] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 22 243–22 255. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/fcbc95ccdd551da181207c0c1400c655-Paper.pdf

[33] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

[34] X. Chen, H. Fan, R. B. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *CoRR*, vol. abs/2003.04297, 2020. [Online]. Available: https://arxiv.org/abs/2003.04297