Efficient Side-Channel Attack Through Balanced Labels Compression and Variational Autoencoder

Nengfu Cai, Zhiqin Yang, Shuhai Wang, Yanling Jiang,
Mingsheng Liu and Gang Li

February 20, 2024

Efficient Side-Channel Attack through Balanced Labels Compression and Variational Autoencoder

Nengfu Cai[1], Zhiqin Yang[1], Shuhai Wang[2*], Yanling Jiang[1], Mingsheng Liu[3], Gang Li[4]
[1]School of Cyber Science and Technology, Beihang University, Beijing 100191, China;
[2]School of Information Science and Technology, Shijiazhuang Railway University, Shijiazhuang 050043, China;
[3]Shijiazhuang Institute of Railway Technology, Shijiazhuang 050041, China;
[4]Zhongke Zidong Information Technology (Beijing) Co. Ltd, Beijing, China;
cainengfu@buaa.edu.cn, yangzqccc@buaa.edu.cn, wsh36302@126.com, yanlingj@buaa.edu.cn, liums601001@sina.com, ligang@people-ai.cn

Abstract—**Recently, side-channel attacks based on deep learning (DLSCAs) have attracted much attention. Many works have improved the performance of DLSCAs by designing advanced neural network architectures and training strategies. There are few studies on leakage models for DLSCAs. Existing researches usually utilize the intermediate value Hamming weight (HW) and the intermediate value itself (ID) as leakage models. Training a classifier with good performance is challenging due to the many label classes in the ID leakage model. The HW leakage model can significantly reduce the number of labels, but it will cause samples imbalance. In this paper, we propose a new DLSCA leakage model, named Balanced Labels Compression (BLC). We consider dividing sensitive intermediate values with same lowest ϵ bits into same class to obtain balanced labels. Then, we train a classifier using the compressed BLC labels and profiling energy traces. At the attack phase, the probability distribution of BLC labels is extended to the probability distribution of sensitive intermediate values. We conduct extensive comparison experiments with HW, ID, and BLC leakage models under the two scenarios of sufficient and insufficient profiling energy traces. Further, we exploit VAE to improve attack performance when energy traces are insufficient. Experimental results show that VAE-based data augmentation can significantly reduce the energy traces required to recover key.**

Index Terms—**Side-channel attacks, Side-channel leakage model, Variational autoencoder, Data augmentation.**

I. INTRODUCTION

Side-channel attacks (SCAs) utilize unintentionally leaked information in the physical implementation of encryption algorithms to recover secret information [1]. Depending on the attacker's power, SCAs include profiled attacks and non-profiled attacks. Among them, profiled attacks are one of the most outstanding SCAs. The profiled attacks assume the attacker has similar test equipment to the target. The attacker builds an estimation model of sensitive variables in advance based on the device and obtains secret information from the target device [2], [3].

In recent years, deep learning-based profiled SCAs have performed better than traditional template attacks in some aspects [4]–[6]. In 2016, Maghrebi *et al.* [7] first used deep learning methods such as deep convolutional neural networks (CNN) and multilayer perceptron (MLP) to SCAs. The CNN algorithm has gained significant advantages in cracking unprotected and protected AES implementations compared to template attacks and traditional machine learning methods. Works [8], [9] built efficient and advanced CNN architectures to improve the attack effect and reduce the network complexity. Kim *et al.* [5] added artificial noise to the input signal to improve the performance of CNN. This way, the energy traces required to recover the key is significantly reduced. Works [10], [11] used Bayesian optimization and deep reinforcement learning technologies for hyperparameter tuning of deep neural networks to improve the attack ability of SCAs. Zaid *et al.* [12] proposed a new loss function for DLSCA, named ranking loss, which achieved more minor estimation errors than the cross-entropy loss function. These studies have achieved satisfactory performance of DLSCAs by designing advanced neural network architectures, training strategies, and loss functions.

However, existing DLSCAs still have some problems worth exploring. Existing DLSCAs usually utilize the intermediate value Hamming weight (HW) and the intermediate value (ID) as the leakage models. For the AES-128 symmetric encryption algorithm, recovering a one-byte key requires training a 256-class classifier. Training such a classifier requires many energy traces and is difficult to converge. The HW leakage model can significantly reduce the number of classifier labels. But, since the number of 1s in the binary number obeys the binomial distribution, the HW leakage model will cause samples imbalance.

In this paper, we aim to design a new DLSCA leakage model that can effectively compress the number of labels without sample imbalance. Luo *et al.* [13] used the last bit of the intermediate value binary representation (LSB) as the label for SCA and proved through experiments that its performance is better than that of the ID leakage model and the HW leakage model. Inspired by this work, we deeply analyze the relationship between intermediate values label selection and key recovery performance. We propose a

[*] corresponding author

balanced labels compression leakage model (BLC) that can compress the number of intermediate value labels evenly. We assume there is some similarity in side channel energy traces when a fraction of the bits in the sensitive intermediate values are the same. Specifically, the BLC leakage model consists of three stages. In labels compression stage, intermediate values are mapped to BLC labels by modulo 2^{ε} operations. ε is an optional parameter. In probability distribution expansion stage, we expand the probability distribution dimension of the classifier output from 2^{ε} to 256. Then, the correct key is recovered by aggregating the probability distributions generated from multiple energy traces.

We compare the attack performance of the BLC with HW, ID, and LSB leakage models in detail under the two scenarios of sufficient and insufficient profiling energy traces on the public ASCAD dataset. Experimental results show that BLC significantly outperforms these three commonly used leakage models.

In some attack scenarios, the attacker may only have a small amount of energy traces to build a model. Further, we utilize variational autoencoders (VAE) for data augmentation to improve attack performance when profiling energy traces are insufficient.

In summary, our contribution includes below.

- We propose BLC, a new DLSCA leakage model that can effectively compress intermediate value labels without sample imbalance.
- We compare the attack performance of the BLC with the commonly used DLSCAs leakage models in detail under the two scenarios of sufficient and insufficient profiling energy traces. Experimental results show that BLC significantly outperforms these leakage models
- We utilize VAE to generate side-channel energy traces to improve attack performance in scenarios with insufficient energy traces. The experimental results verify that the VAE-based data augmentation DLSCA method is very effective.

The paper is organized as follows. Section II briefly introduces deep learning-based side-channel attacks and variational autoencoder. Section III introduces our new SCA leakage model and VAE-based data-augmented SCA. In section IV and section V, we verify the performance of our model and compare with representative methods. Finally, we introduce related work and conclude the paper.

II. PRELIMINARIES

In this section, we first introduce the notations used and then introduce deep learning-based side-channel attacks and variational autoencoders.

A. Notation

In this paper, we use the time series $\mathbf{x} = [t_1, t_2, \ldots, t_\tau]$ to represent a side-channel signal. The target sensitive intermediate value is $Z = H(P, K)$, where H represents the cryptographic primitive, P represents the public variable (such as plaintext or ciphertext), and K represents the key the attacker is trying to obtain. Z takes values in $\mathcal{Z}$.

B. Deep Learning-Based Side-Channel Attacks

Deep learning-based SCA is a type of profiled SCAs. Profiled SCAs include two stages: the profiled phase and the attack phase. In the first stage, the attacker uses the profiled dataset $D_{profil} = \left\{(\mathbf{x}_0, z_0), \ldots, \left(\mathbf{x}_{N_p-1}, z_{N_p-1}\right)\right\}$ to construct a deep learning model $M : \mathbb{R}^{\tau} \rightarrow \mathbb{R}^{|\mathcal{Z}|}$ that estimates the probability $\Pr[\mathbf{X}|Z = z]$. After training on the D_{profil}, the attacker obtains an excellent sensitive intermediate value probability estimator.

In the attack phase, the attacker utilizes the trained model M, the attack dataset D_{attack} and the input used during the encryption to calculate the score vector for each key hypothesis. $F(\mathbf{x}_i)$ is the sensitive intermediate value probability estimate calculated by the model M, $i \in \{0, 1, \ldots, N_a - 1\}$. For each $k \in K$, this score is defined as:

$$\mathcal{L}_{N_a}(k) = \sum_{i=1}^{N_a} \log\left(M\left(\mathbf{x}_i\right)\left[z_i\right]\right), \tag{1}$$

where $z_i = H(p_i, k)$ and H denotes a cryptographic primitive. Finally, the key guess is calculated by:

$$\mathcal{K}_{\text{guess}} = \arg\max_{k} \mathcal{L}_{N_a}(k), \tag{2}$$

where $k \in K$.

C. Variational Autoencoder

The variational autoencoder (VAE) was proposed by Kingma and Welling [14], which is a particular variant of AE and consists of an encoder and a decoder. VAE is a widely used generative model, like Generative Adversarial Networks. The encoder learns the distribution of input variables in the latent space. The decoder samples latent variables from the output distribution to reconstruct input variables. Therefore VAE can be used to generate samples similar to the input variables conveniently. In this study, we exploit VAE to generate side-channel energy traces to augment profiled dataset in sample-poor situations.

The VAE framework is designed based on a variational Bayesian method. First, an encoder $q_{\Phi}(h|x)$ is used to approximate the actual posterior distribution $p_{\theta}(h|x)$. Assume that the latent variable h follows a Gaussian distribution. Specifically, the output of the encoder is the mean vector $\mu_{\Phi}(x)$ and variance vector $\sigma^2_{\Phi}(x)$ of the latent vector h. Then we sample from this Gaussian distribution to obtain the latent vector h. The decoder $p_{\theta}(x|h)$ generates an input vector x from a latent vector h. The parameters Φ and θ in the encoder and decoder are trained through deep neural networks.

III. METHODOLOGIES

In this section, we first introduce the proposed BLC leakage model in detail. Then, we describe a data augmentation method for SCA based on the variational autoencoder.

A. BLC leakage model

Existing DLSCAs usually utilize the intermediate value Hamming weight (HW) and the intermediate value (ID) as the leakage models. When using the ID model as the classifier label, the number of label categories is large. Training such a classifier requires many energy traces and is difficult to converge. The Hamming weight model can significantly reduce the number of label categories but will lead to sample imbalance. In this paper, we aim to design a new DLSCA leakage model that can significantly reduce the number of label categories and avoid the sample imbalance problem, that is, balanced labels compression.

We investigate the relationship between sensitive intermediate label selection and SCA. The AES algorithm performs various calculation operations in the Galois field $GF(2^8)$ to realize encryption and decryption functions. We assume there is some similarity in side channel energy traces when a fraction of the bits in the sensitive intermediate values are the same. Specifically, we consider dividing sensitive intermediate values with the same lowest ε bits into the same class. Sensitive intermediate values will be evenly divided into 2^ε classes. We name this leakage model Balanced Labels Compression (BLC), which consists of the labels compression stage, probability distribution expansion, and key recovery stage.

The labels compression stage. Sensitive intermediate values in the profiled dataset are converted to BLC labels by modulo operation, $D_{profil} \rightarrow D_{profil}^{BLC}$:

$$z^{BLC} = z \bmod (2^\varepsilon), \quad \varepsilon = 1, 2 \cdots 8, \tag{3}$$

where $z = \{z_0, \ldots, z_{N_p-1}\}$. The profiled dataset after labels compression is $D_{profil}^{BLC} = \left\{\left(\mathbf{x}_0, z_0^{BLC}\right), \ldots, \left(\mathbf{x}_{N_p-1}, z_{N_p-1}^{BLC}\right)\right\}$. It will be used to train the classifier. For simplicity, we use the classifier architecture from work [15], with the output layer modified to accommodate the compressed labels. The probability distribution expansion stage. The trained classifier M^{BLC} and attack dataset D_{attack} are used to recover the correct key. The output $M^{BLC}(x_i) \in R^{1\times 2^\varepsilon}$ of each energy trace through the classifier is the probability distribution of 2^ε BLC labels. $M^{BLC}(x_i)$ will be extended to the probability distribution $M(x_i)$ about sensitive intermediate values:

$$M^{BLC}(x_i) \overset{\text{extend}}{\longrightarrow} M(x_i) \in R^{1\times 256}. \tag{4}$$

This way, sensitive intermediate values with the same BLC labels will get the same probability values. The key recovery stage. The classifier uses the N_a energy traces in D_{attack} to generate N_a probability distributions about sensitive intermediate values. These probability distributions are used to compute the score vector for each key hypothesis and key guess $\mathcal{K}_{\text{guess}}$ via Equation (1) and Equation (2).

For each energy trace in D_{attack}, there will be $\frac{256}{2^\varepsilon} - 1$ key hypotheses with the same probability value as k^*. After the classifier is trained on the profile set, this probability value is usually one of the larger ones in the probability distribution. Due to the effect of the cryptographic primitive H, the key hypotheses that obtain the same probability value as k^* are not precisely the same in the probability distributions generated by different energy traces. Therefore, after fusing the probability distributions of multiple energy traces, the score value of $\mathcal{K}_{\text{guess}}$ becomes the maximum value in the score vector for each key hypothesis.

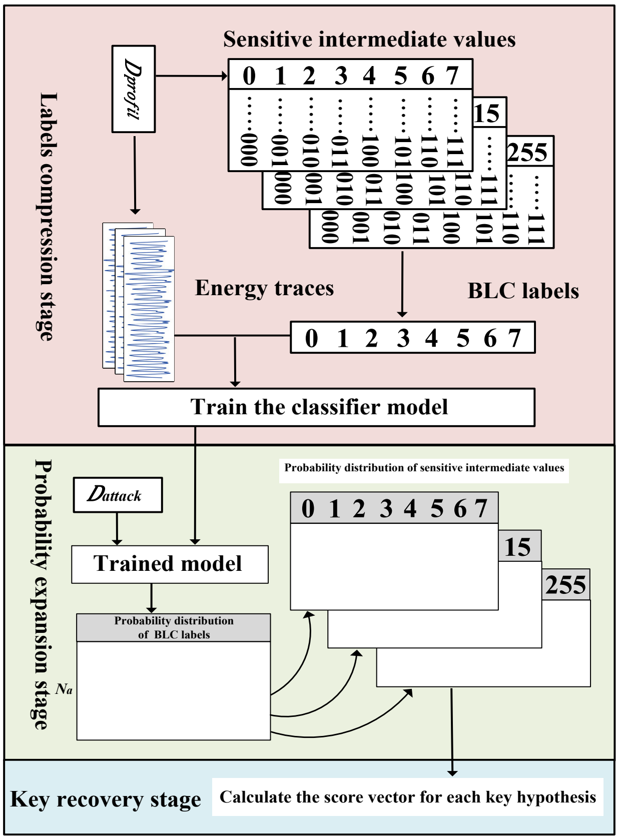


Fig. 1. Method framework of BLC leakage mdoel.

B. Data augmentation SCA based on variational autoencoder with insufficient energy traces

TABLE I
STRUCTURE OF THE ENCODER.

Layer	Output shape	Filter Size	Parameter
Reshape Layer	None*28*25	–	0
Conv1D Layer	None*14*256	256	19456
Conv1D Layer	None*7*128	128	98432
Flatten Layer	None*896	–	0
Dense Layer	None*2	–	1794
Dense Layer	None*2	–	1794

In some attack scenarios, due to the constraints of time, equipment, etc., there are not enough energy traces to train the model, resulting in the degradation of SCA performance. In this paper, we investigate the use of variational

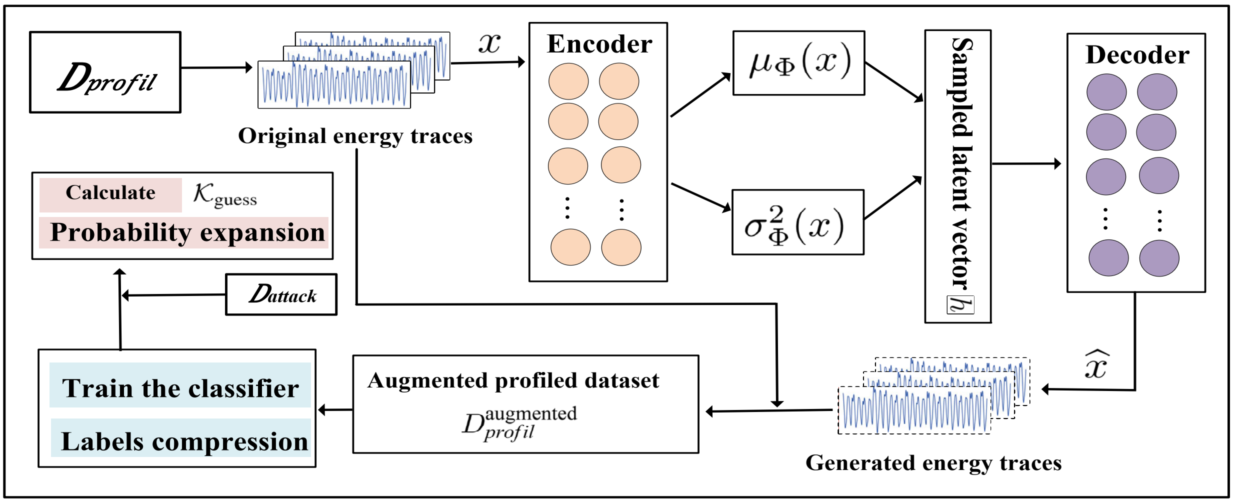


Fig. 2. The workflow of data augmented SCA based on VAE.

TABLE II
STRUCTURE OF THE DECODER.

Layer	Output shape	Filter Size	Parameter
InputLayer Layer	None*2	–	0
Dense Layer	None*896	–	2688
Reshape Layer	None*7*128	–	0
Conv1DTranspose Layer	None*14*256	256	98560
Conv1DTranspose Layer	None*28*25	25	19225
Flatten Layer	None*700	–	0
Dense Layer	None*700	–	490700

autoencoders to data augment original side-channel traces to improve the performance of SCA when the number of samples is limited. As a popular generative model, VAE has the essential advantages of simple training and stable performance.

The workflow of data augmented SCA based on VAE is shown in Figure 2. To verify the effect of data augmentation on SCA when the number of profiled samples is limited, we select a small number of energy traces from the complete dataset as the profiled dataset D_{profil}. Section V will elaborate on the configuration of the number of energy traces. First, the encoder and decoder in the VAE are trained using the energy traces in D_{profil}. Then, decoder generates side-channel samples similar to the original energy trace from latent variable h. The original dataset and the generated energy traces are mixed as a new profiled dataset $D_{profil}^{augmented}$ to train a classifier. The energy traces utilize discrete cosine transform (DCT) to extract features and reduce the signal dimension before inputting into the classifier neural networks. Finally, the key guess value $\mathcal{K}_{\text{guess}}$ is calculated using the trained classifier and the attack dataset D_{attack}.

In this paper, we assume that the latent variable h follows a Gaussian distribution $\mathcal{N}\left(\boldsymbol{h}; \boldsymbol{\mu}_\Phi(x), \boldsymbol{\sigma}^2_\Phi(x)\right)$. The encoder $q_\Phi(h|x)$ is used to approximate the actual posterior distribution $p_\theta(h|x)$. The decoder $p_\theta(x|h)$ generates a reconstructed vector $\widehat{x}$ from a latent vector h. Our encoder and decoder architecture is quite simple. The encoder consists of Reshape Layer, Conv1D Layer, Flatten Layer, and Dense Layer. The decoder includes InputLayer Layer, Dense Layer, Reshape Layer, Flatten Layer, and Conv1DTranspose Layer. The specific parameter settings are shown in Table I and Table II. We use KL divergence to measure the similarity between $q_\Phi(h|x)$ and $p_\theta(h|x)$. The specific expression is as follows:

$$\begin{aligned}
&D_{KL}\left(q_\Phi(h|x)||p_\theta(h|x)\right)\\
&=E_{h\sim q_\Phi(h|x)}\left[\log q_\Phi(h|x)-\log\frac{p_\theta(x,h)}{p_\theta(x)}\right]\\
&=E_{h\sim q_\Phi(h|x)}\left[\log q_\Phi(h|x)-\log p_\theta(x|h)-\log p_\theta(h)\right]\\
&\quad+\log p_\theta(x).
\end{aligned} \tag{5}$$

According to the property of $D_{KL}\left(q_\Phi(h|x)||p_\theta(h|x\right) \geqslant 0$, we can get:

$$\begin{aligned}
&\log p_\theta(x)\\
&\geqslant E_{h\sim q_\Phi(h|x)}\left[-\log q_\Phi(h|x)+\log p_\theta(x|h)+\log p_\theta(h)\right]\\
&=E_{h\sim q_\Phi(h|x)}\left[\log p_\theta(x|h)\right]-D_{KL}\left(q_\Phi(h|x)||p_\theta(h)\right).
\end{aligned} \tag{6}$$

We maximize the lower bound of the log-likelihood $\log p_\theta(x)$ to train encoder and decoder parameters Φ and θ. The loss function of VAE is expressed as follows:

$$\begin{aligned}
&\mathcal{L}(\theta,\Phi,\boldsymbol{x})\\
&=-E_{h\sim q_\Phi(h|x)}\left[\log p_\theta(x|h)\right]+D_{KL}\left(q_\Phi(h|x)||p_\theta(h)\right),
\end{aligned} \tag{7}$$

where $p_\theta(h)=\mathcal{N}(\boldsymbol{z};\boldsymbol{0},\boldsymbol{I})$. $\left\{-E_{h\sim q_\Phi(h|x)}\left[\log p_\theta(x|h)\right]\right\}$ is the reconstruction loss, which ensures that the generated energy trace is sufficiently similar to the original energy trace. $\left\{D_{KL}\left(q_\Phi(h|x)||p_\theta(h)\right)\right\}$ is a regularization loss, which makes $q_\Phi(h|x)$ close to the standard normal distribution and ensures that the model can generate new energy traces.

IV. Performance Evaluation for BLC Leakage Model

A. Experimental Setup

To comprehensively evaluate the BLC leakage model and find a suitable label compression parameter ϵ, we conduct extensive experiments under two scenarios, namely, sufficient profiling energy traces and insufficient profiling energy traces. We train and test model on a computer with an Intel Xeon Gold 5320 @2.2 GHz processor, 32GB of RAM, and an NVIDIA RTX A4000 card with 16GB memory. The batch size is 100. We compare experimental results with commonly used HW, ID, LSB leakage models.

TABLE III
Experimental parameter settings with sufficient energy traces.

Training set	Validation set	Leakage model	Label categories
45000	5000	HW	9
45000	5000	ID	256
45000	5000	LSB	2
45000	5000	BLC4 (ϵ=2)	4
45000	5000	BLC8 (ϵ=3)	8
45000	5000	BLC16 (ϵ=4)	16
45000	5000	BLC32 (ϵ=5)	32

TABLE IV
Experimental parameter settings with insufficient energy traces.

Training set	Validation set	Leakage model	Label categories
45000	5000	HW	9
45000	5000	ID	256
45000	5000	LSB	2
45000	5000	BLC4 (ϵ=2)	4
45000	5000	BLC8 (ϵ=3)	8

B. Dataset

We test the performance of our model on three public datasets. The target platform implemented a masked AES-128 algorithm on an 8-bit AVR microcontroller and captured physical signals [2]. In order to facilitate the use of researchers and simplify data processing, the ASCAD database retains 700 points related to the third byte of the first round of S-box operations of AES-128 as side channel features.

1) ASCAD synchronization (desync0): the signals of the ASCAD synchronization dataset are jitter-free. This dataset contains 50,000 side-channel traces as profiled dataset and 10,000 side-channel traces as attack dataset for key recovery. We use 45,000 of the profiled dataset as the training set and 5,000 as the validation set

TABLE V
The number of energy traces required to recover the key with different leakage models when energy traces are sufficient.

Dataset	HW	ID	LSB	BLC4	BLC8	BLC16	BLC32
desync0	943	87	54	67	**42**	95	107
desync50	1034	178	151	77	**68**	160	120
desync100	1795	368	155	**60**	151	181	165
Overall	3772	633	360	**204**	261	436	392

2) ASCAD desynchronization 50 (desync50): the signals of the ASCAD desynchronization 50 dataset have a maximum jitter (offset) of 50 points. This is to simulate signal jitter caused by poor contact or random delays of sampling equipment.

3) ASCAD desynchronization 100 (desync100): the signals of the ASCAD desynchronization 100 dataset have a maximum jitter (offset) of 100 points.

Table III shows the experimental parameter settings for the sufficient energy traces scenario. Table IV shows the experimental parameter settings for the insufficient energy traces scenario. Experiments are repeated five times under the same experimental settings. The experimental results are average of five times.

C. Performance Metrics

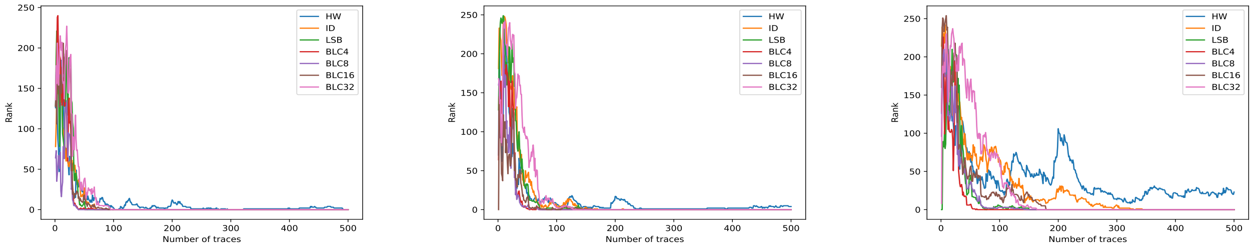
We use the number of traces $\mathcal{N}_{rank}$ required to recover one key byte as performance Metric. As shown in Figure 3, when $Rank$ gradually converges to 0, the number of energy traces is $\mathcal{N}_{rank}$. It is one of the most widely used DLSCA evaluation methods. The smaller the $\mathcal{N}_{rank}$, the better the attack performance.

D. Experimental Results with Sufficient Energy Traces

1) desync0: the detailed experimental results are shown in Table V. The experimental results of ID leakage model are from work [15]. For desync0 dataset, most leakage models can recover a key byte using less than 100 energy traces. The BLC8 (ϵ=3) leakage model outperforms other methods and can recover one key byte using 42 energy traces. There are only eight categories of compressed labels. Compared with the uncompressed 256 categories, the classifier training converges more easily.

2) desync50: for desync50 dataset, most leakage models can recover a key byte using less than 200 energy traces. BLC8 (ϵ=3) and BLC4 (ϵ=2) leakage models are significantly better than others and can recover a key byte using 68 and 77 energy traces, respectively.

3) desync100: as energy trace desynchronization increases, the number of energy traces required to recover one key byte increases more for HW and ID leakage models. The performance of the LSB and BLC leakage models on the three datasets is relatively stable. For desync100 dataset, the BLC4 (ϵ=2) leakage model outperforms other methods and can recover one key byte using 60 energy traces.



(a) desync0

(b) desync50

(c) desync100

Fig. 3. Number of traces required to recover a key byte with sufficient energy traces scenario.

We select the results of one of the five repeated experiments to draw Figure 3. Overall, the performance of the BLC leakage model is satisfactory. From the experimental results on desync0, desync50, and desync100, when the energy traces are synchronized or slightly desynchronized, the BLC8 leakage model is more appropriate. The BLC4 leakage model can recover a key byte with fewer energy traces on the three datasets, and the overall performance is better.

E. Experimental Results with Insufficient Energy Traces

In some attack scenarios, due to the constraints of time, equipment, etc., there are not enough energy traces to train the model. In this case, training a classifier with many classes is more challenging than when energy traces are sufficient. We choose several energy traces from the ASCAD profiling set to simulate this scenario.

1) desync0: the detailed experimental results when energy traces are insufficient are shown in Table VI. For desync0 dataset, except for the ID leakage model, the rest of the leakage models successfully recover the correct key byte. The ID model has 256 categories of labels. When the number of samples is 3000, there are less than 12 samples in each class on average. The BLC8 (ϵ=3) leakage model outperforms other methods and can recover one key byte using 446 energy traces.

2) desync50: for desync50 dataset, HW and ID leakage models cannot recover the correct key using all attack energy traces. The BLC4 (ϵ=2) leakage models are significantly better than others and can recover a key byte using 857 energy traces.

3) desync100: as energy trace desynchronization increases, the number of energy traces required to recover one key byte increases more. For desync100 dataset, the BLC4 (ϵ=2) leakage model outperforms other methods and can recover one key byte using 1245 energy traces.

We select the results of one of the five repeated experiments to draw Figure 4. The BLC4 leakage model can recover a key byte with fewer energy traces on the three datasets, and the overall performance is better.

TABLE VI
THE NUMBER OF ENERGY TRACES REQUIRED TO RECOVER THE KEY WITH DIFFERENT LEAKAGE MODELS WHEN ENERGY TRACES ARE INSUFFICIENT.

Dataset	HW	ID	LSB	BLC4	BLC8
desync0	2355	/	532	508	**446**
desync50	/	/	1477	**857**	1630
desync100	5857	/	2721	**1245**	6999
Overall	/	/	4730	**2610**	9075

/ indicates that the correct key cannot be recovered using all attack energy traces.

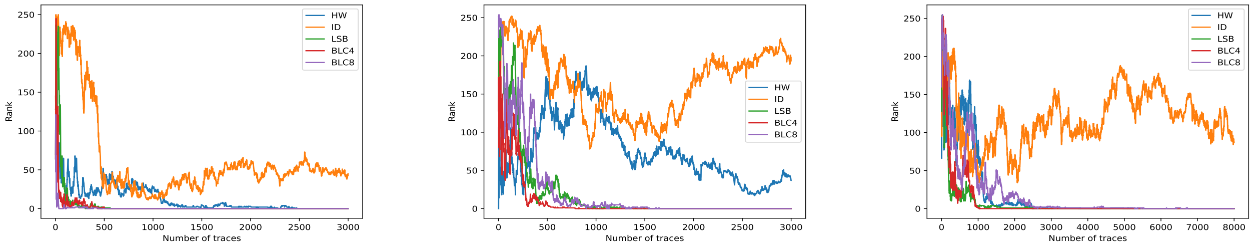
V. PERFORMANCE EVALUATION FOR BLC-VAE WITH INSUFFICIENT ENERGY TRACES

A. Experimental Setup

In this section, we use VAE to improve DLSCA performance when profiling energy traces are lacking. We conduct experiments on the ASCAD synchronization dataset. Specifically, we selected 1000 samples and 3000 samples from ASCAD as the original training set. The validation set is the same size as the training set, but is only used during classifier training. Then, we use the original training set to train the encoder and decoder of the VAE. The structure of VAE is shown in Table I and Table II. The batch size is 100. The learning rate is 0.0005. The epoch is 200. We use the trained VAE to generate the same number of new samples as the original training set. The generated samples are combined with the original samples, and the order is shuffled as a new training set. The detailed dataset configuration is shown in Table III. As shown in Figure 5, the generated energy trace is very similar to the original energy trace, with slight differences around the peaks.

TABLE VII
THE DATASETS CONFIGURATION.

Original training set	Augmented training set	Validation set
1000	2000	1000
3000	6000	3000



(a) desync0

(b) desync50

(c) desync100

Fig. 4. Number of traces required to recover a key byte with insufficient energy traces scenario.

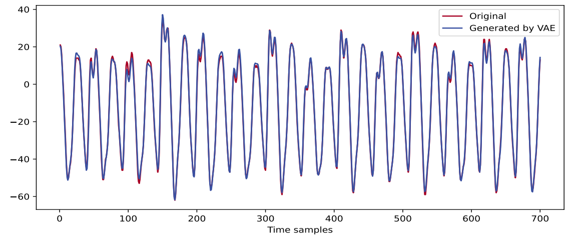


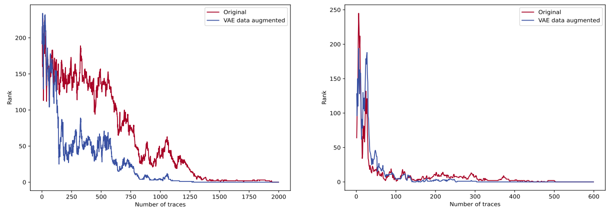
Fig. 5. Comparison of the original energy trace and the generated energy trace.

B. Experimental results

We train classifiers using the original and augmented training sets, respectively, and perform performance comparisons on the attack set. Considering the small number of training samples, we choose BLC4 as the leakage model. The batch size is 100. The epoch is 100. Experiments were repeated five times under the same experimental settings. The experimental results are average of five times.

TABLE VIII
THE DATASETS CONFIGURATION.

Original sample size	Original dataset	Augmented dataset
1000	1902	**1244**
3000	508	**295**



(a) 1000 original energy traces

(b) 3000 original energy traces

Fig. 6. Comparison of attack performance using VAE data augmentation and no data augmentation.

As shown in Table VIII, augmenting the training set with a variational autoencoder can significantly improve the performance of DLSCA. When the original training set samples are 1000, the number of energy traces required to recover a key byte is reduced from 1902 without data augmentation to 1244, a reduction of 34.5%. When the original training set samples are 3000, the number of energy traces required to recover a key byte is reduced from 508 without data augmentation to 295, a reduction of 41.9%.

VI. RELATED WORK

The power, electromagnetic, sound and other signals leaked by cryptographic equipment during the calculation process may be closely related to cryptographic information [16]–[18]. Side-channel attacks are typical exploitation of such signals. Since Kocher first proposed side-channel attacks, SCAs research has been flourishing [19]. Template attacks [3], [20] and machine learning-based profiling SCAs [21], [22] have shown competitive results for some time after the birth of SCAs. In recent years, deep learning techniques have achieved more attractive results in the side-channel community due to their powerful feature learning capabilities [11], [23]. Benadjila *et al.* [2] provided the ASCAD dataset to the SCAs community for researchers to test and compare the performance of new models. Zaid *et al.* [9] constructed an efficient CNN SCA method from the perspective of attack efficiency and network complexity, which improved key recovery performance and reduced network complexity. Zhang *et al.* [24] proposed a multilabel classification SCA method to improve attack performance and reduce network complexity. Zaid *et al.* [25] proposed a SCA method based on stochastic attacks and conditional variational autoencoder provides interpretability for DLSCA.

In recent years, researchers have paid attention to scenarios where the profiling energy traces is insufficient. Pu *et al.* [26] use random shifts on the energy traces to augment the profiled energy traces and show experimentally that this approach can improve the robustness and performance of profiled SCA. Luo *et al.* [13] utilized a technique called mixup in DLSCA to generate new energy traces by mixing different energy traces. Wang *et al.* [27] used conditional generative adversarial networks (CGAN) to generate class-specific energy traces to address the class imbalance due to the Hamming weight model. Picek *et al.* [28] employed the synthetic minority oversampling technique (SMOTE) implements data augmentation and improves SCA performance. Hu *et al.* [29] proposed a subkey combination training method and verified that the method outperforms the individual subkey training model.

VII. Conclusion

In this paper, we propose an SCA leakage model, Balanced Labels Compression (BLC), which can significantly reduce the number of label categories and avoid the sample imbalance problem. We test the BLC leakage model using the ASCAD dataset under two scenarios with sufficient and insufficient energy traces. Experimental results show that the performance of the BLC leakage model is significantly better than that of the HW and ID leakage models. This provides a new leakage model for other DLSCA researchers. We investigate using VAE to augment original side-channel traces to improve the performance of SCA when energy traces are insufficient. We compare the performance without data augmentation and with data augmentation on the ASCAD synchronization dataset. Experimental results show that VAE-based data augmentation can significantly improve DLSCA performance.

References

[1] S. Mangard, E. Oswald, and T. Popp, *Power analysis attacks: Revealing the secrets of smart cards*, vol. 31. Springer Science & Business Media, 2008.

[2] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, and C. Dumas, "Deep learning for side-channel analysis and introduction to ascad database," *Journal of Cryptographic Engineering*, vol. 10, no. 2, pp. 163–188, 2020.

[3] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 13–28, Springer, 2002.

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

[5] 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 Transactions on Cryptographic Hardware and Embedded Systems*, pp. 148–179, 2019.

[6] 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*, pp. 45–68, Springer, 2017.

[7] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *International Conference on Security, Privacy, and Applied Cryptography Engineering*, pp. 3–26, Springer, 2016.

[8] S. Picek, A. Heuser, G. Perin, and S. Guilley, "Profiled side-channel analysis in the efficient attacker framework," in *International Conference on Smart Card Research and Advanced Applications*, pp. 44–63, Springer, 2021.

[9] G. Zaid, L. Bossuet, A. Habrard, and A. Venelli, "Methodology for efficient cnn architectures in profiling attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 1, pp. 1–36, 2020.

[10] L. Wu, G. Perin, and S. Picek, "I choose you: Automated hyperparameter tuning for deep learning-based side-channel analysis," *IEEE Transactions on Emerging Topics in Computing*, 2022.

[11] J. Rijsdijk, L. Wu, G. Perin, and S. Picek, "Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 677–707, 2021.

[12] G. Zaid, L. Bossuet, F. Dassance, A. Habrard, and A. Venelli, "Ranking loss: Maximizing the success rate in deep learning side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 25–55, 2021.

[13] Z. Luo, M. Zheng, P. Wang, M. Jin, J. Zhang, and H. Hu, "Towards strengthening deep learning-based side channel attacks with mixup," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 791–801, IEEE, 2021.

[14] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[15] N. Cai, D. Wang, M. Z. A. Bhuiyan, L. Han, and G. Li, "Lightsca: Lightweight side-channel attack via discrete cosine transform and residual networks," in *2022 IEEE 24th Int Conf on High Performance Computing & Communications*, pp. 793–800, IEEE, 2022.

[16] H. Taneja, J. Kim, J. J. Xu, S. van Schaik, D. Genkin, and Y. Yarom, "Hot pixels: Frequency, power, and temperature attacks on gpus and arm socs," *arXiv preprint arXiv:2305.12784*, 2023.

[17] D. Genkin, A. Shamir, and E. Tromer, "Rsa key extraction via low-bandwidth acoustic cryptanalysis," in *Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34*, pp. 444–461, Springer, 2014.

[18] D. Genkin, I. Pipman, and E. Tromer, "Get your hands off my laptop: physical side-channel key-extraction attacks on pcs: Extended version," *Journal of Cryptographic Engineering*, vol. 5, pp. 95–112, 2015.

[19] 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 16*, pp. 104–113, Springer, 1996.

[20] O. Choudary and M. G. Kuhn, "Efficient template attacks," in *Smart Card Research and Advanced Applications: 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers 12*, pp. 253–270, Springer, 2014.

[21] L. Lerman, G. Bontempi, and O. Markowitch, "A machine learning approach against a masked aes: Reaching the limit of side-channel attacks with a learning model," *Journal of Cryptographic Engineering*, vol. 5, pp. 123–139, 2015.

[22] S. Picek, A. Heuser, A. Jovic, S. A. Ludwig, S. Guilley, D. Jakobovic, and N. Mentens, "Side-channel analysis and machine learning: A practical perspective," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 4095–4102, IEEE, 2017.

[23] L. Wouters, V. Arribas, B. Gierlichs, and B. Preneel, "Revisiting a methodology for efficient cnn architectures in profiling attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 147–168, 2020.

[24] L. Zhang, X. Xing, J. Fan, Z. Wang, and S. Wang, "Multilabel deep learning-based side-channel attack," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 6, pp. 1207–1216, 2020.

[25] G. Zaid, L. Bossuet, M. Carbone, A. Habrard, and A. Venelli, "Conditional variational autoencoder based on stochastic attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 310–357, 2023.

[26] S. Pu, Y. Yu, W. Wang, Z. Guo, J. Liu, D. Gu, L. Wang, and J. Gan, "Trace augmentation: What can be done even before preprocessing in a profiled sca?," in *Smart Card Research and Advanced Applications: 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13–15, 2017, Revised Selected Papers*, pp. 232–247, Springer, 2018.

[27] W. Wan, W. Jun-Nian, H. Fan-Liang, and N. Feng, "Sca-cgan: A new side-channel attack method for imbalanced small samples.," *Radioengineering*, vol. 32, no. 1, p. 125, 2023.

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

[29] F. Hu, H. Wang, and J. Wang, "Cross subkey side channel analysis based on small samples," *Scientific Reports*, vol. 12, no. 1, p. 6254, 2022.