Power Analysis Attack Based on Lightweight Convolutional Neural Network

Xiang Li[1], Ning Yang[1], Aidong Chen[2,4(✉)], Weifeng Liu[3], Xiaoxiao Liu[4], and Na Huang[2,4]

[1] Beijing Key Laboratory of Information Service Engineering, Beijing 100101, China
[2] College of Robotics, Beijing Union University, Beijing 100101, China
aidong@buu.edu.cn
[3] Institute of Semiconductors, Chinese Academy of Sciences, Beijing 100083, China
[4] Research Centre for Multi-intelligent Systems, Beijing 100101, China

Abstract. Since the beginning of the 21st century, modern information technology and electronic integrated circuit technology have developed rapidly. In the chip industry, the ability to resist side-channel attacks has become an important indicator for international mainstream evaluation agencies to evaluate chip security. This paper proposes an improved method for side channel analysis based on the CNN_{best} model, incorporating a lightweight combined channel and space convolutional attention module, optimising the position of the attention module, improving the learning efficiency of key features of the power consumption curve, and effectively reducing the number of traces used by the attack model. The addition of dropout layer network structure solves the problem that the model is prone to rapid overfitting. The optimal value of drop rate is sought through comparative experiments to speed up the convergence of the model and reduce the number of traces required for a successful attack. The experimental results show that the number of traces required by the method in this paper for side-channel attacks is reduced by 88% compared with the original model, which significantly improves the attack performance and can meet the requirements of side-channel modeling and analysis.

Keywords: Side channel analysis · Power consumption attacks · Deep learning · Attention mechanisms

1 Introduction

With the development of cryptography and information technology, the current cryptographic algorithm itself is strong enough. People cannot live without embedded devices, such as smart cards, routers for door locks, etc. A large part of the security of embedded devices relies on cryptographic algorithms to protect them, and mature cryptographic algorithms have been mathematically proven, so brute force attacks are unlikely to break

This work was supported by the Academic Research Projects of Beijing Union University (SK160202103).

E. Ahene and F. Li (Eds.): FCS 2022, CCIS 1726, pp. 105–118, 2022.
https://doi.org/10.1007/978-981-19-8445-7_7

them with current levels of computing power, but side channel attacks target the device, focusing on using physical information leaked during device operation to bypass cryptographic algorithms. This is often an easily overlooked security risk, and research into side channeling can help to remedy these security risks and strengthen device security. Due to the process characteristics of the device itself, the device will leak side channel information such as power consumption, electromagnetism and time during operation, as shown in Fig. 1. If an attacker is able to obtain data on the changes in the energy consumption of the circuit operation, using certain methods to analyse this data, he will be able to obtain some information on the key. The leaked information is directly related to the key or a part of the subkey, and the operations related to this information are called sensitive operations, and the operation obtained is called a sensitive intermediate value. Thereby, the intermediate value leakage model portrays the mapping relationship between the intermediate value of the cryptographic algorithm to the actual leaked value and is an assumption made by the attacker against the cryptographic device [1].

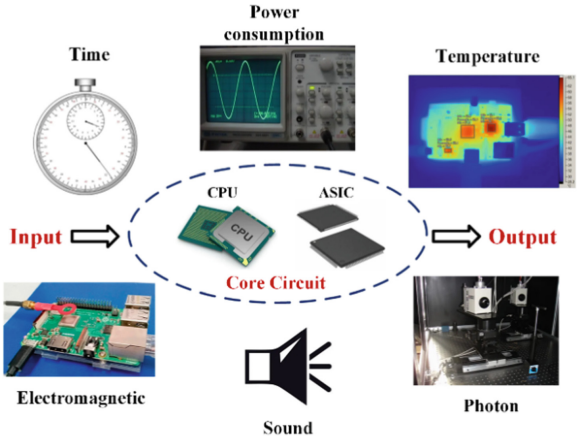

Fig. 1. Side channel information leakage.

Side channel analysis (SCA) can be divided into two categories according to different methods. The first category is the Non-Profiling SCA analysis method, and the second category is the Profiling SCA analysis method. The Non-Profiling SCA analysis method does not require additional equipment, directly collects power consumption on the target chip, and recovers the key by means of statistical analysis. Common Non-Profiling SCA analysis methods include Differential Power Analysis (DPA) [2], Correlated Power Analysis (CPA) [3] and Mutual Information Analysis [4]. In contrast, the Profiling SCA analysis method uses a large number of tagged (plaintext and key) power consumption profiles on a target device that is fully consistent with the target device and then models the power consumption before the key, using this model to attack the target device. Common modeling and analysis methods include Template Attack (TA) [5] and linear regression analysis models [6, 7]. The modeling attack capability of the side channel combined with deep learning is relatively strong. In the attack scenario, if the deep learning technique trains lightweight and easily trained models with good performance,

it is even greatly enhanced the comprehensive attack capability of this technique, which can effectively improve the security testing capability in terms of side channels.

In recent years an increasing number of scholars have concluded that the use of deep learning for side channel modelling is effective, as the greatest advantage of deep learning is that it can learn the feature extraction process to a certain extent and can extract more complex and high-level abstract features. In recent years, the current state of research in the modelling class of side channel analysis using deep learning can be divided into four parts: improved network design or training methods, optimised loss functions, data augmentation and other research directions. With regard to improving network design, in 2020, Ryad et al. used VGG-16 as a starting point to give the effect of hyperparameters on the modelling and attack process, and proposed the CNN_{best} architecture, which provided a great help for further extensive research in this field [8]. In 2021, Lu et al. proposed a neural network architecture that can get rid of the manual extraction of traces on the mask-protected original feature point step and obtain intermediate values by directly analyzing the raw power consumption curve end-to-end [9]. In 2022, Wu et al. proposed a deep learning-assisted template attack, based on a similarity learning approach, using a triadic model for implementation [10]. The evaluation metrics of deep learning models and side-channel attack models are not consistent, with deep learning using accuracy and precision, while side-channel analysis is most commonly used for guessing entropy and success rate. Thus, in 2020, Zhang et al. proposed an evaluation metric CER for the side-channel attack scenario applicable to deep learning, where minimizing $Loss_{CER}$ is equivalent to simultaneously maximizing the score corresponding to the correct key and minimizing the score of the incorrect key [11]. In 2021, Zaid et al. proposed Ranking Loss, where the evaluator uses a prediction function to estimate which intermediate value is processed and uses the power consumption curve to calculate the score, thus computing $\boldsymbol{L_{RKL}}$ [12]. Regarding data augmentation, in 2019, Picek et al. use the data balancing technique SMOTE to analyse minority class samples and manually synthesise new samples to add to the dataset based on minority class samples, reducing the low performance associated with data imbalance [13]. In 2022, Perin et al. investigate the importance of different feature selection, where optimising points of interest leads to better attacks and greater workload [14].

The main contributions of this paper:

(1) Integrate the Convolutional Block Attention Module (CBAM) into the CNN_{best} model, and conduct five insertion position experiments so that to get the optimal insertion position. At the meatime, we modify the feature extraction part, and assign greater weights to the feature points of the power consumption traces, so that the network can learn better Features in the power traces that have strong operational dependencies and strong data dependencies with the encryption step;
(2) A dropout layer is added to our modified CBAM model, and three sets of parameter optimization experiments are carried out to seek the optimal parameter of the drop probability, which solves the problem of rapid over-fitting that is prone to occur in the model, significantly accelerates the speed of model convergence, and effectively improves the accuracy of guessing the correct key.

2 Background Knowledge

2.1 Convolutional Neural Network

Convolutional neural network (CNN) is a variant of MLP. CNN was proposed by Yann Lecun [15], and its essence is an MLP. Each layer in the convolutional neural network has multiple feature maps, each feature map extracts a feature of the input through a convolution filter, and each feature map has multiple neurons, as shown in Fig. 2. The convolutional layer and the pooling layer of the hidden layer are the core modules to realize the feature extraction function of the convolutional neural network. The network model uses the gradient descent method to minimize the loss function to reversely adjust the weight parameters in the network layer by layer, and improve the accuracy of the network through frequent iterative training. The input of the first fully connected layer is the feature image obtained by feature extraction by the convolutional layer and the subsampling layer. The last output layer is a classifier that can use logistic regression, Softmax regression or even a support vector machine to classify the input image.

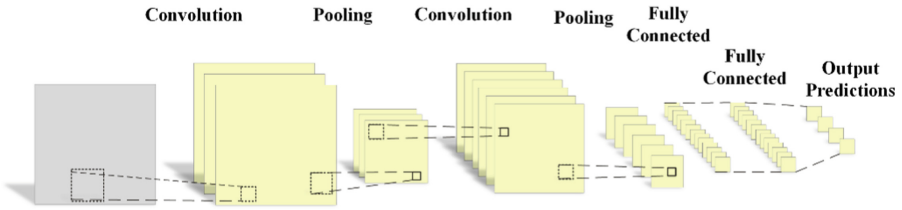

Fig. 2. CNN structure.

2.2 Side Channel Attack Principle

After modeling, the analyst obtains a model $F(\cdot) : X \rightarrow P(Y)$, which can be assimilated into a probability function (possibly normalized). In the attack phase, the analyst tries to recover a fixed k^* and the power consumption profile measured from the target device as $D = \{(x_i^n)\}_{i=1}^n$ can calculate the log-likelihood function score $d[k] = \sum_{i=1}^n log(F(x_i)[f(p_i, k)]$ for all attack power consumption profiles for each $k \in \boldsymbol{K}$. The analyst then selects the key k_{guess} that leads to the highest score in the log-likelihood function, $k_{guess} = argmax_{k \in K} d[k]$. If $k_{guess} = k^*$, the attack succeeds.

2.3 CNN_{best}

IN 2020, the Researchers Detail the Principles of Combining Deep Learning with Template Attacks and How This Works in Practice, Using VGG-16 as a Starting Point, Giving the Impact of Hyperparameters on the Modelling Process and the Attack Process, and Proposing the CNN_{best} Architecture [8]. The Model is Defined as a CNN Architecture with 5 Blocks and 1 Block-By-Block Convolution, a Number of Filters Equal to 64, 128,

256, 512 and 512, a Kernel Size of 11, and a Fill Method of Same-Using ReLU Activation Function and Average Pooling. Two Final Fully Connected Layers, Consisting of 4096 Units. By Using an RMSprop Optimiser with an Initial Learning Rate of 10. This Network Structure is Shown in Fig. 3.

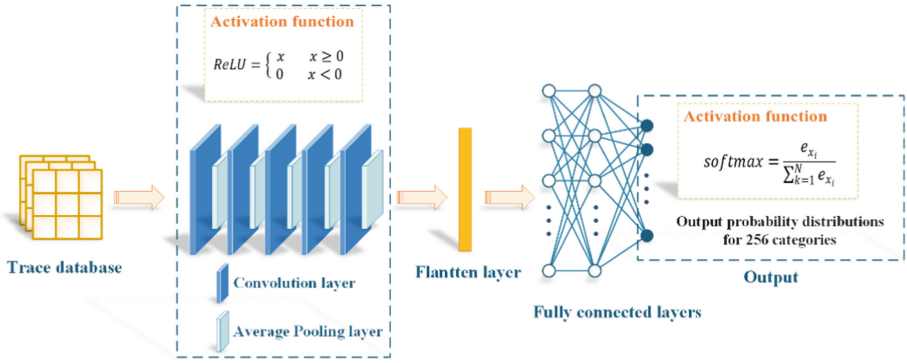

Fig. 3. CNN_{best} structure.

2.4 Evaluation Indicators

Two different metrics were chosen to evaluate the performance of the model: the rank function and the accuracy.

(1) *The rank function*
The *key* used during the acquisition of dataset $D_{profiling}$ using $k^* \in \boldsymbol{K}$. The rank function corresponding to the model trained using dataset D_{train} and tested using dataset D_{test} is defined as $rank\left(\hat{g}, D_{train}, D_{test,n}\right) = \left|\left\{k \in K | d_{n[k]} > d_{n[k^*]}\right\}\right|$. If k^* has the highest (lowest) score, its rank is $0(|K| - 1)$.

(2) *The accuracy*
y_i denotes $|K|$ dimensional output $\hat{g}(l_i, p_i)$, D_{test} may be of unbounded size, as defined in Eq. 1.

$$acc\left(\hat{g}, D_{train}, D_{test}\right) = \frac{\left|\left\{(\iota_i, p_i, k^*) \in D_{test} | k^* = argmax_{k \in K} y_{i[k]}\right\}\right|}{|D_{test}|} \quad (1)$$

3 Methodology

3.1 Convolutional Block Attention Mechanism

In order to improve CNN performance, in addition to studying depth, width and cardinality factors, researchers have also studied increasing the representational power of the network by focusing on important features and suppressing unnecessary features through the attention mechanism, which is one way to achieve adaptive attention in the

network. The data-dependent or operation-dependent time periods required for power consumption curve analysis tend to be a relatively small portion of the entire power consumption curve, meaning that there are small regions with physical information leakage, so to improve the ultimate performance of the model, this paper hopes to give more useful feature data regions greater weight in the useless feature data regions. Electronic noise can affect the performance of the model, so smaller weights are given to the useless feature data regions. At the same time, the above operation may lead to overfitting and slow convergence of the model, so a Dropout layer is added later to alleviate the overfitting and speed up the convergence of the model. The Convolutional Block Attention Module (CBAM) is a lightweight attention module proposed by Woo et al. [16], which combines channel and spatial attention mechanism modules, as shown in Fig. 4. Firstly, the input feature $F \in R^{C\times H\times W}$ is input, then the one-dimensional convolution of CAM $M_C \in R^{C\times 1\times 1}$ is performed, the result of the convolution is multiplied with the original graph, and the output of CAM is used as the input of SAM, while the two-dimensional convolution of SAM $M_S \in R^{1\times H\times W}$ is performed afterwards, and the output is multiplied with the original graph. The procedure is as in Eqs. 2 and 3.

$$F' = M_C(F) \otimes F \tag{2}$$

$$F'' = M_S\left(F'\right) \otimes F' \tag{3}$$

Among them, F represents the input of the feature map $(C \times H \times W)$, M_C is the one-dimensional $(C \times 1 \times 1)$ channel attention map, M_S is the two-dimensional $1 \times H \times W$ channel attention map, $\otimes$ represents the multiplication operation, $F^{'}$ represents the intermediate output $(C \times H \times W)$, and $F^{''}$ represents the final output $(C \times H \times W)$.

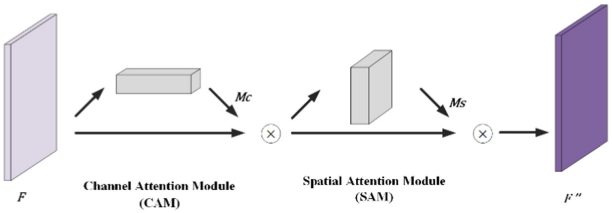

Fig. 4. CBAM structure.

CBAM is generally used for feature extraction of images with two-dimensional convolution, but in this paper one-dimensional convolution is used, so the channel space attention module in this paper is designed for one-dimensional data. And with different insertion positions, the MLP hyperparameters in the channel attention module are slightly different.

Firstly, the Channel Attention module, which uses a global one-dimensional maximum pooling operation and a global one-dimensional average pooling operation to obtain two sets of data. Each channel in the two sets of data has now been compressed into a number, and then two convolutional layers with *kernel size* $= 1$ are used to act as the fully connected layer, the activation function is the ReLU function. The last two sets

of data are summed by the corresponding indexes and the final result is presented by the Sigmoid activation function, as shown in Fig. 5.

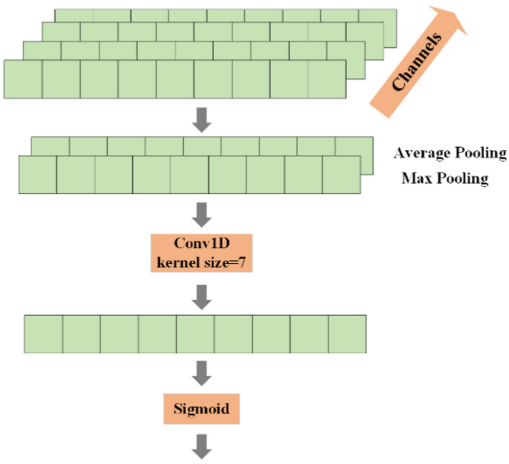

Fig. 5. 1-d Channel attention module.

Secondly is the Spatial Attention module, which first separately conducts average pooling and max pooling operations on the values of each element of the input data on the current channel so that to get two sets of data. And then using a one-dimensional convolution of *kernel size* $=7$ to convert the two sets of data into a set of data, which should be consistent with the size of the input data. Finally, by the Sigmoid activation function, the output of this part can be down, as shown in Fig. 6.

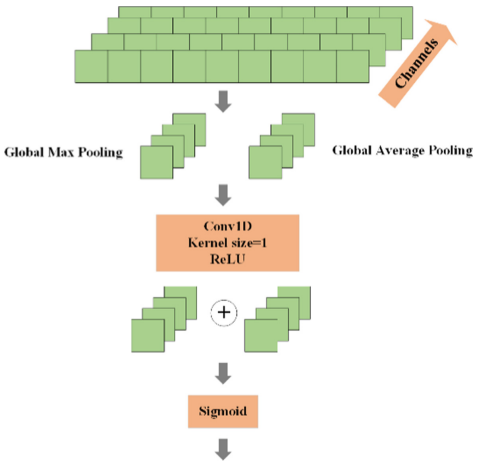

Fig. 6. 1-d Spatial attention module.

The insertion positions of five CBAMs are designed in this paper respectively, at the interval of each layer of the five layers of convolution and behind the fifth layer of convolution, where the MLP neurons number of the channel attention module are set to 0.5 times the number of input channels. Testing the effect of five positions on the performance of the model after insertion of CBAM, and selecting the best performing model for the next step, the CBAM insertion position is shown in Fig. 7.

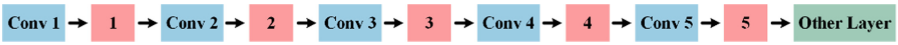

Fig. 7. Location of inserting CBAM.

3.2 Dropout

The problem of overfitting is often encountered when training neural networks. This is reflected in the fact that the model has a small loss function on the training data and has a high prediction accuracy, but has a larger loss function on the test data and has a lower prediction accuracy. If the model is overfitted, the resulting model is almost unusable. Hinton proposed Dropout [17], which tends to cause overfitting when a complex feedforward neural network is trained on a small data set. To prevent overfitting, the performance of a neural network can be improved by blocking the coaction of feature detectors. Subsequently, there have been a number of articles on Dropout [18, 19]. A normal feed-forward neural network is shown in left panel of Fig. 8.

Letting the activation value of a certain neuron stop working with a certain probability p during forward propagation makes the model more generalizable as it does not rely too much on certain local features, dropout principle as shown in right panel of Fig. 8.

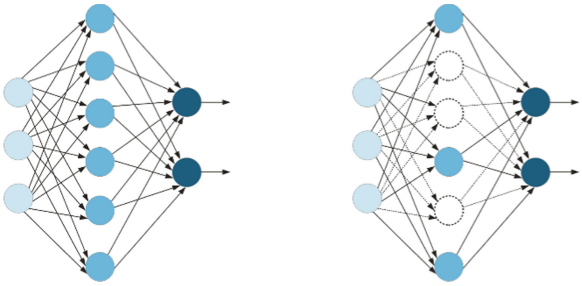

Fig. 8. Left: original network; right: adding dropout to the network.

When performing side-channel attack experiments, the trained model is very prone to rapid overfitting, so the Dropout layer was introduced in the hope of solving the rapid overfit problem. In this paper, the Dropout layer is added after the first fully connected layer in the experiment, and the overall structure is shown in Fig. 9.

Fig. 9. Location of adding dropout.

4 Experiment

4.1 Experimental Platform and Data Set

The experiment is conducted in the CentOS 8 operating system on python 3.6 + pytorch 1.10 and executed by the processor Intel(R) Xeon(R) gold 5218 CPU, memory 128GB, graphics card Nvidia Geforce RTX 2080 Ti, hard disk 512GB SSD + 4TB HDD workstation for training model.

The dataset mainly uses ASCAD public datasets [8], 50,000 training set data, 10,000 test set data, each data contains the power consumption of the third byte for the first round of S-box encryption that has been synchronously aligned, a total of 700 data points.

4.2 Feature Extraction Network Integrated into CBAM

First, in order to verify the performance of the model inserting CBAM at different insertion locations, the experiment trained five models with different insertion locations for testing and comparing them with the original model. These models are all "accuracy optimal" models in the training set to simulate the choices that a model might make in a real attack environment. And in later experimental tests, it may be found that these models may not be the best choice for the attack dataset.

Depending on the insertion location, these models are labeled CBAM1, CBAM2, CBAM3, CBAM4 and CBAM5, respectively, with the number following the tag name referring to Fig. 10, and the original model is labeled No CBAM. The experiment measures the quality of the model from the rank traces of the model and the number of traces required to attack the correct key, so as to select the model for the next experiment.

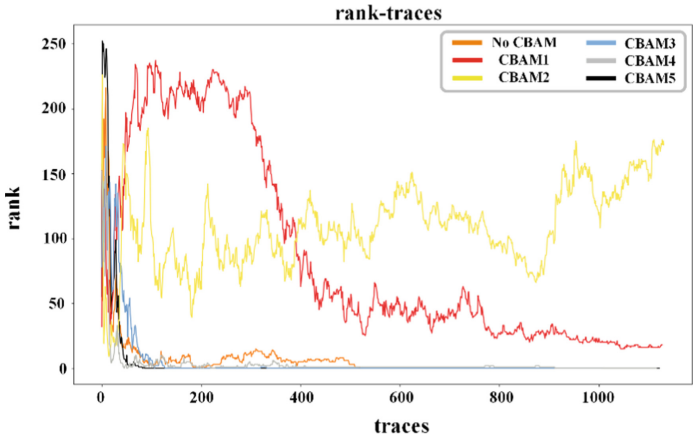

Fig. 10. Rank line chart.

It can be seen from the experimental test rank line chart results that both CBAM3 and CBAM5 predict the correct key earlier than the original model. The CBAM5 performs the best output. The CBAM4 also model performs well in the early stage, but the rank of the model is stable at 1 for a long time, and always cannot be lowered to 0. Finally the rank is successfully lowered to 0 until 913 traces are reached. Besides, CBAM1 and CBAM2 cannot infer the correct key within the 1200 traces, so CBAM1 and CBAM2 are directly abandoned.

Table 1 shows the number of traces required when the correct key rank is stable to 0 when each model was attacking.

Table 1. Number of traces required for each method.

Model	No CBAM	CBAM3	CBAM4	CBAM5
Number of traces	520	128	913	83

In order to understand the change law of the specific performance of the model during each epoch training and compare the changes of the model in the training process between the original model and CBAM5, this paper tests the model generated by each epoch. And using the number of traces required to reduce the rank to 0 for measuring the quality of the model, as shown in Fig. 11. In the experiment, when the number of traces required exceeds 800, this paper regards it as the model cannot attack the true key, and it appears as a peak reaching 1000 in the line chart. It can be seen from the test result curve that although the performance of the CBAM5 model is better than that of the original model, based on the above results, the CBAM5 model is selected for the next experiment.

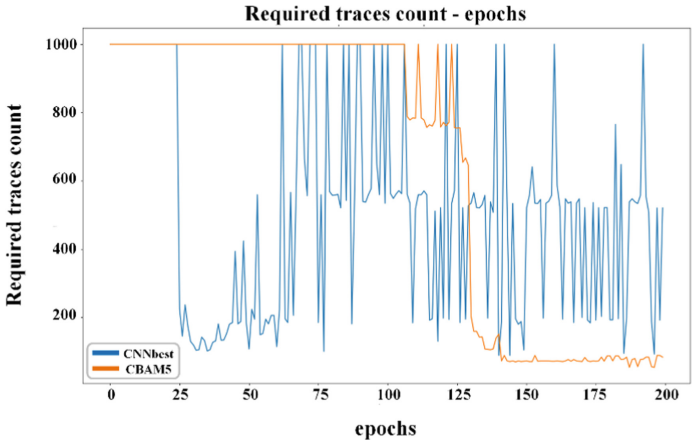

Fig. 11. Comparing CNN_{best} and CBAM5.

4.3 Add Dropout to the Model

Figure 11 shows that the performance of the model inserted with the attention module is improved compared to the original model, and it is also more stable, but the convergence speed is not as fast as the original model, and the index quickly drops to less than 100 after 125 epochs. At the same time, the trend of rapid overfitting of the original model can be clearly seen in this figure.

In order to improve the above problems, this paper conducts an experiment of adding a Dropout layer. The experiment finally selects $p = 0.3$ as the dropout rate added by the improved model. The final experimental results are shown in Fig. 12. Although the accuracy of the classification model is not an effective indicator to measure the performance of the side channel model, the final performance is related to the accuracy of the classification model. The accuracy on the test data set is shown in Table 2. The table is a selection of epochs where each model has the best performance when the rank drops to 0 during training, where the minimum number of traces is needed for comparison.

Table 2. Acc of three models.

Model	NO CBAM/%	CBAM5/%	CBAM + Dropout0.3/%
Acc	0.49	0.5	0.57

Figure 12 clearly shows that after adding Dropout to the model with the attention module, the model converged significantly faster and showed better performance in the subsequent epochs. The model with Dropout added is the model with the highest accuracy.

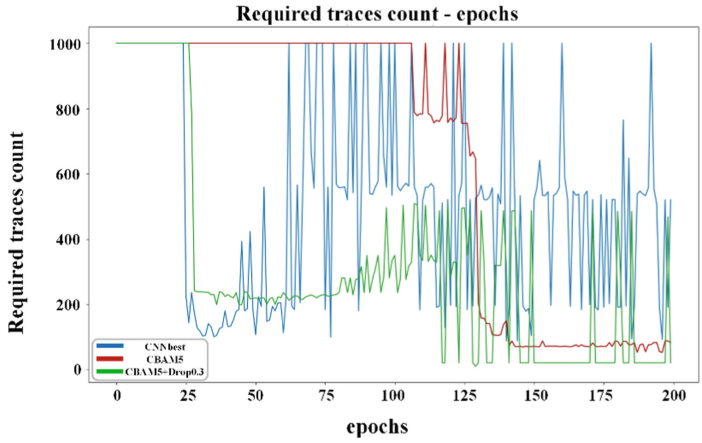

Fig. 12. Compare CNN_{best}, the model with the CBAM5 and the model with both the CBAM5 and the dropout.

Table 3 clearly shows the performance comparison of the three, demonstrating the improved convergence speed and performance results with CBAM5.

Table 3. Minimum number of traces required for different models.

Model	NO CBAM	CBAM5	CBAM5 + Dropout0.3
Number of traces	88	53	10
Epochs	140	187	129

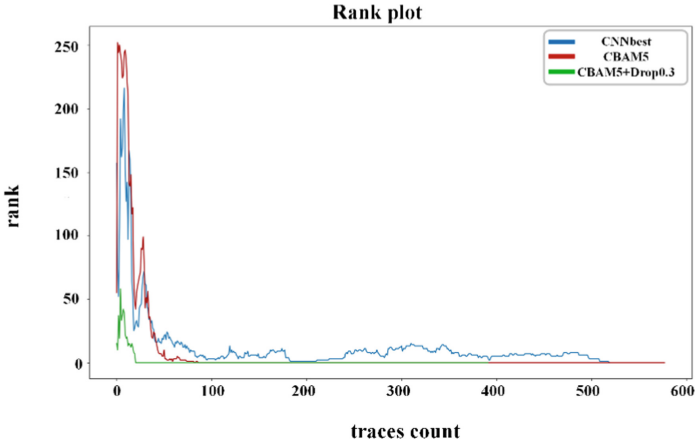

Fig. 13. Rank of three models.

In order to verify the three models more clearly, a comparison of the rank curves of the "best trained" models is shown in Fig. 13, where it is clear that the model with both the attention module and the Dropout layer outperforms the other two models.

Table 4. Results of different model attacks on ASCAD dataset.

Model	CNN_{best} [8]	SincNet [20]	Zaid's CNN [21]	Ours
Number of traces	520	170	191	17

Table 4 shows that the model proposed in [8] requires 520 traces before the rank drops to 0; the model proposed in [20] requires 170 traces for a successful attack on the SincNet network; and the model proposed in [21] requires 191 traces for a successful attack. In contrast, the improved model proposed in this paper requires only 10 traces for a successful attack, and the model is more effective.

5 Conclusion

Information security is the backing of national development. It is an important research direction to organically integrate deep learning theory and technology with classical side-channel attacks, and to explore new analysis methods and evaluation indicators. At present, with the rapid development of science and technology, side-channel attacks can find vulnerabilities in cryptographic algorithms, and can also prompt researchers to improve the defense capabilities of cryptographic algorithms. The two are interdependent and jointly promote the development of cryptographic algorithms and information security. This paper proposes an improved side-channel modeling attack method based on the CNN_{best} network. By introducing the CBAM in the best position, the feature extraction network is optimized to suppress noise features and improve the learning of key features; adding a dropout layer can effectively alleviate the over-modeling. The fitting phenomenon occurs, helping to reduce the number of power consumption traces used by the attack. The experimental results show that the model proposed in this paper can effectively improve the modeling side channel analysis method based on neural network. Considering the power consumption traces collected in reality, due to hidden countermeasures or device settings, an offset power consumption traces, that is, an asynchronous power consumption traces, will be generated. This is the direction we need to improve the network adaptation.

References

1. Mangard, S., Oswald, E., Standaert, F.X.: One for all–all for one: unifying standard differential power analysis attacks. IET Inf. Secur. **5**(2), 100–110 (2011)
2. Kocher, P., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. J. Cryptogr. Eng. **1**(1), 5–27 (2011)
3. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28632-5_2
4. Batina, L., Gierlichs, B., Prouff, E., Rivain, M., Standaert, F.X., Veyrat-Charvillon, N.: Mutual information analysis: a comprehensive study. J. Cryptol. **24**(2), 269–291 (2011)
5. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski, B.S., Koç, çK., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36400-5_3
6. Doget, J., Prouff, E., Rivain, M., Standaert, F.X.: Univariate side channel attacks and leakage modeling. J. Cryptogr. Eng. **1**(2), 123–144 (2011)
7. Schindler, W.: Advanced stochastic methods in side channel analysis on block ciphers in the presence of masking. J. Math. Cryptology **2**(3), 291–310 (2008)
8. Benadjila, R., Prouff, E., Strullu, R., Cagli, E., Dumas, C.: Deep learning for side-channel analysis and introduction to ascad database. J. Cryptogr. Eng. **10**(2), 163–188 (2020)
9. Lu, X., Zhang, C., Cao, P., Gu, D., Lu, H.: Pay attention to raw traces: a deep learning architecture for end-to-end profiling attacks. IACR Trans. Cryptographic Hardware Embed. Syst. **2021**(3), 235–274 (2021)
10. Wu, L., Perin, G., Picek, S.: The best of two worlds: deep learning-assisted template attack. IACR Trans. Cryptographic Hardware Embed. Syst. **2022**(3), 413–437 (2011)

11. Zhang, J., Zheng, M., Nan, J., Hu, H., Yu, N.: A novel evaluation metric for deep learning-based side channel analysis and its extended application to imbalanced data. IACR Trans. Cryptographic Hardware Embed. Syst. **2020**(3), 73–96 (2020)
12. Zaid, G., Bossuet, L., Dassance, F., Habrard, A., Venelli, A.: Ranking loss: maximizing the success rate in deep learning side-channel analysis. IACR Trans. Cryptographic Hardware Embed. Syst. **2021**(1), 25–55 (2021)
13. Picek, S., Heuser, A., Jovic, A., Bhasin, S., Regazzoni, F.: The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. IACR Trans. Cryptographic Hardware Embed. Syst. **2019**(1), 1–29 (2019)
14. Perin, G., Wu, L., Picek, S.: Exploring feature selection scenarios for deep learning-based side-channel analysis. IACR Trans. Cryptographic Hardware Embed. Syst. **2022**(4), 828–861 (2022)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
16. Woo, S., Park, J., Lee, J.-Y., Kweon, I.S.: CBAM: convolutional block attention module. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 3–19. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_1
17. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing coadaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
18. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
19. Bouthillier, X., Konda, K., Vincent, P., Memisevic, R.: Dropout as data augmentation. arXiv preprint arXiv:1506.08700 (2015)
20. Chen, P., Wang, P., Dong, G., Hu, H.: SincNet-based side channel attack. J. Cryptologic Res. **7**(5), 583–594 (2020)
21. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Methodology for efficient CNN architectures in profiling attacks. IACR Trans. Cryptographic Hardware Embed. Syst. **2020**(1), 1–36 (2020)