Deep Learning Method for Power Side-Channel Analysis on Chip Leakages

Amjed Abbas Ahmed[1,3,*], Rana Ali Salim[2], Mohammad Kamrul Hasan[1]
*[1]Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia,
Bangi, Selangor 43600, Malaysia*
*[2]College of Fine Arts, University of Baghdad,
Baghdad, Iraq*
*[3]Department of Computer Techniques Engineering, Imam Al-Kadhum College (IKC),
Baghdad, Iraq*
[*]amjedabbas@alkadhum-col.edu.iq; raname78@yahoo.com; mkhasan@ukm.edu.my

***Abstract*—Power side channel analysis signal analysis is automated using deep learning. Signal processing and cryptanalytic techniques are necessary components of power side channel analysis. Chip leakages can be found using a classification approach called deep learning. In addition to this, we do this so that the deep learning network can automatically tackle signal processing difficulties such as re-alignment and noise reduction. We were able to break minimally protected Advanced Encryption Standard (AES), as well as masking-countermeasure AES and protected elliptic-curve cryptography (ECC). These results demonstrate that the attacker knowledge required for side channel analysis, which had previously placed a significant emphasis on human abilities, is decreasing. This research will appeal to individuals with a technical background who have an interest in deep learning, side channel analysis, and security.**

***Index Terms*—AES implementation; Convolutional neural network; Deep learning; Neural network; Side-channel analysis.**

I. INTRODUCTION

Side channel attack (SCA) is the most typical kind of attack [1]. SCA can uncover previously hidden information using data obtained from the execution of algorithms. Time, electric current, and electromagnetic radiation are all examples of types of leak. SCA searches for device keys. This study is made possible by the use of algorithm leakages and secret keys. An attacker will utilise models and metrics to gather leakages to discover the secret key. Consequently, it is possible to classify SCA. The patterns in the data are recognised by the machine learning algorithms. Due to the fact that it classifies data, machine learning can be used to SCA. To get better results, machine learning requires human engineering.

Amrouche, Boubchir, and Yahiaoui [2] have been investigating "deep learning" since the middle of the 1990s. Deep learning was first disregarded because there was inadequate processing power and a lack of training data. Deep learning algorithms have seen significant improvement over the last ten years due to the availability of enormous amounts of data and increased computing power [3]. Recent studies have proven that deep learning algorithms are effective in processing images, audio, and spoken languages. The use of deep learning in SCA is enhanced by these findings. Side-channel analysers will likely be interested in the process through which deep learning automatically learns features and generalises data representation. It would be beneficial for deep learning-based side channel attack (DLSCA) to include deeper learning. Recent studies have shown that supervised machine learning (SML) can adapt deep learning features to improve the accuracy of key retrieval and classification [4]. In a number of different trials, deep learning has been shown to be more effective than side-channel attacks. SCA should use deep learning techniques.

II. LITERATURE REVIEW

Multilayer perceptrons (MLPs), which use numerical values without taking into account the data topology, are less resistant to the effects of data distortions than convolutional neural networks (CNNs) [1]. Despite the fact that data distortion might be distorted, CNNs are known to perform well when it comes to picture identification. SCA traces get distorted when side-channel countermeasures and measurement environment noise are used. Therefore, CNNs are an obvious choice for SCA.

The CNN architecture was used for SCA by Maghrebi, as well as others [2]. Cagli and colleagues proposed a CNN-based SCA that could be used on a protected Advanced Encryption Standard (AES) and hide information via jitter [3]. They began by proving that a CNN could remove jitter-based concealing countermeasures even before the data were preprocessed. In [5], the resilience to CNN data distortion was shown by learning the Sbox output of AES while it was protected from clock jitter and random delay insertion. CNN has fewer weights to train than MLP does, but it requires a significant amount of learning data to understand the general invariant properties of traces coming from a concealing method-protected device. In addition to this, the authors advised augmenting the data to circumvent hiding techniques and prevent overfitting. Emulation of jitter-based concealing tactics was achieved by randomly relocating actual traces and adding or deleting a predetermined number of sites at random. To augment training, these simulated traces were used.

Manuscript received 2 June, 2023; accepted 11 October, 2023.

Therefore, the training data were sufficient for the purpose of learning. According to the results of their investigation, CNN-based SCA does not need trace alignment. CNN-based SCA has the potential to potentially provide an objective assessment of side-channel resistance. According to the findings of the study, DLSCA is improved by CNNs that have more input neurons. The extraction of the features and classification or regression of those features are the two primary components of CNN. Hettwer and his colleagues proposed using a CNN that had neurons that represented domain knowledge (DK) [6]. DK neurons are sent to the fully connected layer, where they are used for feature categorisation and regression. Their investigations made the assumption that having this additional information can improve learning performance. DK neurons improved performance. Learning the round key is more effective than learning the Sbox output. Hettwer, Horn, Gehrer, and Güneysu [7] were unable to explain why memorising the round key as a label would be more effective than using Sbox, which is why further study is required. Kwon, Hong, and Kim [8] explained that the spectrogram is superior for CNN since it simultaneously provides information on time and frequency. In the temporal domain, spectrogram-based DLSCA is analogous to power-trace-based DLSCA in terms of performance. Kim and others proposed adding artificial noise to the input trace to improve the robustness of the DLSCA. This is analogous to the denoising autoencoder.

Carbone and his colleagues [9] proposed a CNN-based profiled SCA as a side-channel countermeasure to be used in conjunction with a secure Rivest-Shamir-Adleman (RSA) implementation that included message, exponent, and modulus blinding. The authors presented the demonstration of the effectiveness of DLSCA against public-key cryptosystems. The CNN design protected the confidentiality of the addresses and values of the register. The authors proved that DLSCA is capable of evaluating cryptosystems that use public keys. Following the development of CNN-based DLSCA, they evaluated its performance compared to that of earlier profiling SCA. It was recommended that the ASCAD public dataset be used to evaluate DLSCA and other SCA methods for profiling in a fair manner. In addition, emphasis was placed on reproducibility by providing hyperparameters to share the results of previous studies [10]–[14].

III. METHODOLOGY

A. How Deep Learning Works

There are several layers in neural networks. There must be one input layer and one output layer in the neural network construction. There are not many hidden layers between the input and output layers in classical neural networks. Traditionally, structures have one to three hidden levels. Deep neural networks can create structures with hidden layers that are substantially more than this, ranging from tenths to hundreds. A multiple layer perceptron or fully connected network is shown in Fig. 1.

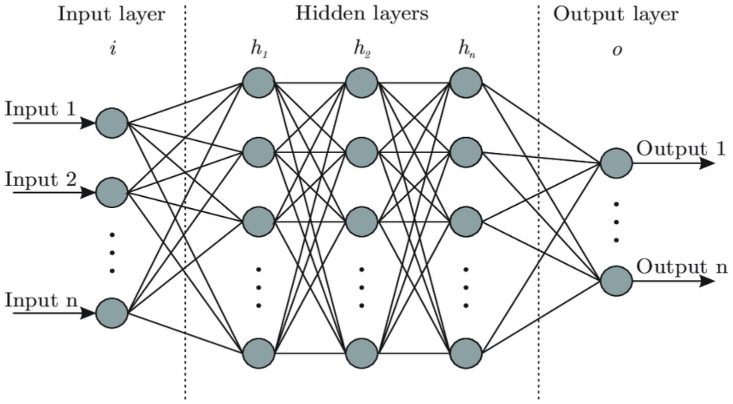


Fig. 1. Basic neural network structure.

It contains a single layer of densely linked neurons. Neurons in hidden layers are tightly connected to those in their preceding and following layers. Common artificial neural network (ANN) topologies include convolutional and multiple layer perceptron networks. In side-channel analysis, these two topologies have shown their efficacy against first-order AES masking and time-desynchronised traces. In the third case, convolutional neural networks are especially useful, since they can identify leaks at numerous trace locations.

B. Multiple Layer Perceptron (MLP)

MLP is a feed-forward ANN. It has a function named F that consists of many non-linear activation and several linear functions. These cells were formerly called perceptrons. The MLP layers have many neurons. The neuron, the fundamental building block, is connected to all the neurons in the layer below and above it. This makes it the fully-connected neural network design. Connection weights define how each neuron links to the layer above or below it. All neurons have bias values and activation functions. Training with the MLP perceptron changes these properties. Activation functions include RELU, TANH, and Sigmoid. The activation value of a neuron output is defined in (1)

$$a = f\left(\sum_{i=1}^{n} w_i I_i + bias\right). \tag{1}$$

Here f represents the activation function, w_i represents the weight connection, I_i represents the activation value of the neuron i in the layer above it, bias represents the bias value, and n represents the number of neurons in the layer above it.

C. Convolutional Neural Networks

Convolutional neural networks (CNNs) can automatically extract a wide range of features. The authors, Hu and Ni, in the research work in [15], increased the accuracy of image item recognition. This implementation excel at processing one-dimensional information such as time series. CNNs are built from an output layer, a dense network, and a convolution layer (or many convolution layers). Figure 2 depicts a two-layer convolutional neural network. Three components make up the convolution layer: the convolution operation, the RELU activation function, and the pooling layer. The downsampling performed by the pooling layer reduces the dimensionality of the data. Each element of this convolution layer is given a weight and a bias. This is accomplished by using a certain number of convolution filters, all of which have the same stride and kernel size. Input convolution is defined here. The filter muddles the information that is being input. Therefore, it is expected that the results of the convolution layers would be filtered features. After the convolution output has been pooled (if desired), it is fed into an activation function.

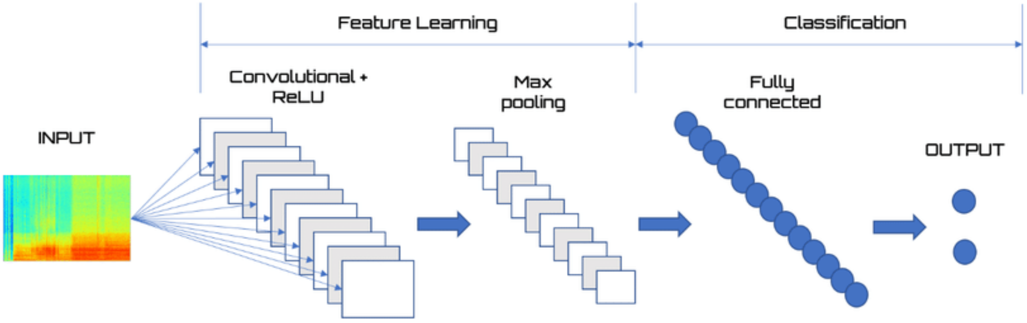


Fig. 2. Example of a CNN model.

The weights and bias of the convolution filter are adjusted at each iteration. The updater mechanism, the regularisation method, and the learning rate all have direct effects on backpropagation-based updates.

D. *Deep Learning for Side-Channel Analysis*

During side-channel analysis, neural networks are used to detect trace leakages [16]. This leakage is caused by the encryption process, which results in power consumption (or electromagnetic emission), as well as intermediate states being handled. The values of the intermediate states are determined by the input and key material. Acquisition noise minimises side-channel leakage.

To begin, get a solid understanding of the deep learning architecture behind side-channel attacks. Learning through supervision requires either an open sample (which can have a user-defined key) or a closed sample with a known key. After the acquisition, the key and input are written on each side-channel trace that was obtained. In Fig. 3, the trace set is seen after being divided into the training set and the validation set. A new trace set would ideally be measured from a second identical device using a key that was kept a secret, since this would be the best possible scenario. The subsequent step is to test this entire trace collection. Due to the high computational burden of deep learning, the training phase is somewhat slow, but the validation and test phases are quite quick.

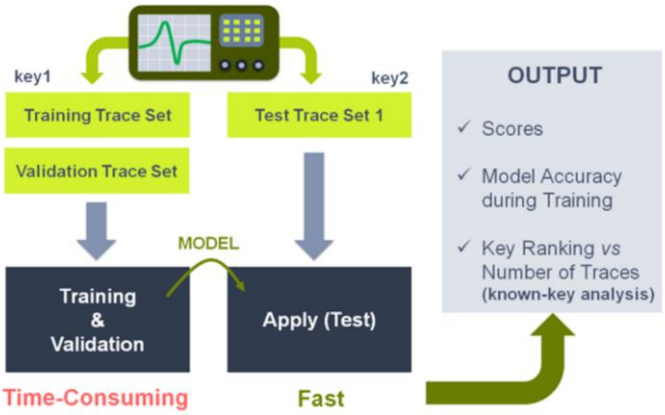


Fig. 3. Proposed framework.

The use of leaky models, such as differential power analysis and template attacks (TA) [17], is necessary for deep learning. Multiple-layer perceptron or convolutional neural networks require supervised evaluation. The leaky model must label training and validation traces. The attacker also specifies how many classes a neural network must detect and organise depending on input trace collecting. A neural network expects to see the following training data classes if the leaky model is created on the S-box output of the first AES round [18]:

– 9, if the Hamming weight model is chosen;
– 256, if an identity model is chosen;
– 2, if the leakage model indicates a 1-bit in the S-Box output.

According to the principle of learning [19], to learn more output classes, one needs more training traces. To identify the classes, the neural network has to examine enough examples of each. The neural network cannot be accurate or generalised if this condition persists because both the training key and the validation key are identical when using a closed sample for training; it is important to avoid developing an identity-leaking model in this scenario. This situation requires the Hamming weight-leaking model. Deep learning with leaky model labels classifies side-channel traces. The attacker can divide and conquer one key byte at a time with symmetric key algorithms such as AES. Therefore, a neural network has to be trained the same number of times as there are bytes in the key. By categorising traces, the attacker is rendered unable to reveal the key. This calls for some kind of key enumeration. Class classification probabilities can be generated using neural networks. Combining these class probabilities with the key byte hypothesis allows for the calculation of key byte candidate likelihoods.

1. Training Accuracy

A neural network will convert data with a certain label class as input and produce probabilities as output. These probabilities will indicate "how much" the input data will fit each class. It is possible that a miniature training set will be too big for the model. The trained neural networks can classify the smaller training sets with almost perfect accuracy, but they are unable to generalise their findings to data from other sources. To stop this from happening, the training set has to have at least enough traces to account for the number of parameters of the neural network.

To achieve minimal classification accuracy, the size of the training set must be validated. A strategy that can enumerate keys or recover lost keys uses a trace set that was classified by a neural network according to a leaking model. Using the output layer probability, this approach ranks the key candidates and returns the top candidates. Even with poor validation (or test) set classification accuracy, it is possible for key recovery to be successful. Symmetric algorithms like AES evaluate numerous traces with the same key material during the attack phase; thus, the key guess must be more probable than the wrong key guess. The process of enumerating potential keys has the potential to cut down on wrong key hypotheses and speed up arrival at the right one. The accuracy of classification can be improved by using larger training sets. Therefore, major recoveries can be made with minimal traces in validation and testing.

Precision is important when training. By monitoring the level of training accuracy that is produced after each epoch that is processed, the user can ascertain whether or not the neural network has sufficient training traces to learn and generalise. The evolution of training accuracy demonstrates whether or not the backpropagation algorithm has converged to the appropriate weight and bias values. During training, the learning rate has an effect on the steps of the backpropagation algorithm. The progression of training accuracy is shown in Fig. 4, which shows what happens when the learning rate is too high to allow the model to settle after 250 epochs.

Stability will be reduced using training that incorporates a learning rate decay rate. Training brings the learning rate up to date (and in most cases, lowers it). After 250 epochs, as shown in Fig. 5, both the accuracy and the loss function evolve to their final states.

The phase of deep learning and neural networks known as hyperparameter selection is considered the most important step. Different hyperparameters have varying effects on the performance of neural networks.

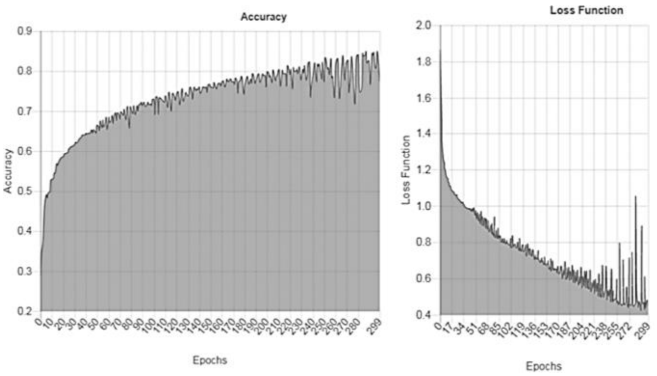


Fig. 4. Training accuracy with higher learning rates.

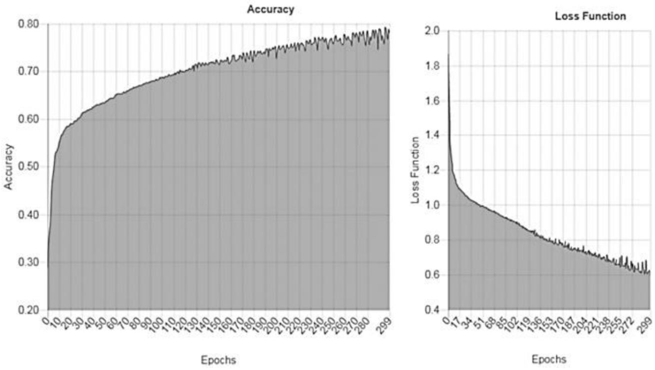


Fig. 5. Training accuracy with corrected learning rates.

Training time might vary depending on the learning rate, the size of the mini-batch, and the epoch. There are several hyperparameters that influence generalisation. There are a few different approaches to generalisation in deep learning:

– Dropout;
– Early stopping;
– Regularisation L1;
– Weight decay;
– Data augmentation;
– Batch normalisation.

Regularisation weight modifications punish L1 and L2 to prevent them from becoming too large. After a lengthy period of training, an instance of overfitting occurs when the weights of the neural network become too large. L1 and L2 regularisers avoid this problem. L1 assesses a penalty for weight amounts. The weight sum squared is the variable that is penalised by L2. In addition, regularisation of L1 removes characteristics that are not essential by setting the weights of these characteristics to zero. The optimal values for the regularisation parameters L1 and L2 result in a less accurate training and facilitate a simpler generalisation. This is because the model no longer completely matches the input data. Similar effects are produced by increasing the number of training traces.

During training, the neurons in the layers are intermittently turned off via dropout. During the training process, this method prevents a single neuron from dominating the attention of all the other neurons in a layer to prevent overfitting. It is essential to stop training early and take the best model out of the data before moving on. The accuracy of the validation should be used to decide whether to stop early.

The primary data set is altered through the data augmentation process to enhance the ability of the neural network to deal with a wide variety of data properties. The size of the training mini-batch has an effect on generalisation. The stochastic gradient descent approach, along with its many variants, is used in mini-batch regimes (32, 64, and 512) [5]. Generalisation errors [20] become more prevalent when larger batch sizes are used. The study will establish a cutoff point for the quality the model, after which it will begin to deteriorate if the batch size for a particular problem increases. The accuracy of the tests suffers. Training with complete batches brings down the cost of the function more gradually than training with mini-batches does. There is also the possibility of warm-starting with a tiny batch and gradually expanding it over the course of epochs. Additionally, small batch sizes have an intrinsic influence on regularisation and can be as a result of noise effects in the learning process. A very low learning rate (0.001 or less) is required for very small batch sizes to maintain stability [21].

2. The number of hidden layers

Deep learning derives its name from the tens of hidden layers that are used in neural networks [22]. On the other hand, a side-channel attack that requires just a moderate amount of accuracy can sometimes recover a key without a significant number of hidden layers. Both the training set and the input samples have an impact on the number of hidden layers that are created. The neural network is able to learn sophisticated leakages from a training set when it has extra hidden layers. This is beneficial for use with large training sets. A bigger neural network will, of course, have more internal parameters, which means that the training process will take longer and need more computational resources. After the neural network converges, having more hidden layers helps reduce the value of the loss function more quickly. In this particular scenario, the training phase could need a smaller number of epochs to produce desirable results or achieve crucial recovery. According to what was covered in Section III-A, it is possible for a neural network to readily overfit the training set if it has an excessive number of parameters for the input data. Processing an epoch might take a significant amount of time if there are many hidden layers.

3. Automatic search for hyperparameter

Parameterisation is a key feature of neural networks [23]. Expertise in neural networks is required for the majority of hyperparameter settings. However, the training accuracy and validation recall are what decide the regularisation hyperparameters, learning rates, epochs, and mini-batch sizes. Activation functions are also determined by these two factors. To accomplish this goal, the neural networks will need to undergo extensive training on several occasions, during which the user will need to make incremental changes to the hyperparameters in order to converge on either a local or global minimum in the landscape. It is necessary to conduct a number of tests to locate the optimal combination of hyperparameters, which consumes both time and computing resources. This is so because neural networks need to have their training reset whenever a new batch of hyperparameters is used. Grid, Random, and Optimised searches are examples of hybrid parameter search solution.

The most basic forms of hyperparameter search are the random and grid varieties [24]. Without doing any optimisation, both methods look for hyperparameters that fall inside predetermined bounds. The model with the highest accuracy, recall, and loss function is selected. According to the findings of the study, random search is superior to grid search. Techniques such as simulated annealing, evolutionary algorithms (also known as genetic algorithms), and Bayesian optimisation are used in optimised searches [25].

IV. PRACTICAL EXPERIMENTS

This section contains deep learning experiments that aim to recover cryptography keys from countermeasure implementation.

A. *Bypassing Misalignments with CNNs*

CNNs can process traces that are not properly aligned, as stated in [15]. According to the findings of this work, data augmentation has the potential to help CNNs in overcome misalignment, as well as jitter-based countermeasures. The severe misalignment is seen in an AES software implementation in Fig. 6. The first processing performed by the S-box is shown here. There are now 45,000 trained traces and 5,000 certified traces.

Convolutional neural networks that have been trained have two layers of convolution and three layers of dense layers. The training set is partitioned into nine different groups by the hamming weights. The ranks that are crucial for CNNs, correlation power studies and TAs are shown in Fig. 7. Only CNN is capable of recovering the key against the target that was analysed.

B. *Breaking of a Masked Advanced Encryption Standard*

The public database DPAcontestV4 is an application case. First-order masking countermeasure AES target trace set. This countermeasure eliminates the power consumption relationship between the intermediate cryptographic steps projected. 40000 AES-256 software power side-channel traces exist in the database. Low-entropy rotating S-box masking (RSM) protects the implementation against first-order attacks. Figure 8 depicts the traces.

The whole trace displays the power consumption during the first AES cycle. Each raw trace consists of 435000 individual samples. We are intrigued by S-boxes, as well as shift rows.

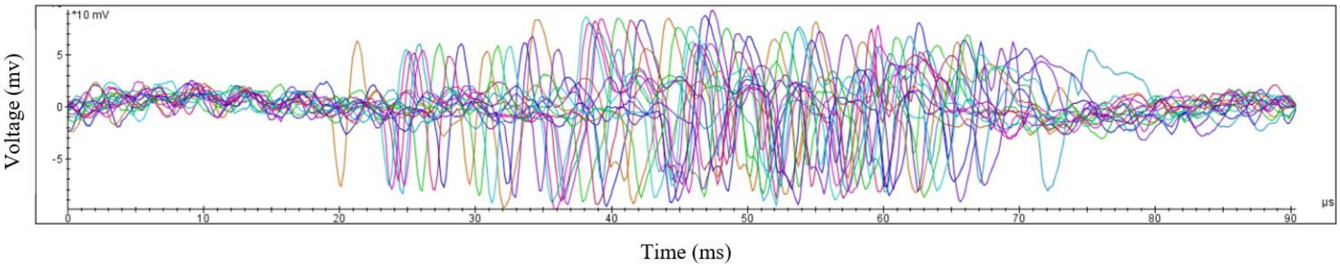


Fig. 6. AES traces with misalignment.

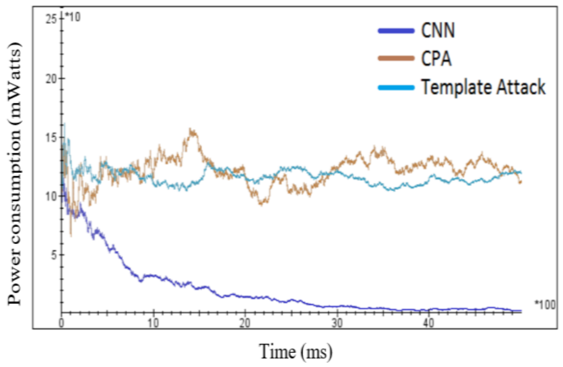


Fig. 7. AES traces with misalignment.

The examination of black boxes using deep learning requires around 200,000 samples. Memory and processing time constraints on our systems prohibit us from training with as many input samples per trace. To train the neural network for all intervals, the input trace should be segmented into smaller portions.

V. RESULTS AND DISCUSSION

The neural network is trained using many black-box intervals. As a result, we determine the high-accuracy validation interval for the target key byte. It was sufficient to determine the period that included the samples with the highest leakage rate, but it was not sufficient to extract the key. After improving the training of the neural network on the specified timeframe, key recovery was achieved successfully. We trained a convolutional neural network using only 40,000 traces and 1000 samples to recover the key bytes.

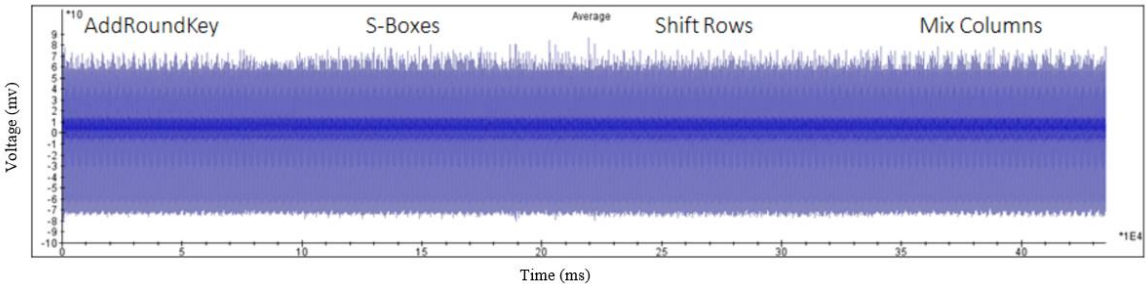
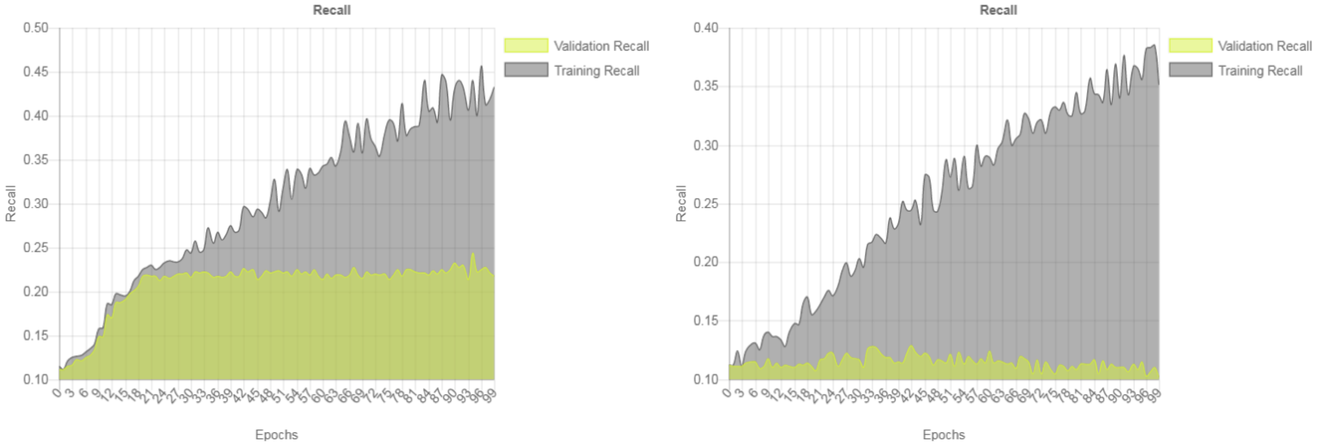


Fig. 8. DPAcontestV4 traces.

The network has one convolution layer, three fully connected layers, and one fully connected layer. During the key enumeration stage, it is often sufficient to have 10 traces to have the correct key byte rank first.

To make key recovery believable, profiled side-channel analysis needs a number of different keys to be trained on and validated (or tested). The neural network can learn from leakage or other input trace characteristics if the same key is used for training and validation. For the purpose of ensuring that this is not happening in the current environment, the neural network is trained and tested using both right and wrong (random) key bytes. Correct and incorrect label recalls from training and validation are shown in Fig. 9. The recall score in classes with uneven student participation is equal to the class average of correct responses.



(a) (b)

Fig. 9. Training and validation recalls: (a) Correct labels; (b) Wrong labels.

As can be seen in the graphic, erroneous labels bring validation recall and generalisation down to around 11 %, which is an indication that our model is picking up information from the traces that it is not supposed to. On the other hand, a 22 % validation recall for the correct labels was sufficient for key recovery. Consequently, a disguised AES implementation was broken down using deep learning.

The hardware used in elliptic curve cryptography has a variety of countermeasures to prevent side-channel analysis. Side-channel analysis can be avoided by using techniques. The goal of this project is to implement scalar multiplication using the Montgomery ladder while maintaining scalar and coordinate blindness. Curve 25519 is used for implementation [16]. In protected public-key implementations such as RSA, side-channel analysis is carried out in a manner that differs from that described in the study. To avoid this restriction, the attackers have to focus their attention on a single trace. In the event that an adversary is able to reconstruct the randomised private key from a single trace, a simple calculation will reveal the actual private key. This is achieved through the use of horizontal side-channel attacks.

The following describes how horizontal attacks are carried out. A component of the power consumption trace is used to represent each bit that is involved in the multiplication of scalar values. Each subsidiary component of the Montgomery ladder performs an addition and a doubling of points. Even in supervised horizontal attacks, the performance of identifying areas of interest might be negatively impacted by misalignment.

In our work using error-correcting codes (ECCs), deep learning is able to recover one hundred percent of the scalar bits when the traces are time-aligned. Attacks based on profiled templates can be designed to match the performance of the attacked trace set. Around 90 % of the lost scalar bits can be recovered using convolutional neural networks. The most effective hyperparameters improve the accuracy of categorisation. The accuracy of classification is improved by using data augmentation as an approach to regularisation. To do this, we augment the initial training set with a number of randomly shifted traces. This strategy provides the network with more mismatched traces from which it can learn. The accuracy of the categorisation of the test phase was increased to 99.4 % with data augmentation. The remaining can be recovered by brute force.

VI. CONCLUSIONS

Side-channel analysis and deep learning are also topics covered in this study. The use of neural networks as important recovery models allows us to understand side-channel attacks. The training of SCA neural networks is described in the paper. CNNs are able to successfully correct trace misalignment and extract crucial bytes from a wide variety of destinations. We used a convolutional neural network to carry out a black-box attack against the masked AES used in the DPAcontestV4. In conclusion, CNNs are instructed to learn how to avoid ECC trace misalignment. When applying the approach, single ECC traces were correct 99.4 % of the time.

CONFLICT OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] G. Vijayakanthi, J. P. Mohanty, A. K. Swain, and K. Mahapatra, “Differential metric based deep learning methodology for non-profiled Side Channel Analysis”, in *Proc. of 2021 IEEE International Symposium on Smart Electronic Systems (iSES)*, 2021, pp. 200–203. DOI: 10.1109/iSES52644.2021.00054.

[2] A. Amrouche, L. Boubchir, and S. Yahiaoui, “Side channel attack using machine learning”, in *Proc. of 2022 Ninth International Conference on Software Defined Systems (SDS)*, 2022, pp. 1–5. DOI: 10.1109/SDS57574.2022.10062906.

[3] X. Jin, J. Feng, and B. Huang, “Side channel attack on SM4 algorithm with deep learning-based analysis”, in *Proc. of 2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 2022, pp. 749–752. DOI: 10.1109/AEECA55500.2022.9919093.

[4] 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. 136448–136458, 2021. DOI: 10.1109/ACCESS.2021.3117761.

[5] S. Ghandali, S. Ghandali, and S. Tehranipoor, “Profiled power-analysis attacks by an efficient architectural extension of a CNN implementation”, in *Proc. of 2021 22nd International Symposium on Quality Electronic Design (ISQED)*, 2021, pp. 395–400. DOI: 10.1109/ISQED51717.2021.9424361.

[6] T. Cultice, D. Ionel, and H. Thapliyal, “Smart home sensor anomaly detection using convolutional autoencoder neural network”, in *Proc. of 2020 IEEE International Symposium on Smart Electronic Systems (iSES)(Formerly iNiS)*, 2020, pp. 67–70. DOI: 10.1109/iSES50453.2020.00026.

[7] B. Hettwer, T. Horn, S. Gehrer, and T. Güneysu, “Encoding power traces as images for efficient side-channel analysis”, in *Proc. of 2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2020, pp. 46–56. DOI: 10.1109/HOST45689.2020.9300289.

[8] D. Kwon, S. Hong, and H. Kim, “Optimizing implementations of non-profiled deep learning-based side-channel attacks”, *IEEE Access*, vol. 10, pp. 5957–5967, 2022. DOI: 10.1109/ACCESS.2022.3140446.

[9] M. Carbone *et al.*, “Deep learning to evaluate secure RSA implementations”, *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 2, pp. 132–161. DOI: 10.13154/tches.v2019.i2.132-161.

[10] T. M. Ghazal *et al.*, “Security vulnerabilities, attacks, threats and the proposed countermeasures for the Internet of Things applications”, *Solid State Technology*, vol. 63, no. 1s, 2020.

[11] B. Pandey, V. Bisht, D. M. A. Hussain, M. Jamil, and M. K. Hasan, “Energy-efficient implementation of AES algorithm on 16nm FPGA”, in *Proc. of 2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*, 2021, pp. 740–744. DOI: 10.1109/CSNT51715.2021.9509662.

[12] M. K. Hasan, A. K. M A. Habib, Z. Shukur, F. Ibrahim, S. Islam, and M. A. Razzaque, “Review on cyber-physical and cyber-security system in smart grid: Standards, protocols, constraints, and recommendations”, *Journal of Network and Computer Applications*, vol. 209, art. 103540, 2023. DOI: 10.1016/j.jnca.2022.103540.

[13] T. M. Ghazal, M. K. Hasan, S. N. H. S. Abdullah, K. A. A. Bakar, and H. A. Hamadi, “Private blockchain-based encryption framework using computational intelligence approach”, *Egypt. Inform. J.*, vol. 23, no. 4, pp. 69–75, 2022. DOI: 10.1016/j.eij.2022.06.007.

[14] A. H. A. AL-Jumaili, R. C. Muniyandi, M. K. Hasan, M. J. Singh, and J. K. S. Paw, “Analytical survey on the security framework of cyber-physical systems for smart power system networks”, in *Proc. of 2022 International Conference on Cyber Resilience (ICCR)*, 2022, pp. 1–8. DOI: 10.1109/ICCR56254.2022.9995780.

[15] F. Hu and F. Ni, “Software implementation of AES-128: Side channel attacks based on power traces decomposition”, in *Proc. of 2022 International Conference on Cyber Warfare and Security (ICCWS)*, 2022, pp. 14–21. DOI: 10.1109/ICCWS56285.2022.9998437.

[16] T. Kubota, K. Yoshida, M. Shiozaki, and T. Fujino, “Deep learning side-channel attack against hardware implementations of AES”, *Microprocessors Microsystems*, vol. 87, art. 103383, 2021. DOI: 10.1016/j.micpro.2020.103383.

[17] J.-E. Woo, J. Han, and D.-G. Han, “Deep-learning-based side-channel analysis of block cipher PIPO with bitslice implementation”, *IEEE Access*, vol. 10, pp. 69303–69311, 2022. DOI: 10.1109/ACCESS.2022.3187201.

[18] W. Liu, Y. Zhang, Y. Tang, H. Wang, and Q. Wei, “ALScA: A framework for using auxiliary learning side-channel attacks to model PUFs”, *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 804–817, 2023. DOI: 10.1109/TIFS.2022.3227445.

[19] 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 Systems*, vol. 40, no. 6, pp. 1207–1216, 2021. DOI: 10.1109/TCAD.2020.3033495.

[20] R. Ding, Z. Zhang, X. Zhang, C. Gongye, Y. Fei, and A. A. Ding, “A cross-platform cache timing attack framework via deep learning”, in *Proc. of 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022, pp. 676–681. DOI: 10.23919/DATE54114.2022.9774612.

[21] B. Hettwer, K. Das, S. Leger, S. Gehrer, and T. Güneysu, “Lightweight side-channel protection using dynamic clock randomization”, in *Proc. of 2020 30th International Conference on Field-Programmable Logic and Applications (FPL)*, 2020, pp. 200–207. DOI: 10.1109/FPL50879.2020.00041.

[22] D. Das, S. Gosh, A. Raychowdhury, and S. Sen, “EM/Power side-channel attack: White-box modeling and signature attenuation countermeasures”, *IEEE Design Test*, vol. 38, no. 3, pp. 67–75, 2021. DOI: 10.1109/MDAT.2021.3065189.

[23] P. Upadhya, J. Sangeethapriya, A. Kumar, R. Dhumale, R. Singh, and V. Tripathi, “A critical analysis on the security attacks and relevant countermeasures using ML”, in *Proc. of 2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*,

2022, pp. 96–101. DOI: 10.1109/IC3I56241.2022.10072972.

[24] D. Mouris and N. G. Tsoutsos, “Zilch: A framework for deploying transparent Zero-Knowledge Proofs”, *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3269–3284, 2021. DOI: 10.1109/TIFS.2021.3074869.

[25] H. Yu, H. Shan, M. Panoff, and Y. Jin, “Cross-device profiled side-channel attacks using meta-transfer learning”, in *Proc. of 2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 703–708. DOI: 10.1109/DAC18074.2021.9586100.