International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 16, No.1 (Jul-24)

http://dx.doi.org/10.12785/ijcds/160127

Design of Time-Delay Convolutional Neural Networks (TDCNN) Model for Feature Extraction for Side-Channel Attacks

Amjed Abbas Ahmed[1,2], Mohammad Kamrul Hasan[1], Shahrul Azman Mohd Noah[1] and Azana Hafizah Aman[1]

[1]*Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi 43600, Malaysia*
[2]*Department of Computer Techniques Engineering, Imam Al-Kadhum College (IKC), Baghdad 10011, Iraq*

Received 5 Feb. 2024, Revised 14 Apr. 2024, Accepted 20 Apr. 2024, Published 1 Jul. 2024

Abstract: This work explores a novel method of SCA profiling to address compatibility problems and strengthen Deep Learning (DL) models. Convolutional Neural Networks are proposed in this research as a countermeasure to misalignment-focused countermeasures. "Time-Delay Convolutional Neural Networks" (TDCNN) is more accurate than "Convolutional Neural Network," yet it's still acceptable. It's true that TDCNNs are neural networks based on convolution learned on single spatial information, just as side-channel tracings. However, given to recent surge in popularity of CNNs, particularly from the year 2012 when CNN framework ("AlexNet") achieved Image Net Large Scale Visual Recognition Competition which is a notable image detection competition, a novel TDCNN has been termed out in DL literature. Currently, it needs to employ the characteristics related to CNN design, including declaring that one input feature equals 1 for instance, to establish a TDCNN in the most widely used DL libraries.

Keywords: Deep Learning, Convolutional neural networks, Side channel Analysis, Side channel attacks, cryptography.

1. INTRODUCTION

We focus on deep learning (DL) methods, and convolutional neural networks (CNNs) in particular, in the context of cryptographic implementations that are guarded by countermeasures that try to make side-channel acquisitions more misaligned or desynchronized. The second group of safeguards could be software-based (such as the introduction of random delays via dummy operations) or hardware-based (such as unpredictable clocks or non-deterministic CPUs) [1], [2].

As in the leaking model suggested in [3], desynchronization could be seen as an acquisition noise component. In any case, it amplifies the background noise that conceals private data in the traces. If the assault method, in terms of exploiting statistical tools, stays the same, raising the total amount of acquires by a measure that is substantially proportional in the misalignment impact, as stated in [4], could be sufficient to render the attack equally successful as in a synchronised situation. This is a theoretically satisfying answer from a statistical perspective to such a noise rise. Such an augmentation cannot be feasible in real life due to several factors. To start, the acquisition campaign could be time-or memory-bound for the attacker or evaluator. Secondly, the compromised system can block an infinite number of executions by implementing a security measure. Thirdly, rather than increasing linearly with the quantity of data to be analysed, the level of complexity of attack techniques can develop in a cubic pattern with an increasing amount of traces. The KDA analysis for non-linear extraction of features is one instance of this.

To address misaligned trace sets, the second method suggested in SCA literature is to do realignment preprocessing before to the assault. It is possible to classify realignment approaches into two groups: one focused on signal processing (see, for example, [5]) and better suited to hardware countermeasures; and another, more probabilistic in nature (see, for example, [6]), designed for detecting fake operations (software countermeasures).

We discovered that Convolutional Neural Networks provide the potential for end-to-end profiling attacks, where sensitive information can be directly extracted from raw data without the need for any preprocessing. We are of the opinion that dimensionality reduction approaches and realignments both carry the danger of erasing valuable information from data. Indeed, in order to get a well-

E-mail address: amjedabbas@alkadhum-col.edu.iq, mkhasan@ukm.edu.my, shahrul@ukm.edu.my, azana@ukm.edu.my

synchronized dataset, a realignment method modifies signals so that traces are somewhat comparable to one another. One the one hand, determining if the preprocessing was satisfactory by assessing the precision of a realignment is not an easy task. A realignment's purpose, however, is not to glean discriminatory and sensitive information from traces. Attempts to realign the trace set can potentially remove important information, even if we could prove that a resynchronization is flawless using certain criteria. The excellent scalability of CNNs and DL technologies in general to "big-data" settings is making them stand out nowadays. Their ability to readily take use of computing resources, such as graphics processing units (GPUs) or the so-called Tensor Processing Units (TPUs) designed specifically for neural networks (NNs), enables computational accelerations, which is one of their strengths. With more data at hand, ML models can learn more complicated issues with less risk of overfitting, and larger capacities mean more data to work with. The ever-increasing data availability and NNs' scalability, according to this opinion, are the major reasons for their recent success. Data Augmentation (DA) is a strategy in machine learning literature that can help large capacity NNs avoid overfitting and perform effectively, even in situations where there can be little data, such as side-channel settings with limited acquisitions.

A. *Objectives*

The objectives of our proposed research work are as follows,

1) Using the time leaking patterns identified in side-channel data and accurately modeling them are the major objectives. In order for TDCNNs to capture temporal linkages, neural layers are enhanced with time delays.
2) Due to their ability to quickly learn vital features, TDCNNs are able to detect subtle leakage patterns that can go unnoticed by more conventional research approaches.
3) The objective is to increase attack efficiency and accuracy by exploiting both geographical and temporal connections in side-channel data. Cryptographic keys and other sensitive information can be located more quickly and precisely manner.

B. *Contributions*

Our major contributions in this research work are listed below,

1) It has been observed that the improvement in SCA detection performance by introducing delays in the convolutional layers in TDCNNs
2) Instead of requiring human intervention for feature developing, TDCNNs autonomously train discriminative features from raw side-channel traces.
3) TDCNNs are more resilient in real-world situations because they can learn to differentiate between real leakage patterns and noise.

2. Literature review

Many studies have focused on detecting Side-Channel Attacks (SCAs) using CNNs. The works of some notable authors from that year are listed below:

Using deep learning and CNNs, Jin et al. [5] investigated side-channel analysis. This research looked into how well CNN architectures could mine side-channel leakage traces for useful characteristics. Findings from this study showed that CNNs are a promising tool for automating feature extraction with the goal of enhancing side-channel attack awareness. The authors demonstrated that CNNs can learn discriminative features from raw side-channel data, which could mean that human feature engineers won't be needed to identify side-channel attacks.

Panoff et al. [6] presented a great deal of their research to study CNNs and other deep learning approaches for side-channel analysis. For the purpose of practicing side-channel attacks, a variety of network setups and training methodologies were used. Experiment design and model tweaking were crucial for comprehending CNNs and other deep learning methods for side-channel studies. This part of the procedure was crucial. The authors compared CNNs to other deep learning methods in order to identify the parameters that impact side-channel attack detection using deep learning.

The encryption solution that Ahmed and his co-authors [7] proposed uses deep learning and convolutional neural networks to prevent side-channel attacks. Using deep neural networks to automate feature extraction allowed them to improve profiled side-channel attacks. This study claims that CNNs were used to detect side-channel attacks by automatically extracting data from side-channel leakage traces. Profiled attacks can become more successful and scalable if this happens. The authors' study proved that side-channel analysis and deep learning can automate feature extraction, which means domain-specific expertise and manually constructed features won't be needed to identify side-channel attacks. The need for designs customised to certain domains was met by Zhang et al. [8] with their convolutional neural network (CNN) architecture for power side-channel attacks. Several benefits are associated with this approach. The study's overarching goal is to provide light on key recovery mechanisms by means of power consumption traces. Both precision and productivity are enhanced in the end by this.

The design cannot be applicable to other forms of side-channel leakage, such electromagnetic emissions, since its main emphasis is power side-channel attacks. We can learn more about the framework's pros and cons by comparing it to other methods that have been tested and found to function. By presenting domain adaptation approaches that are exclusive to CNNs, this study by Palisse et al. [9] addresses the issue of domain shift in side-channel attack detection. This is a good thing about the research. Improved

CNN model transferability from synthetic to real-world data could lead to enhanced CNN model robustness and generalizability, as shown in the aforementioned methodologies. There is a chance that computational complexity and scalability will both rise when CNN systems include domain adaptation mechanisms. Given the scale and distribution of the dataset, it is probable that more testing in real-world settings is necessary for domain adaptation methodologies.

Ou et al. [10] provide efficient methods for enhancing data to make models more generalizable and resilient.

Additionally, these techniques address the issue of insufficiently tagged data in detecting side-channel attacks. Convolutional neural network (CNN) models trained for side-channel analysis can benefit from a more robust and efficient training set. The availability and quality of training datasets could limit the efficiency of data augmentation methodologies, which in turn limits their applicability to different circumstances. In order to correctly analyze the work's performance and efficacy, it could be helpful to do thorough dataset evaluations and compare it with alternative augmentation techniques.

Hettiarachchi et al. [11] state that deep learning approaches, which take into account both the topography and the side channels, can help build robust defenses against side-channel attacks. The findings could be helpful in improving security measures against side-channel attacks by tackling the increasing risks and suggesting appropriate response tactics. Researching certain methodology or strategies in detail can be challenging since the study includes such a wide variety of subjects related to side-channel attacks and countermeasures. Additional experimental validation and analysis can be necessary to assess the efficacy and scalability of the suggested attack and defensive tactics to other situations and datasets.

3. Methodology

In this case, the NNs value the results of the categorization task. Here, Z's components, which stand for classes set, were represented using one-hot encrypting technique, and, as we recall from groping problem, learning algorithm are expected in producing $F : RD \rightarrow 0, 1|Z|$ function. Since "Z" is function's categorical output, it is a continuous finite set. One variation of the categorization job is to find an expression $F : RD \rightarrow [0, 1]|Z|$ that establishes a dispersal of probabilities across groups. Our favoured formulation is the last one since it makes it easier to execute both simple and sophisticated attacks by giving us easy access to the categorization solution.Developing discriminative models, which seek to directly simulate the subsequent contingent likelihood of labelling given the observed trace, represented as F, is an efficient application of NNs in this context. TAs are based on the construction of dynamic models, or approximate the templates. When a label is applied, these models ought to match the conditioned likelihoods of the trace.

With NNs, a large number of smaller functions, or layers, are combined to generate the function F. The identity is stored over the source datum x in the layer of inputs, which is the main layer of a neural network. All levels between are hidden layers, and the output layer is the last layer. The output of F is an array y of numbers for the $|Z|$ labels, with a size of $|Z|$. These vector can be a common approximating function for distribution of probabilities distribution. For us, this tends to work. Quantity and dimensions of tiers that comprise the network are particularly referred to as the NN's architecture. All of the characteristics that specify a building's design and a few additional ones regulate the training process. The network's computational components, dubbed neurons after its namesake, generate the product of scalars among the input dimensions and the vector of weights that require training, or trainable weights. In the layer below, new input vectors are created based on the evaluation outcomes of neurons in that layer. As we'll see, operating linearly — that were products of scalar which neurons processes — often establish the weights that are trainable of a neural network. Neural networks can be programmed to operate in simultaneously and GPUs are reasonably efficient at processing and differentiating between them.

Multi-Layer Perceptrons (MLPs), sometimes referred to as Feedforward Neural Networks, are one type of NN design. A function, such as F, which has both non-linear and linear components, is connected to MLPs. the activations. The word "feedforward" describes a model in which data goes straight from the input to the output without any kind of feedback loop, meaning that the model's outputs do not feed back into itself. That goes against the very principles of what are known as Recurrent Neural Network architectures. A generalisation of the MLPs is the CNNs.

Here is one way to represent a typical MLP that is focused on classification:

$$F(\vec{x}) = s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(\vec{x}) = \vec{y} \qquad (1)$$

Where Most of the time, the λ_i functions are affine functions that can be represented as Fully-Connected (FC) layers. For instance, an FC that receives an input of $x \in RD$ will produce an output of Axe + b, where b is a bias vectors $\in RC$and A is the weight matrix $\in RD \times C$. The biases and weights are trainable of FC substrates. Because $A[i, j]$ weight ties every i-th feeding coordinates to every j-th outcome, they are known as fully-connected. It is possible to think of FC layers as a specific example of the linear layers found in generic network designs because not all interconnections are present in them. Setting the (i, j)-th coordinates of Matrix A to 0 will formalise it as containing no (i, j)-th linkages.

The σ_i represents what are known as activation functions (ACT). A realistic non-linear functional that is applied independently to each and every input coordinate is called an activation function. Frequently, using trainable weights

is not necessary. Since they have functions similar to the logical sigmoid, which is likewise represented by σ, we usually reference to them as σ. In actuality, actual measure, restricted, differentiable functions, monotonic consisting of positive primary derivative are typically suggested for sigmoidal functions. Rectified linear unit (ReLU), regrettably, $\mathrm{ReLU}(x)[i] = \max(0, x[i])$, which was first introduced by [7], is an equation that is most frequently cited in current neural network research. Even though such is non sigmoidal—definitely, which are unbounded nor differentiable-it is nevertheless piecewise linear, which allows for the maintenance of most characteristics which generates linear prototypes less difficult to maximise using gradient-dependent technique.

$$s\left(\vec{x}\right)[i] = \frac{e^{\vec{x}[i]}}{\sum j^{e^{\vec{x}[i]}}} \tag{2}$$

s is shorthand for softmax1 function.

The softmax function is the final layer used by most neural network classifiers. Because of this, model F can be viewed as an improved extended version of the binaries classifier, with the softmax acting as the sigmoid's replacement in the multi-class model and all of the earlier layers of F acting as linear arguments. The previous layers are responsible for handling any preprocessing and forecasting the unnormalized log probabilities. Normalising the output scores to fit the distribution of probabilities $F(x) \approx pZ|X = x$ is the goal of the softmax.

4. Learning Algorithm

A neural network's weights are adjusted in the training phase. They start with an arbitrary collection of values. Subsequently, they are updated by an iterative process that eliminates a loss function that gauges functions $F(X)$'s classification error for set of training. An (Stochastic) gradient descent method is used locally in this method [8].

A. Training

When all of the training data is processed before making any changes to the weights, it was observed that the NN was trained using complete batch learning. The inverse is true when dealing with stochastic approaches, which only handle one training input at a time. In reality, a middle ground strategy known as mini-batch learning is generally used. This method involves learning with tiny batches, or groupings of training inputs, at a time.

Epochs are iterations of the Stochastic Gradient Descent that cover the whole training dataset. Hyperparameters also include the number of epochs. It stands to reason that underfitting can result from using a small number of epochs, whereas overfitting could occur from using a large number of epochs. It can be choosen to use the so-called early halting technique [10] in our studies to sidestep the need to tune the number of epochs in advance. The first step is to decide on a stop criteria that will be used throughout the training. Typically, the decision is made when the validation accuracies or losses remain stable or deteriorate over time.

B. Entropy Cross Metric

This is frequently used as default, is one conventional technique for generating the loss functions of a classification-oriented neural network [10]. It can be optimised using conventional gradient-based approaches as it is smooth and decomposable. Here, zi and yi are the probability mass functions that describe two distributions; the cross-entropy between them provides a measure of how different they are.

$$\begin{aligned}\mathbb{H}\left(\vec{x}_i, \vec{Y}_i\right) &= \mathbb{H}\left(\vec{z}_i\right) + D_{KL}\left(\vec{z}_i \| \vec{Y}_i\right) = \mathbb{E}_{\vec{z}_i}\left[-\log\vec{y}_i\right] \\ &= -\sum_{t=1}^{|\mathcal{Z}|} \vec{z}_i\,[t] \log\vec{y}_i\;\;[t]\end{aligned} \tag{3}$$

where the entropy is represented by $\mathbb{H}$ and the Kullback-Leibler divergence is denoted by D_{KL} [11]. Therefore, This idea is similar to the negative log-likelihood expression in 3 and comes from the field of information theory. The loss function 3 remains a cross-entropy amounted on entire sets of traces in that batches. There are two ways to look at minimising it: either decreasing dissimilarity among estimation of network, what distribution is and required by us for approximation, or optimising a-posterior probabilities of the accurate label. It has chosen to use the loss functional 2 for our research. Conversely, alternative metrics can be investigated in order to yield more advantageous results [12].

For the experiments listed below, in two subsets side-channel profiling collection will be divided by us: a set to be trained and a validation set. The NN's parameters will be updated by sequential processing of the set of training data. It is typical practice to use the validation set at the conclusion of each epoch to keep an eye on training, and more specifically to detect when overfitting is about to happen. Surprisingly, it has not been used cross-validation to enhance the precision of our observation. Alternatively, it has been assessed that the trained model's and the resulting attack strategy's performance on the classification task using a side-channel attack set.

C. Attack Strategy with an MLP

This method of SCA with an MLP apart from the conventional Pattern Approach. A generative model underpins TA, whereas a discriminative one is built using MLPs; this is the key distinction. It is true that TA uses prior estimations for the templates, while MLPs directly estimate the posterior probability $F(x) \approx pZ|X = x$. The next step in the attack plan for both techniques is identical when this approximation is completed. By acquiring pairs (x_i, e_i) $i = 1, \ldots, N_a$, New attack traces are discovered by the attacker, which he can only associate with the public variable E. Afterwards, he establishes important hypotheses $k \in K$ and, assuming that every acquisition represents a separate observation of X, he assigns a score d_k to each

hypothesis $k \in K$. This score is rewritten in terms of the MLP model as:

$$d_k = \prod_{i=1}^{N_a} F(\vec{x}) [f(k, e_i)] \quad (4)$$

In the end, the optimal option for the crucial role $\hat{k}$ is the one that maximises the joint probability.

$$\hat{k} = \underset{k}{\operatorname{argmax}}\, d_k \quad (5)$$

5. Performance Estimation

A. Maximal Accuracies and Confusion Matrix

One of the majority popular metrics used to monitor and assess a classification-oriented NN is its accuracy. When applied to a dataset, accuracy is the percentage of correct classifications. A model's accuracy can be defined as the percentage of correct classifications made over all training, test sets and validation respectively. After every conclusion session, estimate and compare the accuracy of the validation and training; this is useful. We reasoned that adding these two more parameters to our research would be intriguing.

- The greatest training accuracy, which is determined at the conclusion of each epoch by adding together all training accuracies;
- The maximum validation accuracy, which is the total of all validation accuracy calculations made at the conclusion of each period.

Along with the two metrics mentioned above, we will additionally compute a test accuracy to assess how well our trained model performed. It can occasionally be useful to look at the referred to as confusion matrix in order to complete this assessment. In fact, the latter matrix makes it possible to identify the confused classes in the process of misclassification. It is possible to infer the distribution of the pairs (real label, predicted label) from the test set's classification results, and this distribution gives rise to the confusion matrix. A confusion matrix with diagonal entries represents a test accuracy of 100%.

B. Side-Channel-Oriented Metrics

Although it is directly related in comparison to the side-channel language achievement rate of a Modest Attack, the accuracy measure is well-suited to the machine learning classification challenge. The accuracy measure is insufficient to assess the attack performance when the attacker can get many traces with different plaintexts. While a SCA considers all labels, this statistic just considers the one associated with the highest score. This observation is taken into consideration by consistently linking the test accuracy to a side-channel metric. The fewest possible N about side-channel traces required to guarantee that the predicting entropy is always identical to one is the definition of this measure. Using ten separate assaults, we will attempt to determine such a guessing entropy in our trials.

6. Convolutional Neural Networks

By adding a pooling layer and a referred to as the convolutional layer, which uses convolutional filtering, Convolutional Neural Networks (CNNs) strengthen the conventional Multi-Layer Perceptron (MLP) model against misalignment. We will go into detail about these two specific levels later on.

The linear layers known as convolutional layers (CONV) share their weights in all directions. Since CNNs were initially developed for images, Figure 1 presents an alternative representation from the most common one, which organises tier interactions resembling 3D-patterns (depth, height and weight) [13]. Similar to side-channel traces, Figure 1 depicts a 2D convolutional neural network, or CNN, tuned for 1D data. Convolutional filters, which are small matrices with an area of $W \times V$ (where W is the kernel size), can be moved down the segments measurement of input in 1D data. Through fixed number strides, to apply CONV on an input with initial width V of 1 and size D $\times$ V. Filters through window (termed a patches in machine learning) that creates new matrix of size $1 \times n$ filter from $W \times V$ successive data points. The way these matrices are arranged indicates that the n filter represents the layer output depth. Several parameters, including as the input dimension, stride, and padding, influence the output length metric of a convolutional layer. The two most widely used techniques for input padding are "valid padding" and "same padding."The former ensures that with a stride of 1, outcome having same duration as inputs; concerning 2 stride, duration of input remains precisely half; for a stride of 3, it is exactly split by 3; and so on. This is achieved by preceding and following the input with a string of zeros. The best way to handle padding is to avoid using any kind of cushioning at all. Since only information that is valid readings were utilized like inputs, resultant duration is aligned accordingly, output duration is often equal to $D - W + 1$ when the stride length is set to 1. Among the model's trainable weights are the window's coordinates. The input is multiplied by various portions of the datum as they slide over it, but they are bounded to remain unaltered while sliding, meaning they must act consistently regardless of where entries at input are on globalised input data. The purpose of such restriction is in ensuring that CONV substrate can acquire data features that are unaffected by changes in location, sometimes known as shift-invariant features. CNNs were developed because to the prevalence of shift-invariant features in image recognition environment. A person's eyes, nose, and mouth, for instance, are distinguishing characteristics of that person regardless of where they are in the photograph. Convolutional neural networks (CNNs) are resistant to geometric distortions [14] and temporal distortion when taking side-channel data into account because of their capacity to find shift-invariant features. This is why they can successfully countermeasures based on misalignment.

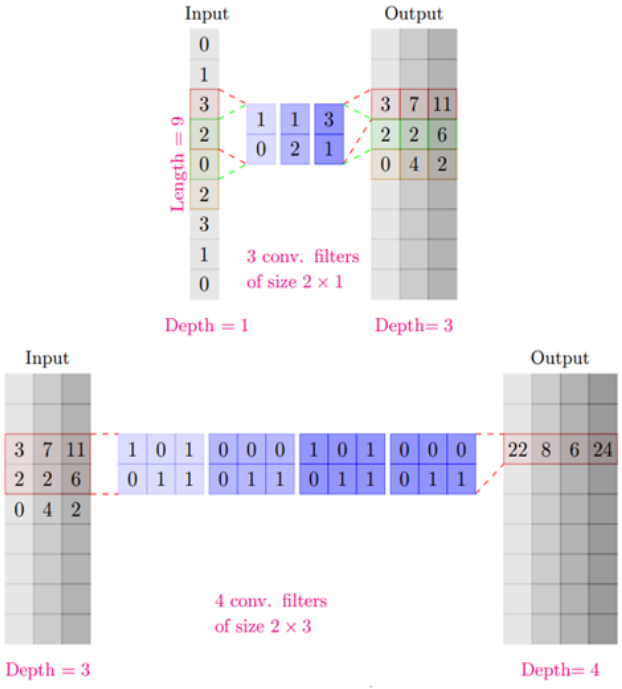

Figure 1. Layers of Convolution. Bottom: V=3, W=2, nfilter=4. Top: V=1, W=2, nfilter=3, stride=1

A. *Layers for pooling (POOL)*

The most popular example of a convolutional layer is an array having stride equivalent to 1, identical padding; size of output is identical to its input size multiplied by n filter. As more data is added throughout the tiers, the complexity exponentially grows when several of these convolutional layers are layered. The installation of pooling (POOL) layers is suggested as a means to prevent this complexity growth. Non-linear POOL layers shrink the spatial dimension (see to Figure 2). As CONV layers, their function is to filter the incoming data as it comes in. The filter has a length of W and is one-dimensional. The stride is usually chosen to match the length; in Figure 2, for example, both stride and length are set to 3, guaranteeing chosen input segments is non overlapping. Layers of Convolution have trainable weights, whereas pooling filters do not. Rather, they just select a segment by swiping over the input, and then they implement a pooling function. Max-pooling and average-pooling are two common pooling functions. Whereas the latter delivers the median of the segment's coordinates, the former returns the greatest values throughout the segment.

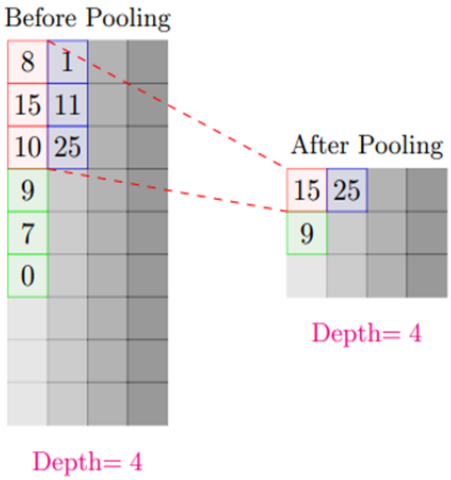

Figure 2. Max-pooling layer: W = stride = 3

B. *Common Architecture*

A CNN is constructed with an ACT layer and a CONV layer as its main building pieces. The first one uses filters to extract data from the input at a local level, while the second one uses non-linearity to boost the model's capacity. It is common practice to add a POOL layer δ after a few $(\sigma \circ \gamma)$ blocks in order to decrease the total number of neurons: $\delta \circ [\sigma \circ \gamma]n2$. The network iteratively generates this new block until it reaches a sizeable output.

The next step is to include some FCs so that we can get a global output that is dependent on the whole input and not just local characteristics. This formula summarises the most prevalent features of convolutional networks:

$$s \circ [\lambda]^{n_1} \circ [\delta \circ [\sigma \circ \gamma]^{n_2}]^{n_3} \tag{6}$$

Layer by layer, the network employs convolution filters to boost spatial depth, activation functions to introduce non-linearity, and pooling layers to reduce the size of objects that are spatial (or temporal, if side-channel traces are included). When narrow and deep showcasing gets structured, more or one layers of FC gets attached to it through application function of softmax. A CNN design in operation are shown in Figure 3.

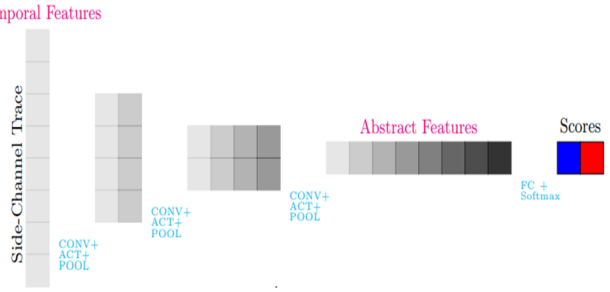

Figure 3. Proposed CNN architecture

C. Datasets

The use of cryptographic device traces in training TD-CNNs to detect side-channel attacks is widespread. These remnants can point to power usage, electromagnetic interference, or other side-channel data lost from the device during cryptographic calculations. Testing was conducted using the DPA Contest v3. Standard datasets for side-channel inquiry are those that are part of the DPA Contest v3. Several distinct cryptography systems are the sources of the power usage traces. The DPA Contest v3 datasets include a large number of traces for both plaintexts and key values. Various implementations, including AES, DES, and RSA [20], use these traces. The inclusion of power consumption indications from the cryptographic devices being assessed is the most distinctive aspect of DPAContest datasets. Cryptographic operations, including encryption and decoding, leave traces that show how much power is being utilized. Oscilloscopes and other measurement instruments can get traces as discrete samples.

7. Results and discussions

A conventional mechanical countermeasure against side-channel assaults is to introduce clock instability. The result is a misalignment of the enemy's traces and a deforming effect that builds up and impacts every gained clock cycle. Actually, different numbers of time samples are used to sample clock cycles throughout the assault since their durations are not uniform. Since there is a certain clock cycle to align with, it is not enough to just translate the acquisitions. Such deformations can be managed with the use of certain realignment methods, such as [15]. Here, we want to demonstrate that the realignment pre-processing can be eliminated, allowing the CNN deep structure to implicitly handle it.

A. Performances over Artificial Augmented Clock Jitter

The results obtained for double datasets, DS_high_jitter and DS_low_jitter, are shown here. Ten thousand tagged traces and one hundred thousand attack traces are utilised for training (more precisely, nine thousand for training and one thousand for validation) in each. One thousand eight hundred sixty time samples make up the traces. The two sets of data were created by intentionally interpolating a jitter effect over a subset of the original, synchronised traces. The same Atmega328P CPU was used for measuring the original traces. We discovered that 34 instructions had leaks at the beginning, including two nops, sixteen NVM loads and sixteen look-up table accesses. Our experimentation of attack are assuming that 19th clock cycle-the first look-up table access-is the intended target. $Z = HW(Sbox(P/K))$ is the name of the critical variable. We were able to reproduce the jitter effect using the DS low_jitter dataset (sigma = 4 and B = 2) and DS high_jitter dataset (B = 4 and sigma = 6). Desynchronization increases with time, thus we can observe the cumulative effect of the jitter; Figure 2 displays few tracings of DS high jitter and DS low jitter. Few traces regarding both DS high and DS low jitter are displayed in Figure 2. Without performing any point-of-interest (PoI) selection on each dataset, we entered the entire collection of traces into our CNN.

It's been helped with a severe overfitting problem once again, and this time it was able to lessen it by using the newly-introduced DA method. This research work trains the CNN model using nine permutations of the shifting deformation SHT and the add-remove deformation ARR, with $T = 200$ and $T \in \{0, 20, 40\}$ and $R \in \{0, 100, 200\}$, respectively. To further demonstrate that the DA's advantages are limited, it has been conducted a second experiment using $SH_{200}AR_{500}$, which has much higher DA values. The results were poorer than those with, say, $SH_{40}AR_{200}$, since, as predicted, excessive deformation negatively impacts CNN performances.

Table I summarises findings that were obtained during our experimentation. Case SH0AR0 is a reference that has experienced overfitting as it relates to a training that did not use the DA approach. Accuracy of validation enhances and accuracy of training downfalls as DA parameters are increased. This demonstrates through trial how well the DA approach reduces overfitting. Take the $SH_{100}AR_{40}$ DS low jitter dataset scenario as an example. There are times when highest accuracy of validation is more compared to best precision of training. Figure 4 shows the accuracies acquired in this scenario, epoch by epoch, for both training and validation. It is clear that the peculiar relationship between training and validation accuracies is maintained practically epoch after epoch, and it is not limited to the maximum values. It is not the result of some unfortunate erroneous estimates, as can be shown in the figure, since this occurrence happens at several epochs. In order to make sense of this occurrence, as it can be seen that the training collection also contains material that has been made enhanced—that is, corrupted through DA—whereas sets of verification set solely consist of non-augmented information. CNN can have been effective at extracting the relevant features for the original data categorization, but it cannot have learned how to categorise the enriched data, as evidenced by the observation that its validation efficiency is higher than the acquired training accuracy. Our observations about the DA approaches indicate that they perform well when used singly and much better when combined.

Based on our findings in Table I, the model generated using the $SH_{200}AR_{40}$ approach was selected concerning DS_low_jitter information set, thus model generated by deploying $SH_{200}AR_{20}$ technique was selected for the DS_higher_jitter dataset. Their findings are contrasted with those of a realignment technique and a Gaussian TA in Figure 4. In our experiment, when the training curve is smooth while the validation curve fluctuates drastically, is due to the overfitting problem. Overfitting refers to the process whereby the model learns how to do well on the training data only but fails in generalizing in future unseen examples.

TABLE I. The output of our CNN when subjected to several DA methods with an artificially-generated jitter countermeasure

DS_low_jitter								
a b / c d	SH_0		SH_{20}		SH_{40}		SH_{200}	
AR_0	100.0% 57.4%	68.7% 14	99.8% 82.5%	86.1% 6	98.9% 83.6%	84.1% 6		
AR_{100}	87.7% 86.0%	88.2% 6	82.4% 87.0%	88.4% 5	81.9% 87.5%	89.6% 6		
AR_{200}	83.2% 88.6%	88.6% 6	81.4% 85.7%	86.9% 6	80.6% 87.7%	88.9% 5		
AR_{500}							85.0% 86.2%	88.6% 5
DS_high_jitter								
a b / c d	SH_0		SH_{20}		SH_{40}		SH_{200}	
AR_0	100.0% 40.6%	45.0% 35	100.0% 51.1%	60.0% 9	98.5% 62.4%	67.6% 11		
AR_{100}	90.4% 50.2%	57.3% 15	76.6% 72.4%	73.6% 11	78.5% 73.5%	76.4% 9		
AR_{200}	83.1% 64.0%	67.7% 11	82.0% 75.5%	77.1% 8	82.6% 74.4%	77.0% 8		
AR_{500}							83.6% 68.2%	73.4% 11

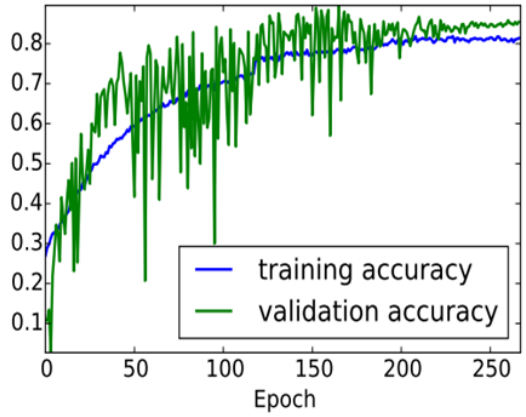

Figure 4. CNN model training using DA $SH_{100}AR_{40}$. As the difficulty of the training classification issue rises relative to that of the actual classification problem, the validation accuracy remains consistently greater

In most cases this leads to a situation in which the model's performance continue to have a smooth training curve while it's performance on validation data decrease or undergoes fluctuations. Numerous tests of cutting-edge Gaussian TA have improved this comparison. This research work employs a simple realignment strategy, whereby, firstly identify the peaks that exceed a threshold and then keep all the data within a window surrounding these peaks. This is because, with experiment, leakage accumulates in peaks which were easy visible because of their comparatively large amplitude. Then, two methods were used to choose the points of interest (PoIs): first, it was picked 3–20 points that maximised the estimated instantaneous signal-to-noise ratio (SNR), and second, it was picked sliding windows of 3–20 points that covered the PoI area in sequential order. While processing the templates, (1) it was tested the traditional method [16], which estimates a class-specific mean and covariance matrix, (2) implemented the pooled covariance matrix technique [17], and [18] tested the stochastic method [19], [20]. It was used a stochastic technique across windows of varying sizes, and the best results are shown in Figure 5. In comparison to the Gaussian templates, the CNN method achieves much better results, regardless of whether realignment is used or not. This verifies that the CNN method is resilient to the jitter effect: the realignment that is built into the training phase and the selection of points of interest work.

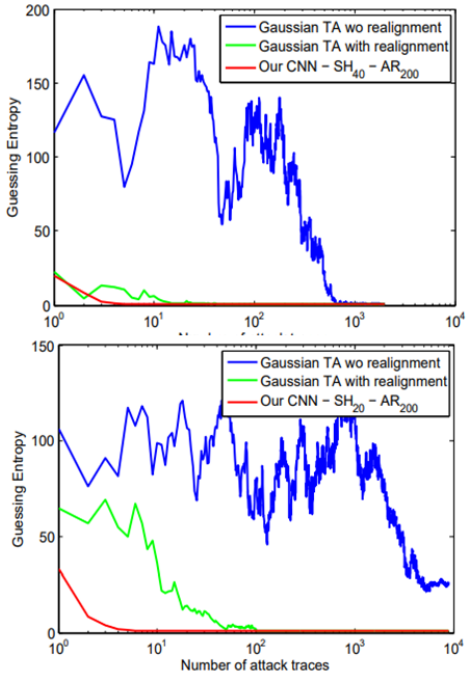

Figure 5. The comparison shows the CNN approach and a Gaussian template attack throughout DS low jitter and DS high jitter, correspondingly, with and without realignment

B. *Significant Findings*

Following are the significant outcomes observed during the experiment.

1) TDCNNs' ability to automatically learn discriminative features from side-channel traces, feature engineering is no longer necessary for their detection of side-channel attacks. This has led to considerable breakthroughs in the field. This work has simplified the detection of side-channel attacks without requiring domain-specific traits or specialist knowledge.
2) Regardless of device heterogeneity, environmental variables, and countermeasures, the side-channel leakage resistance of TDCNNs has been shown. Their ability to identify side-channel attacks on various cryptographic systems is due to their ability to adapt to different leakage patterns and noise levels.
3) In the presence of complex leakage patterns, TDCNNs outperform conventional side-channel attack detection approaches. Their exceptional performance is a result of their remarkable comprehension of complex patterns and relationships in side-channel traces.
4) Successful testing of TDCNNs in discovering cryptographic equipment vulnerabilities can be conducted using a variety of real-world side-channel attack scenarios.

C. *Limitations of our Study*

When it comes to detecting side-channel attacks, TDCNNs have a few drawbacks despite their many advantages:

1) It is conceivable that TDCNNs will fail when the leakage patterns are significantly non-linear or have long-term dependencies outside of the network's receptive area.
2) Because of the enormous size of datasets and complicated topologies involved, training TDCNN is computationally costly and takes a long time. This could cause problems in practical production settings when time or resources are limited.

8. Conclusions

In this section, it has been successfully outlined scratch to top profile attack approach that depends over CNN. Here, it has been demonstrated regarding our hypothesis—that the proposed models will continue to be efficient though in midst of misaligned traces was valid by executing CNN-based attacks on several kinds of misaligned data. Unlike state-of-the-art Template Attacks, which need painstaking trace realignment for optimal performance, this characteristic offers a significant practical benefit. In comparison to standard TA, our CNN-based approach varies in two key respects. To begin with, rather of a generative model, it employs a discriminative one. Second, it incorporates all of the preprocessing steps that will be required for a TA to be effective into a single training phase. In fact, it not only solves the problem of selecting PoIs, but it also eliminates the need for trace realignment, which is irrelevant to the CNN approach: The attacker can easily choose big windows since CNNs handle high-dimensional data effectively. Our CNN can collect data with nearly no instantaneous signal-to-noise ratio from a variety of sources because the results are very indicative of the real world. Using a CNN with a bit of complicated architecture, it was able to address the traces misalignment and successfully pinpoint the overfitting problems. To tackle this fundamental machine learning issue, it has been introduced two Data Augmentation techniques designed for misaligned side-channel traces. Our testing findings consistently demonstrate their valuable contribution to the CNN approach. In this work, it was presented the attacks that target implementations that do not use masks.

References

[1] C. Jin and Y. Zhou, "Enhancing non-profiled side-channel attacks by time-frequency analysis," *Cybersecurity*, vol. 6, no. 1, p. 15, 2023.

[2] K. Pu, H. Dang, F. Kong, J. Zhang, and W. Wang, "A quantitative analysis of non-profiled side-channel attacks based on attention mechanism," *Electronics*, vol. 12, no. 15, p. 3279, 2023.

[3] K. Nagarajan, R. Roy, R. O. Topaloglu, S. Kannan, and S. Ghosh, "Scann: Side channel analysis of spiking neural networks," *Cryptography*, vol. 7, no. 2, p. 17, 2023.

[4] A. A. Ahmed, M. K. Hasan, N. S. Nafi, A. H. Aman, S. Islam, and M. S. Nahi, "Optimization technique for deep learning methodology

on power side channel attacks," in *2023 33rd International Telecommunication Networks and Applications Conference*. Melbourne, Australia: IEEE, 2023, pp. 80–83.

[5] S. Jin, S. Kim, H. Kim, and S. Hong, "Recent advances in deep learning-based side-channel analysis," *Etri Journal*, vol. 42, no. 2, pp. 292–304, 2020.

[6] M. Panoff, H. Yu, H. Shan, and Y. Jin, "A review and comparison of ai-enhanced side channel analysis," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 18, no. 3, pp. 1–20, 2022.

[7] A. A. Ahmed, M. K. Hasan, N. S. Nafi, A. H. Aman, S. Islam, and S. A. Fadhil, "Design of lightweight cryptography based deep learning model for side channel attacks," in *2023 33rd International Telecommunication Networks and Applications Conference*. Melbourne, Australia: IEEE, 2023, pp. 325–328.

[8] 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.

[9] A. Palisse, H. Le Bouder, J.-L. Lanet, C. Le Guernic, and A. Legay, "Ransomware and the legacy crypto api," in *Risks and Security of Internet and Systems: 11th International Conference, CRiSIS 2016*. Roscoff, France: Springer, 2017, pp. 11–28.

[10] Y. Ou and L. Li, "Side-channel analysis attacks based on deep learning network," *Frontiers of Computer Science*, vol. 16, no. 2, p. 162303, 2022.

[11] M. Hettiarachchi, H. Sandanuwan, R. Balasooriya, R. Hettiarachchi, K. Abeywardena, and K. Yapa, "Time analysis side channeling attack in symmetric key cryptography," in *2023 8th International Conference on Information Technology Research (ICITR)*. Colombo, Sri Lanka: IEEE, 2023, pp. 1–5.

[12] F. Kong, X. Wang, K. Pu, J. Zhang, and H. Dang, "A practical non-profiled deep-learning-based power analysis with hybrid-supervised neural networks," *Electronics*, vol. 12, no. 15, p. 3361, 2023.

[13] S. Hajra, M. Alam, S. Saha, S. Picek, and D. Mukhopadhyay, "On the instability of softmax attention-based deep learning models in side-channel analysis," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 514 – 528, 2023.

[14] M. Krček, H. Li, S. Paguada, U. Rioja, L. Wu, G. Perin, and Ł. Chmielewski, "Deep learning on side-channel analysis," in *Security and Artificial Intelligence: A Crossdisciplinary Approach*. Cham: Springer, 2022, pp. 48–71.

[15] F. Aydin and A. Aysu, "Leaking secrets in homomorphic encryption with side-channel attacks," *Journal of Cryptographic Engineering*, pp. 1–11, 2024.

[16] L. Weissbart, "Performance analysis of multilayer perceptron in profiling side-channel analysis," in *Applied Cryptography and Network Security Workshops: ACNS 2020 Satellite Workshops, AIBlock, AIHWS, AIoTS, Cloud S&P, SCI, SecMT, and SiMLA, Rome, Italy, October 19–22, 2020, Proceedings 18*. Rome, Italy: Springer, 2020, pp. 198–216.

[17] Y. L. Liu, Y. K. Chen, W. X. Li, and Y. Zhang, "Model design and parameter optimization of cnn for side-channel cryptanalysis," *PeerJ Computer Science*, vol. 8, p. e829, 2022.

[18] A. A. Ahmed, M. K. Hasan, N. S. M. Satar, N. S. Nafi, A. H. Aman, S. Islam, and S. A. Fadhil, "Detection of crucial power side channel data leakage in neural networks," in *2023 33rd International Telecommunication Networks and Applications Conference*. IEEE, 2023, pp. 57–62.

[19] H. Wang, S. Forsmark, M. Brisfors, and E. Dubrova, "Multi-source training deep-learning side-channel attacks," in *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)*. Miyazaki, Japan: IEEE, 2020, pp. 58–63.

[20] R. Y. Acharya, F. Ganji, and D. Forte, "Information theory-based evolution of neural networks for side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2023, p. 401–437, 2023.

Amjed A. Ahmed received the B.Sc. degree in computer science from the University of Baghdad, and the M.Sc. degree in computer science from Binary University, Kuala Lumpur, Malaysia, in 2012. He is currently pursuing the Ph.D. degree with University Kebangsaan Malaysia, Malaysia, with a focus on artificial intelligence. From July 2013 to July 2022, he was a Lecturer with the Imam Al-Kadhum College, Baghdad, Iraq.

Mohammad K. Hasan received the Doctor of Philosophy (Ph.D.) degree in electrical and communication engineering from the Faculty of Engineering, International Islamic University, Malaysia, in 2016. He is currently with the Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), as a Senior Lecturer.

Shahrul A. M. Noah received the B.Sc. degree (Hons.) in mathematics from the Universiti Kebangsaan Malaysia, in 1992, and the M.Sc. and Ph.D. degrees in information studies from The University of Sheffield, U.K., in 1994 and 1998, respectively. He is currently a Professor with the Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, where he also heads the Knowledge Technology Research Group. His research interest includes information retrieval and ontology with special emphasis on semantic search and recommender systems. He has published more than 200 research articles in these areas. He is a member of International Association for Ontology and its Applications (IAOA) and of the IEEE Computer Science Society associations. He is currently the Chair of the Persatuan Capaian Maklumat dan Pengurusan Pengetahuan (PECAMP).

Azana H. M. Aman received the B.Eng., M.Sc., and Ph.D. degrees in computer and information engineering from International Islamic University Malaysia, Malaysia. She is currently working as Senior Lecturer at the Research Center for Cyber Security, Faculty of Information Science and Technology (FTSM), The National University of Malaysia, Malaysia. Her research interests include computer system and networking, computer information and network security, the Internet of Things (IoT), cloud computing, and big data.