Exploration into the Explainability of Neural Network Models for Power Side-Channel Analysis*

Anupam Golder
anupamgolder@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

Ashwin Bhat
ashwinbhat@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

Arijit Raychowdhury
arijit.raychowdhury@ece.gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

ABSTRACT

In this work, we present a comprehensive analysis of explainability of Neural Network (NN) models in the context of power Side-Channel Analysis (SCA), to gain insight into which features or Points of Interest (PoI) contribute the most to the classification decision. Although many existing works claim state-of-the-art accuracy in recovering secret key from cryptographic implementations, it remains to be seen whether the models actually learn representations from the leakage points. In this work, we evaluated the reasoning behind the success of a NN model, by validating the relevance scores of features derived from the network to the ones identified by traditional statistical PoI selection methods. Thus, utilizing the explainability techniques as a standard validation technique for NN models is justified.

CCS CONCEPTS

• **Security and privacy** → **Side-channel analysis and countermeasures**.

KEYWORDS

Neural Networks, Explainable Machine Learning, Side-Channel Analysis, Attribution-based Methods, PoI Selection

ACM Reference Format:
Anupam Golder, Ashwin Bhat, and Arijit Raychowdhury. 2022. Exploration into the Explainability of Neural Network Models for Power Side-Channel Analysis. In *Proceedings of the Great Lakes Symposium on VLSI 2022 (GLSVLSI '22), June 6–8, 2022, Irvine, CA, USA.* , 6 pages. https://doi.org/10.1145/3526241.3530346

1 BACKGROUND AND INTRODUCTION

Cryptographic algorithms usually undergo massive scrutiny by the cryptography community prior to standardization. Their security is analyzed from a mathematical point of view, and they are adopted only if they are assumed secure against known cryptanalysis attacks. Their implementations can, however, still leak information about their undergoing operations or data, in the form of, timing [17], power [16] or Electromagnetic (EM) [10] side channels. An adversary would typically use some known information (e.g. plaintext, or ciphertext), in conjunction with the side channel leakage, and utilize a leakage model to perform a non-profiled or profiled attack. Due to the requirement of large number of traces for non-profiled attack, or its non-applicability due to implementation of certain countermeasures, profiled attacks [6] have gained attraction in the community as a highly efficient form of attack, even if there are countermeasures in place [4, 18]. Optimized Multi-Layer Perceptron (MLP) [22] and Convolutional Neural Network (CNN) models [5, 15, 18] have been proposed to increase attack efficiency over traditional template based profiled attack methods (which are optimal from an information theoretic point of view). This approach (Black-box NN, Fig. 1 (a)) provides highly efficient attack methods, but little insight into why it works better. Explainablity incorporated into NN models, or in other words, White-box NN approach (1 (b)) would provide better understanding of how the networks learn, and help identify leakage points or Points of Interest (PoI) in newly proposed implementations. This is of value to designers devising and implementing countermeasures. In this work, we focus on this perspective. Instead of proposing efficient attack methods, we demonstrate how explainability helps understand the model's classification decision or inference.

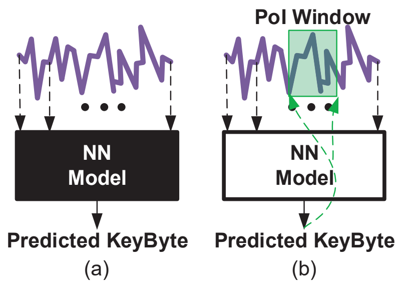

Figure 1: (a) Traditional Black-box NN Approach (b) White-box NN Approach to offer explainability (premise for this work)

Explainability techniques have been largely utilized in image classification problems [3, 25, 26, 28] to provide insight into the success or failure of a NN model. These are usually validated by visual inspection as the features that should be given importance by the explanation method are interpretable to a human as well. In contrast, the power or EM traces obtained from any cryptographic implementation look almost the same to a human eye if they are

*This work was supported in part by the National Science Foundation (NSF) under CNS Grant No. 1935534 and 1717467, and in part by Semiconductor Research Corporation (SRC) Task 3059.001 and 2969.001.

performing the same operation. Only with the application of statistical methods, any discriminative set of features can be identified. Thus, validating the explainability of widely successful NN models using traditional PoI selection methods can be a useful way to build confidence in the inference of the models. This can be extended to the cases where traditional PoI selection methods are not directly applicable, but NN models are still successful.

Our contributions can be summarized as follows:

- We provide validation of explainability of NN Models using PoI selection methods well known to the side-channel analysis community. This demonstrates that MLP and CNN models, when trained appropriately, give attributes to the same leakage points.
- We demonstrate that when the same model is retrained to attack different keybytes, the attribution window changes from one keybyte to the next, as expected for a sequential software implementation, which processes different keybytes at different points in time.
- We show how increasing the number of training traces improves the relevance scores obtained from NN analysis.

2 RELATED WORKS

MLP and CNN are among the most widely used NN models in the context of side-channel analysis. CNN has been the method of choice for its success against trace misalignment, jitter, and masking countermeasures. Recently, there is a growing interest in the explainablity of NN models used in SCA. The first reported work on attributions of NN models for SCA was presented in [12], where the authors used Saliency Maps [25], Layerwise Relevance Propagation (LRP), and Occlusion [34], to provide explanations for the predictions of NN Models by visualizing the features contributing the most to the inference. The work in [23] concentrated on a perturbation-based analysis to mimic uncertainties associated with measurements, sub-optimal classifiers, and countermeasures, where authors used Pearson correlation coefficient [13] to select the most important features, and proposed a framework for resilient NN model training. The works in [32, 33] used Gradient Visualization, and Weight Visualization, and utilized Signal-to-Noise Ratio (SNR) as a way to evaluate the precision of leakage detection of trained CNN models. Weight visualization, and Occlusion were used in [11] as a way to infer leakage attributions for two different keybytes. Guessing Entropy Bias-Variance Decomposition was used in [30] as a way to understand how a change in experimental setting influences the attack performance. The work in [20] used Gradient Visualization to identify leakage points, and compared them with the ones identified by SNR. Singular Vector Canonical Correlation Analysis (SVCCA) tool was used in [29] to find similarity in representations learned by trained NN models. The work in [24] did not look into explainability, rather provided an interesting way of combining Correlation Power Analysis (CPA) with NNs, to maximize correlation coefficient instead of minimizing cross-entropy loss. Ablation-based methodology was used in [31] to identify which layers of a trained NN models processes countermeasures. The work in [14] proposed Class Gradient Visualization and Weight visualization and compared the identified leakage points with those obtained by CPA. In all of these, what is lacking, though, is a comparative validation using traditional PoI selection methods for the relevance scores of the features obtained by explainable NN analysis techniques.

3 EXPERIMENTAL SETTING

Power traces were captured from CW308T-XMega, an 8-bit Atmel AVR XMega128 microcontroller based SCA platform from ChipWhisperer [21], while running an unprotected software implementation of Advanced Encryption Standard (AES)-128 at 7.37 MHz. The traces were sampled at 29.48 MHz, by inserting a small resistor ($500m\Omega$) in series with the power supply to measure instantaneous power consumption. In this attack setting, the plaintext was fixed, and we collected $50k$ traces with $2k$ samples each by varying the keybytes (from $0x00$ to $0xFF$) . The NN model is trained to recover a target keybyte and this process is repeated to recover all the keybytes. The ChipWhisperer Capture platform provides traces that are already synchronized, enabling us to utilize PoI selection methods and employ MLP, without any pre-processing.

The NN models were trained and evaluated on the same dataset containing $50k$ traces, where, $40k$ traces were used for training and validation (90%-10% split), and the remaining $10k$ traces were used for testing. Prior to training/testing, the features were normalized using:

$$trace_i' = \frac{trace_i - E[trace_i]}{\sqrt{\sigma_i^2 + \epsilon}} \tag{1}$$

where, E[.] denotes expected value, σ_i represents the standard deviation for the feature or sample index, i, and a small ϵ is chosen to avoid division by zero. We do not apply any PoI selection method before feeding the data into the NN models. Rather, we expect that if the NN models are trained properly, they should give high relevance scores to the same leakage points we would otherwise select. We assume an identity leakage model, labeling each trace based on the keybyte value it is operating on. The models were trained for 100 epochs, for a batch size of 128, using Adam optimizer with a learning rate of 0.001, to minimize categorical cross-entropy loss. The layers of the NN models used in this work are listed in Table 1. The Dense layers except the last one have Rectified Linear Unit (ReLU) activation, while the last one has softmax activation. All the kernels have L2 regularization penalty of 0.0001. NN models were implemented in Python, using the keras [7] library with tensorflow [1] as backend. The test accuracy reported in Table 1 is obtained by averaging test accuracy across all the keybytes for single-trace attack. In this work, we solely focus on test accuracy as a metric for how efficiently the NN models were trained, though, to measure the efficiency of an attack, Guessing Entropy [27] is sometimes preferred as a metric.

4 TRADITIONAL POI SELECTION METHODS

To address curse of dimensionality, different PoI selection methods have been proposed and evaluated empirically in literature, such as, Difference of Means (DoM), Pearson Correlation Coefficient, SNR, etc., all of which work reasonably well in identifying the leakage points. We also used Minimum Redundancy Maximum Relevance (MRMR) feature selection algorithm in this context to find minimal-optimal set of points.

Table 1: Neural Network Model

	MLP	CNN

Layers	Dense(100)
	BatchNorm
	Dropout(0.2)
	Dense(100)
	BatchNorm
	Dense(100)
	BatchNorm
	Dense(256)
		Conv1D(50,60)
		Conv1D(100.60)
		Maxpooling1D(3)
		Flatten
		Dropout(0.5)
		Dense(100)
		BatchNorm
		Dropout(0.5)
		Dense(256)

Test Accuracy	99.6%	99.7%

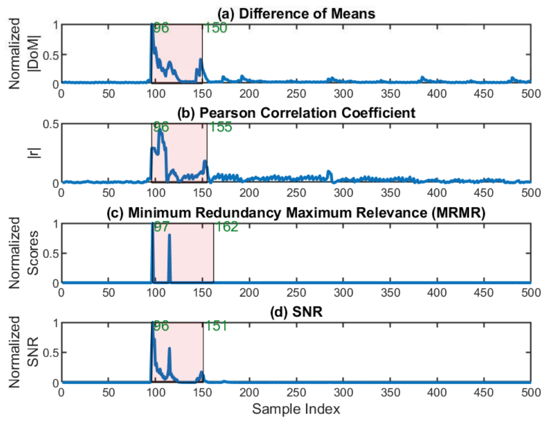

Figure 2: PoI Selection for the first keybyte using: (a) DoM, (b) Pearson Correlation Coefficient, (c) MRMR, (d) SNR

4.1 Brief Overview of PoI Selection Methods

4.1.1 Difference of Means (DoM). DoM is typically used in template attacks [6],[8] as a PoI selection method to reduce complexity of computation. DoM for each sample index i of trace, $trace_i$ can be computed by evaluating the sum of the absolute value of pairwise differences for different class labels, using:

$$DoM_i = \sum_{k=0}^{255}\sum_{j=0}^{255} |trace_{avg}(label_k)_i - trace_{avg}(label_j)_i| \qquad (2)$$

where $trace_{avg}$ is obtained by averaging all the traces for the same class labels or keybyte values.

4.1.2 Pearson Correlation Coefficient. Pearson correlation coefficient indicates the linear dependency between two variables, where +1 denotes perfect positive correlation, -1 denotes perfect negative correlation, and 0 denotes no correlation. Correlation coefficient, r_i between i-th sample of traces, $trace_i$ and the corresponding label, $label$ can be computed using the following equation [13]:

$$r_i = \frac{\sum_j((trace(j)_i - E[trace(j)_i])(label(j) - E[label])}{\sqrt{\sum_j((trace(j)_i - E[trace(j)_i])^2}\sqrt{\sum_j(label(j) - E[label])^2}} \qquad (3)$$

4.1.3 Signal to Noise Ratio (SNR). SNR [19] for each sample $trace_i$ for different keybyte candidates can be computed using the equation :

$$SNR_i = \frac{Var[E[trace_i|label]]}{E[Var[trace_i|label]]} \qquad (4)$$

where E[.] denotes expected value and Var[.] denotes the variance.

4.1.4 Minimum Redundancy Maximum Relevance (MRMR). Mutual Information between two discrete random variables X and Y,

$$I(X,Y) = \sum_{x,y} P(X=x, Y=y).log\frac{P(X=x,Y=y)}{P(X=x)P(Y=y)} \qquad (5)$$

quantifies the reduction in uncertainty of one variable after seeing the other variable. This is used by MRMR algorithm [9], which ranks the set of input features required to represent the output variable, by computing Mutual Information Quotient (MIQ) value of each feature $trace_i$,

$$MIQ_i = \frac{I(trace_i, label)}{\frac{1}{|F|}\sum_j I(trace_i, trace_j)} \qquad (6)$$

where the numerator denotes the relevance, the denominator represents the redundancy, and $|F|$ is the total number of features. A large value of MIQ suggests that the corresponding feature is important, as well as, a large drop in MIQ for the next important feature indicates the confidence in feature selection.

4.2 Application of PoI Selection Methods to the dataset

Fig. 2 illustrates the PoI window and feature scores obtained by applying different PoI selection methods to the dataset. This forms the ground truth in our effort to explain the attributions obtained from NN models. From Fig. 2 (a)-(d), we observe that, the PoI window roughly spans from sample index 96 through sample index 155. Pearson correlation coefficient Fig. (2 (b)) indicates that many of the samples are correlated to the output class label, which is why, MRMR (2 (c)) can reduce the feature set to a much smaller size. DoM (Fig. 2 (a)) and SNR (Fig. 2 (d)) are very similar in their PoI selection, because they focus on similar metrics.

5 EXPLAINABLE NEURAL NETWORK

Despite Deep Neural Network (DNN) models showing good performance in a diverse set of domains, in most cases, they are still treated as black-boxes with no real insight into why they perform so well. Feature attribution methods have recently been developed with the purpose of shedding light onto the classification decisions taken by these models. They assign relevance scores to each input feature to quantify and explain its contribution to the overall decision. In image-classificaiton tasks, these scores are visualized as a heatmap to identify regions of interest in the input domain. Some

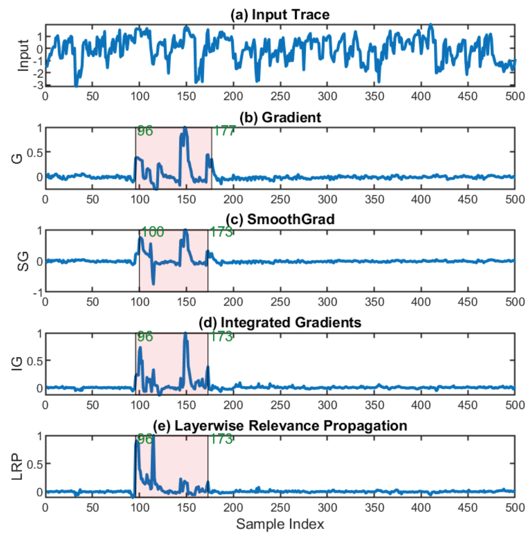

Figure 3: Analysis of MLP for the first keybyte: (a) Input trace (b) Gradient (c) SmoothGrad (d) Integrated Gradients (e) LRP

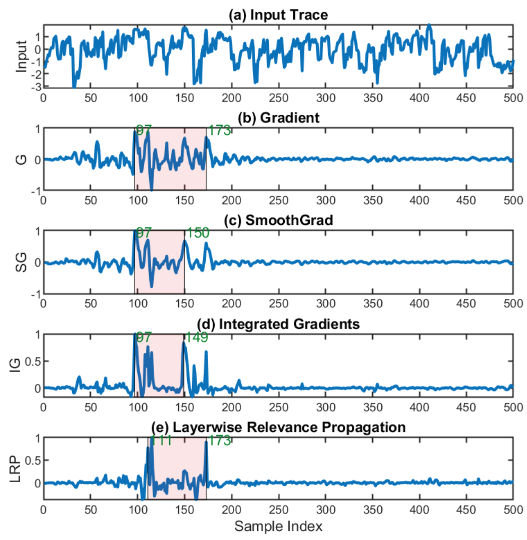

Figure 4: Analysis of CNN for the first keybyte: (a) Input trace (b) Gradient (c) SmoothGrad (c) Integrated Gradients (d) LRP

of the attribution methods that we employed are described in this section.

5.1 Brief Overview of Attribution-based Methods

5.1.1 Gradients or Saliency Map. In classification tasks, the output layer has as many neurons as there are classes in the dataset. The class corresponding to the output neuron with the highest activation is chosen as the decision of the overall model. The simplest way to evaluate feature importance ($R_i(x)$) is to evaluate the gradient of the classification probability ($S_c(x)$) of the chosen output neuron with respect to the input features (x_i). This is done by adding a gradient backpropagation step to the inference. A large value of the gradient implies that small changes in the value of the input feature would produce large changes in the model's classification score; thereby indicating the higher relevance of that particular input feature [25]. Saliency maps do not differentiate between positive and negative contributions of the input features by considering the absolute value of the gradient, computed by:

$$R_i(x) = \frac{\partial S_c(x)}{\partial x_i} \tag{7}$$

5.1.2 SmoothGrad. DNN models with several stacked layers are highly non-linear. This non-linearity leads to large variation in the gradients in the neighborhood of the input. However, the attribution methods should be stable - small changes in the input (noise) should not perturb the relevance scores. In order to overcome this limitation of gradients, SmoothGrad [26] proposes to run inference in the presence of noise. Multiple copies of the input are created by adding a bounded zero-mean Gaussian noise ($\mathcal{N}(0, \sigma^2)$). The gradient heatmaps of these noisy copies, $R(x + \mathcal{N})$, are then averaged to generate the final heatmap $\hat{R}(x)$, defined by:

$$\hat{R}(x) = \frac{1}{N}\sum_{n=1}^{N} R(x + \mathcal{N}(0, \sigma^2)) \tag{8}$$

The input features that are actually relevant would have a large gradient value in all copies while spuriously large gradient values would be suppressed owing to the averaging process.

5.1.3 Integrated Gradients (IG). Gradient based attribution is a local explanation method. It only looks at how small changes in the input features affect the classification probability. At times, the contributions of certain features can saturate beyond a certain threshold value. This leads to small gradient values even when that particular feature is relevant. Integrated gradients (IG) [28] aims to solve this vanishing gradient attribution problem. It first defines a baseline input (x') for the task at hand. For instance, in image classification, a good baseline is all the pixel values set to white or black. The product of the gradients and the deviation from baseline is integrated along this straight-line interpolation path. By doing so, IG accumulates the effect of each feature as it moves from its baseline (zero contribution) to its actual value (relevance). In practice, the integration is replaced by a discrete sum over N interpolated inputs as shown in the equation below:

$$IG(x) = (x - x') \times \frac{1}{N} \times \sum_{n=1}^{N} R(x' + \frac{n}{N}(x - x')) \tag{9}$$

5.1.4 Layerwise Relevance Propagation (LRP). LRP [3] defines rules to progressively redistribute the output prediction score in a layer by layer manner till we reach the input features. This process adheres to relevance conservation, i.e., the total relevance that flows into a neuron at a particular layer, flows out to the neurons at the layer below it. The parameters in the LRP rule allow us to control the relative importance given to features that contribute positively against those that contribute negatively to the overall classification. The advantage of LRP over SmoothGrad is that it performs gradient smoothing in a single backward pass through the network. The equation below (LRP-ε) defines the rule to redistribute the relevance from layer k to the previous layer j. Here, a_j is the activation value in layer j and w_{jk} is the weight. The parameter ε is included for numerical stability.

$$R_j = \sum_k \frac{a_j w_{jk}}{\varepsilon + \sum_{0,j} a_j w_{jk}} R_k \tag{10}$$

5.2 Analysis of NN models using Attribution-based methods

We used the iNNvestigate toolbox [2] to analyze the NN models. This provides an API-like interface to analyze NN models using different analysis methods, such as, IG, Smoothgrad, Gradients, LRP etc.

Fig. 3 (b)-(e) shows the relevance scores obtained from the attribution based methods for the trained MLP, where Fig. 3 (a) shows the input trace. All the analysis methods highlight nearly the same region of the input trace, although the relevance scores are different. Similar trend can be seen for CNN in Fig. 4. The noisier scores can be attributed to the overparameterization of the CNN model. It is illuminating to observe that both MLP and CNN attribute relevance to the same PoI window, which the traditional PoI selection methods identify in Fig. 2. This matches the intuition that, if the NN models are trained properly, they should learn from the same leakage points that traditional PoI selection methods would identify, if applicable. Fig. 5 illustrates how number of training traces affects the relevance scores provided by the IG method. We notice that the relevance scores are noisier with limited training traces, and improves as the number of training traces is increased. Fig. 6 illustrates that the attribution scores obtained from trained MLP and CNN accordingly moves as the attacked keybyte is changed. In this case, 4 different keybytes ($KB0$-$KB3$) were attacked by retraining the same networks with corresponding labels. This also matches the intuition that for a sequential software implementation, the leakage for different keybytes will be at different sample indices, which an adequately trained NN model should be able to demonstrate. Fig. 7 illustrates how the number of training epochs influences relevance scores. We note that it remains relatively unaffected by the number of epochs, if the number of training traces is sufficient (in this case, the whole training set was used).

6 CONCLUSION

In this work, we presented validation for our trained NN models using domain knowledge obtained by applying PoI selection methods. This gives the confidence that the predictions obtained from the NN models are learned and based on valid leakage points. We verified the intuition that PoI window should move along with the index of the attacked keybyte for an adequately trained NN model. In the SCA context, we expect that incorporating attribution-based

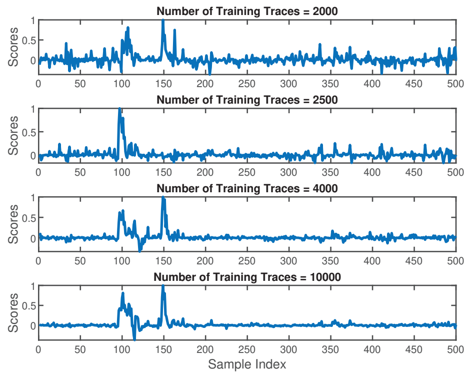

Figure 5: Effect of Number of Training Traces on Relevance Scores for MLP using Integrated Gradients

validation methods will help designers understand both the performance of the NN models and identify the leakage points missed by traditional PoI selection methods. Future works may focus on explainable NN models for implementations with countermeasures to assess vulnerabilities.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. 2019. iNNvestigate neural networks! *J. Mach. Learn. Res.* 20, 93 (2019), 1–8.

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

[4] Ryad Benadjila, Emmanuel Prouff, Rémi Strullu, Eleonora Cagli, and Cécile Dumas. 2020. Deep learning for side-channel analysis and introduction to ASCAD database. *Journal of Cryptographic Engineering* 10, 2 (2020), 163–188.

[5] Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. 2017. Convolutional neural networks with data augmentation against jitter-based countermeasures. In *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 45–68.

[6] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. 2002. Template attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 13–28.

[7] François Chollet et al. 2018. Keras: The python deep learning library. *Astrophysics Source Code Library* (2018).

[8] Marios O Choudary and Markus G Kuhn. 2018. Efficient, Portable Template Attacks. *IEEE Transactions on Information Forensics and Security* 13, 2 (2018), 490–501.

[9] Chris Ding and Hanchuan Peng. 2005. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology* 3, 02 (2005), 185–205.

[10] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. 2001. Electromagnetic analysis: Concrete results. In *International workshop on cryptographic hardware and embedded systems*. Springer, 251–261.

[11] Aron Gohr, Sven Jacob, and Werner Schindler. 2020. Subsampling and Knowledge Distillation on Adversarial Examples: New Techniques for Deep Learning Based Side Channel Evaluations. In *International Conference on Selected Areas in Cryptography*. Springer, 567–592.

[12] Benjamin Hettwer, Stefan Gehrer, and Tim Güneysu. 2019. Deep neural network attribution methods for leakage analysis and symmetric key recovery. In *International Conference on Selected Areas in Cryptography*. Springer, 645–666.

[13] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning*. Vol. 112. Springer.

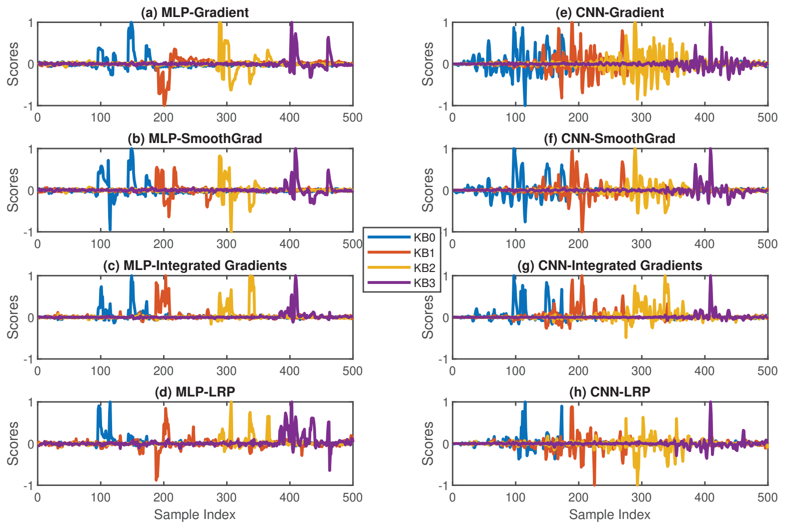

Figure 6: Movement of attribution window with the index of the attacked keybyte for MLP and CNN, on the same trace

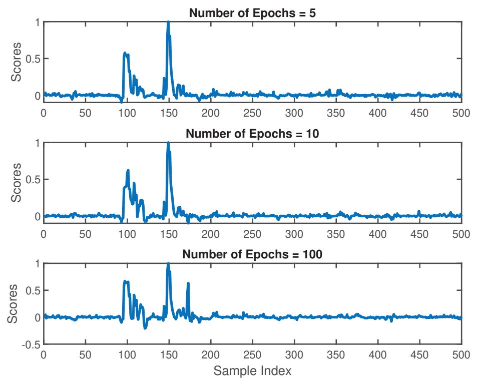

Figure 7: Effect of Training Epochs on Relevance Scores for MLP using Integrated Gradients

[14] Minhui Jin, Mengce Zheng, Honggang Hu, and Nenghai Yu. 2020. An enhanced convolutional neural network in side-channel attacks and its visualization. *arXiv preprint arXiv:2009.08898* (2020).

[15] Jaehun Kim, Stjepan Picek, Annelie Heuser, Shivam Bhasin, and Alan Hanjalic. 2019. Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2019), 148–179.

[16] Paul Kocher, Joshua Jaffe, and Benjamin Jun. 1999. Differential power analysis. In *Annual international cryptology conference*. Springer, 388–397.

[17] Paul C Kocher. 1996. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Annual International Cryptology Conference*. Springer, 104–113.

[18] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. 2016. Breaking cryptographic implementations using deep learning techniques. In *International Conference on Security, Privacy, and Applied Cryptography Engineering*. Springer, 3–26.

[19] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. 2008. *Power analysis attacks: Revealing the secrets of smart cards*. Vol. 31. Springer Science & Business Media.

[20] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. 2019. Gradient visualization for general characterization in profiling attacks. In *International Workshop on constructive side-channel analysis and secure design*. Springer, 145–167.

[21] Colin O'Flynn and Zhizhang David Chen. 2014. Chipwhisperer: An open-source platform for hardware embedded security research. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 243–260.

[22] Stjepan Picek, Annelie Heuser, Alan Jovic, Shivam Bhasin, and Francesco Regazzoni. 2019. The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019, 1 (2019), 1–29.

[23] Stjepan Picek, Annelie Heuser, Lichao Wu, Cesare Alippi, and Francesco Regazzoni. 2018. When theory meets practice: A framework for robust profiled side-channel analysis. *Cryptology ePrint Archive* (2018).

[24] Pieter Robyns, Peter Quax, and Wim Lamotte. 2019. Improving cema using correlation optimization. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2019), 1–24.

[25] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[26] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017).

[27] François-Xavier Standaert, Tal G Malkin, and Moti Yung. 2009. A unified framework for the analysis of side-channel key recovery attacks. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 443–461.

[28] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*. PMLR, 3319–3328.

[29] Daan van der Valk, Stjepan Picek, and Shivam Bhasin. 2020. Kilroy was here: The first step towards explainability of neural networks in profiled side-channel analysis. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 175–199.

[30] Daan van der Valk and Stjepan Picek. 2019. Bias-variance decomposition in machine learning-based side-channel analysis. *Cryptology ePrint Archive* (2019).

[31] Lichao Wu, Yoo-Seung Won, Dirmanto Jap, Guilherme Perin, Shivam Bhasin, and Stjepan Picek. 2021. Explain some noise: Ablation analysis for deep learning-based physical side-channel analysis. *Cryptology ePrint Archive* (2021).

[32] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli. 2020. Methodology for efficient CNN architectures in profiling attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2020, 1 (2020), 1–36.

[33] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli. 2020. Understanding methodology for efficient CNN architectures in profiling attacks. *Cryptology ePrint Archive* (2020).

[34] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*. Springer, 818–833.