J Cryptogr Eng manuscript No.
(will be inserted by the editor)

Deep Learning Mitigates but Does Not Annihilate the Need of Aligned Traces and a Generalized ResNet Model For Side-channel Attacks

Yuanyuan Zhou[1,2] · **François-Xavier Standaert**[1]

Received: date / Accepted: date

Abstract We consider the question whether synchronization / alignment methods are still useful / necessary in the context of side-channel attacks exploiting deep learning algorithms. While earlier works have shown that such methods / algorithms have a remarkable tolerance to misaligned measurements, we answer positively and describe experimental case studies of side-channel attacks against a key transportation layer and an AES S-box where such a preprocessing remains beneficial (and sometimes necessary) to perform efficient key recoveries.

Our results also introduce generalized Residual Networks as a powerful alternative to other deep learning tools (e.g., Convolutional Neural Networks and Multi-Layer Perceptrons) that have been considered so far in the field of side-channel analysis. In our experimental case studies, it outperforms the other three published state-of-the-art neural network models for the data sets with and without alignment, and it even outperforms the published optimized CNN model with the public ASCAD[1] data set. Conclusions are naturally implementation-specific and could differ with other datasets, other values for the hyper-parameters, other machine learning models and with other alignment techniques.

Keywords Deep learning · residual networks · side-channel attack · alignment methods · embedded security

Yuanyuan Zhou[1,2]
[1] ICTEAM/ELEN/Crypto Group, UCLouvain, Belgium.
Tel.: +31-15-2692500
Fax: +31-15-2692555
E-mail: zhou@brightsight.com
[2] *Brightsight BV, The Netherlands.*

François-Xavier Standaert
[1] ICTEAM/ELEN/Crypto Group, UCLouvain, Belgium.

[1] https://github.com/ANSSI-FR/ASCAD

1 Introduction

Past research has shown an increasing interest from the side-channel community regarding the use of machine learning / deep learning techniques as a powerful way to exploit physical leakages with limited knowledge of the target implementations: see for example [4, 10, 11, 12, 14, 15, 17, 18, 19, 20, 22, 23]. In general, one potential advantage of these techniques compared to more conventional statistical tools (e.g., Gaussian templates [5] and linear regression [24]) is that they quite efficiently deal with large dimensionalities (which may prevent the need of estimating large covariance matrices, or to rely on dimensionality reduction [2]). This intuition has been recently put forward by Cagli et al. at CHES 2017, they demonstrate that (e.g., deep) learning algorithms are good candidates to exploit the leakage of implementations protected with jitter-based countermeasures [3].

In this paper, we mitigate a tempting shortcomingin the interpretation of these past results, namely the fact that side-channel attacks based on deep learning do not benefit from re-synchronization. We insist that such a shortcoming is not induced by previous authors, in particular the CHES 2017 ones. We only consider it as a natural question to confirm whether or not such algorithms sometimes benefit from some sort of pre-processing. We believe the question is of importance since a general negative answer would significantly simplify the life of evaluation laboratories. For this purpose, we describe experimental case studies based on two protected implementations (one targeting a key transportation layer, the other targeting an AES S-box), the measurements of which are affected by misalignments and hardware interrupts. In both cases we show that the application of a re-synchronization pre-processing before the application of a deep learning algorithm actually allows reducing the data complexity of the attacks. In the second case, we even con-

sider a "compressive" alignment (i.e., reducing the number of samples in the pre-processed traces because we can use shorter interval after the alignment), which was not only necessary from the data complexity point of view, but also highly beneficial from the time complexity point of view (to maintain a reasonable execution time for the deep learning attacks). From an information theoretic viewpoint, the latter can only reduce the total amount of information in the traces, hence showing clear experimental evidence that re-synchronization can make the traces easier to exploit even for deep learning algorithms.

As an additional result, we also introduce the use of Residual Networks (ResNet) as an alternative to the traditional Convolutional Neural Networks (CNN) and Multi-Layer Perceptrons (MLP) that have been previously considered in the literature. We compare its performance with the other three state-of-the-art neural network models in a side-channel context, namely the ASCAD_CNN model [22], the SPACE_CNN model [15] and the SCANet model [20]. The results with all our 6 data sets with or without alignment demonstrate that this ResNet model quite systematically outperforms the other three models.

Cautionary note. We acknowledge that these conclusions may be affected by the level of profiling of the implementations. In theory, for an infinite amount of profiling on raw traces, re-synchronization may become useless for deep learning as for any well-specified multivariate side-channel attack. A similar statement holds for the choice of parameter for the ResNet we exploited. In this respect, our goal is only to show that for a realistic amount of profiling and standard use of a popular deep learning algorithm, re-synchronization may help. More generally, our conclusion is admittedly based on an experimental basis. So the conclusions of this paper could be different with other datasets, other values for hyper-parameters, other machine learning models or with other alignment techniques.

The rest of the paper is organized as followed. Section 2 introduces the alignment method that we used, the necessary background on deep Residual Networks and the target implementations that we investigated. Section 3 describes the verification of the ResNet model we adopted regarding its capability in a side-channel context by analyzing AES traces obtained from the ChipWhisperer Lite board and the comparison with the state-of-the-art neural network models regarding performance. Section 4 and Section 5 respectively show the experimental results for the application of this ResNet model with the impact of misalignment against our two main targets, and the comparison of their performances. Finally, in Section 6 we further demonstrate the good generalization of our ResNet model by comparing its performances with the published optimized ASCAD_CNN model using all 16 S-boxes data from their published ASCAD data set.

2 Background

2.1 Target implementations

We investigated two main target implementations in this work, and additionally used the simple case of an AES software implementation on the ChipWhisperer Lite board for preliminary assessments. In this warming up case, we capture 90,000 profiling power traces with a 16-byte randomized AES input and key, and 10,000 attack power traces with a fixed random AES key and randomized AES input.

For our first Device Under Test (DUT1), we target the AES key during its transportation. The latter is important for security evaluations, since it frequently happens that the keys encrypted in some Non-volatile Memory (NVM) has to be transported to the cryptographic co-processor when invoking the corresponding cryptographic encryption/decryption operations. In case the key is decrypted/masked/unmasked during this transportation, it may lead to additional sources of leakages that could be exploited by an adversary. Concretely, our DUT is a modern 32-bit secure microcontroller with a built-in secure AES coprocessor and the AES key is encrypted and stored in an EEPROM. To invoke AES encryption, the encrypted AES key is decrypted and masked during its transfer from EEPROM to the AES coprocessor. We acquire 80,000 EM traces for profiling and 20,000 traces for attack.

Finally, our second DUT is a more standard case of an AES co-processor for another secure microcontroller resistant to first-order leakage, where we target the S-box output of the first block cipher round. For this one, we measure 500,000 EM traces for profiling and 30,000 traces for attack.

We note that the different ratios between the numbers of profiling and attack traces can be connected to the fact that it is in general more complex to profile a leaking device than to attack it once a model is well estimated [27]. As a result, the more secure an implementation (e.g., due to masking or other countermeasures), the larger this ratio can be.

2.2 Alignment method

Due to the complex architecture (and potential countermeasures) of our DUT, their raw measured EM traces are very misaligned. To get better aligned traces for our investigations, we use the same method as in [21], which exploits correlation in order to synchronize the EM traces focusing on the leakage part of targeted sensitive data (e.g., the AES key for the transportation case in Section 4 and the AES S-

box output for the case in Section 5). This method works in three steps.

– Firstly, a searching interval A that contains the operation to be synchronized is manually selected among all the traces.
– Secondly, a smaller reference interval B_q specific to each trace q is also manually chosen.
– For each trace, we finally find the portion to be synchronized by using the second window B_q to search over the whole interval A. The right portion is selected as the one having the maximum correlation with the reference interval. If the correlation is lower than a given threshold (chosen by the attacker/evaluator), the trace is assumed not good enough and discarded.

Note that concretely, this method was usually applied multiple times for each DUT by targeting different time intervals (for both the searching interval and reference interval). During the measurement, no very distinguishable feature close to the target interval can be used to trigger the oscilloscope. So we have to align the traces step by step to get close to the target interval, and then within the target interval we do more local alignments. Roughly, we choose new intervals when the misalignment is getting larger, and we repeat this process until the target interval is well aligned. This allowed us to recursively improve the alignment for the part of the traces that corresponds to the target leakage. Before we measure the traces for profiling attacks, we used SPA/SEMA and CPA/CEMA techniques to narrow down the potential target interval. For DUT1, we first try to find the leakage of key data using a CPA by measuring the whole AES encryption command execution with a randomized AES key data per execution, so the interval is rather large. We try different alignments and calculate the correlation of key data after each alignment. For DUT2, we cannot find the leakage of the S-box output because it is a masked implementation, but we figure out which interval is likely related to AES encryption by SPA and SEMA techniques. The length of the target interval is changing when the input length is increasing. In order to recursively align the traces to the target interval, we accordingly choose the distinguishable feature of most traces step by step. After we recursively aligned the traces to the target interval, the segment around the target interval of each trace is taken off from the original traces to be used for profiling attacks. In general, the required time of this recursive alignment process depends on the number of traces, the number of sample points per trace and the size of the chosen interval. For DUT1 it takes less than one hour and for DUT2 it takes about 7 hours.

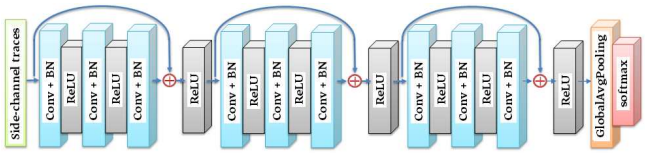


Fig. 1: Structure of ResNet.

2.3 Deep Residual Network

Since they have been proposed/applied in the computer vision field with great successes [8], deep ResNets [9] have been widely applied in different fields such as machine translation [30], speech synthesis [29], speech recognition [31] and AlphaGo [25]. Thanks to the open source deep learning libraries Keras [6] and Tensorflow [1], it is straightforward to build a model for performing profiling attacks in a side-channel context. We refer to these original papers for the details of the method and next list the parameters that we used in our experiments.

The core design idea of ResNets is to extend neural networks to very deep structures without degradation problems thanks to a so-called *shortcut connection*. ResNet is a stack of residual blocks: each residual block consists of several layers and a shortcut connection, the shortcut connection connects the input and output of that residual block. As depicted in Figure 1, the latter is inserted in each residual block such that the gradient flows directly through the bottom layers. Each residual block consists of three basic blocks and each basic block is composed of three layers: a convolutional layer γ followed by a batch normalization layer β [13] and a ReLU activation layer σ [16]. After stacking three residual blocks, a global average pooling layer δ is adopted (instead of a fully connected layer), in order to reduce the number of weights to be trained. Finally a softmax layer s is adopted to generate the class label of the input side-channel trace. In summary, the ResNet model can be written as:

$$\texttt{ResNet} = s \circ \delta \circ [\sigma \circ [\beta \circ \gamma \circ [\sigma \circ \beta \circ \gamma]^{n_1-1} \oplus 1]]^{n_2}, \qquad (1)$$

where n_2 denotes the number of residual blocks (we set it to 3 for all our experiments) and n_1 is the number of basic blocks per residual block (we also set it to 3 in this work). We further use 128, 256 and 256 filters for these three residual blocks. That is, all the convolutional layers within one residual block are using the same amount of filters. Following the original ResNet paper [8], we also did not use a dropout layer.

2.4 Accuracy, Loss and key rank

Accuracy and *Loss* are twin metrics that are widely used in the machine learning community to monitor and evaluate

neural network models. Training accuracy is the successful classification rate over the training data and training loss is the error rate over the training data. After each epoch, the trained model is applied to the validation data to calculate validation accuracy and validation loss. These two values indicate how good the trained model is at predicting outputs for inputs it has never seen before. Validation accuracy increases initially and saturates as the model starts to overfit. We also use the key rank (i.e., guessing entropy) as classical metric for side-channel security evaluations [28].

3 Warming up: results on CWL (AES S-box)

In order to verify that the ResNet model is applicable in a side-channel context, we used it to analyze an unprotected AES implementation on the ChipWhisperer Lite board, and fed the Hamming weight of the first-round S-box outputs into the ResNet model as described in 2.3. We then performed key recovery by classifying the Hamming weight of the S-box outputs of the attack traces and translating this into key information. We followed the best practice of deep learning to use balanced data per class (also for our other experiments), and we captured 90,000 profiling traces as mentioned in Section 2.1. Due to the uneven distribution of the 9 Hamming weight classes, in the end we have only 320 profiling traces per Hamming weight class. We use 20% profiling traces as validation data to improve the training of the weights and 3,000 sample points per trace to feed into the ResNet model. Figure 2 displays the rank of the correct subkey candidates of all 16 S-boxes. As can be seen, with a few attack traces the correct subkeys of S-box 8 and S-box 16 can be disclosed. All the other S-boxes show similar results except that S-box 15 needs a few hundred traces to recover the subkey. For the CWL experiments, we use a batch size of 32 and 100 epochs, “Adadelta” optimizer with an initial learning rate of 1.0, adaptively reducing the learning rate with a factor of 10 if the validation loss is not decreased within 5 consecutive epochs.

3.1 Performance comparison with the state-of-the-art

To compare the performance of our ResNet model with other state-of-the-art neural networks in a side-channel context, we performed the similar attacks using other three neural network models published in [15, 20, 22].

Those three models are the best ones according to their experiments and we are using their default hyper-parameters. (We did not have enough information to replicate the CHES 2017 CNN model [3]). We adapt those models to 9 Hamming weight classes. Figure 3 compares the rank of correct subkey candidates of S-box 8 and S-box 16 respectively using our ResNet model and the other 3 models. It can be observed that our ResNet model compares positively to the others (in terms of convergence).

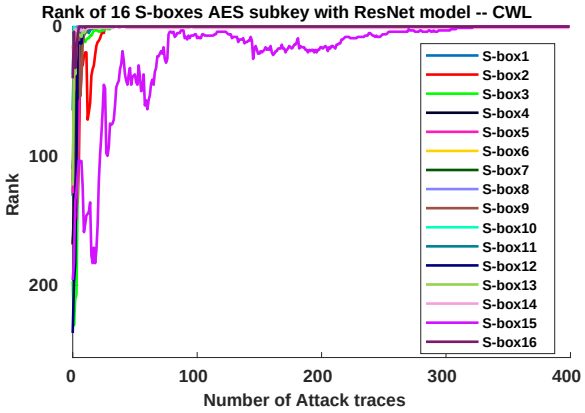


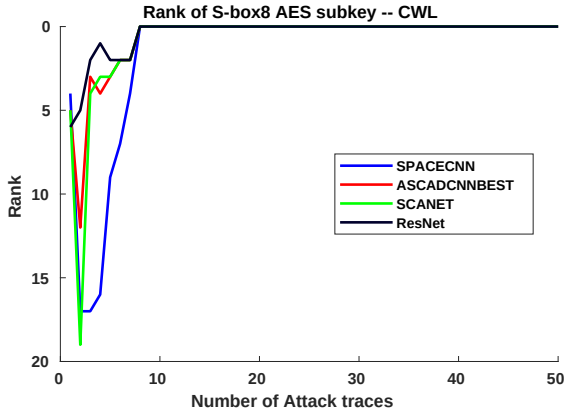
Fig. 2: Rank of correct subkey candidates of all 16 S-boxes with CWL data set.

4 Experimental results on DUT1 (AES key transfer)

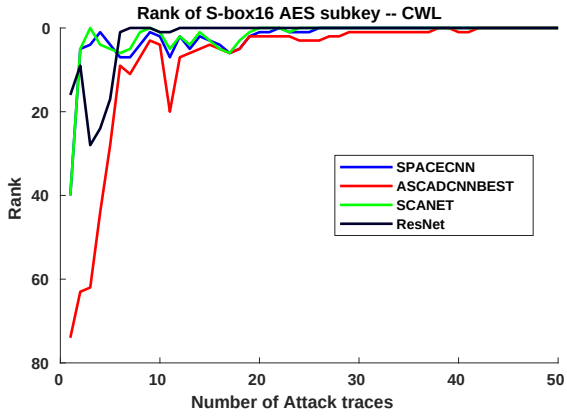
We start with an analysis of a key transportation layer which is a less usual target in the academic literature, but a quite important one in industry. The fact that key transportation leaks information on the key is indeed a critical weakness.

In this context, our profiling traces are obtained by randomizing all the 16 bytes of AES key and fixing the 16-byte AES input to execute AES encryption (the fixed input data is not detrimental since we are targeting the key transportation). For the attack traces, it is pretty different: we are targeting the value of the first byte of a 16-byte AES key considering the other bytes are leaking in a similar way. We randomize the first byte from $0x00$ to $0xFF$, so that we have 256 classes of attack traces for it and set the other 15 bytes to fixed random values. That is a realistic scenario that the attackers normally encounter: when an attacker wants to attack a fixed AES key, he needs to attack the AES key bytes one by one while the other bytes are fixed random values. From a security evaluation viewpoint, we want to simulate the attack scenario as realistic as possible. The key bytes cannot be changed and we cannot attack all the 2^{128} possible keys (for 16-byte AES key). We still want to evaluate how well we can correctly identify all 256 possible values of one key byte (consider it as an example of all 16 bytes), so normally we choose one example byte and vary it from $00x00$ to $0xFF$, but the other 15 bytes are fixed random values just like what the attacker will face for attacking a 16-byte key .

We use 256 classes of profiling traces to train the ResNet model, and 20% of profiling traces as validation data to improve the training. We then use the trained model in order



(a) S-box 8



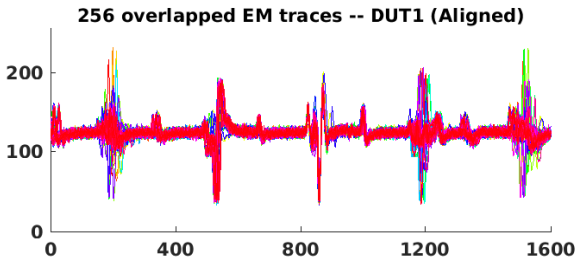
(b) S-box 16

Fig. 3: Performance comparison of 4 NN models using CWL data set.

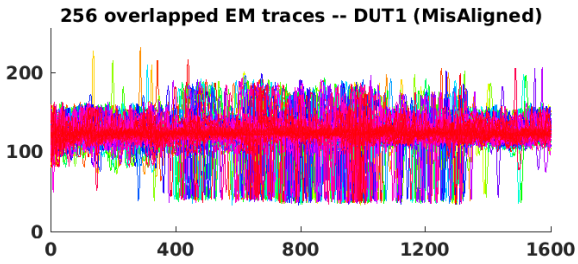
to classify all the 256 classes of attack traces to see how many classes can be correctly identified. From attack perspective, it is a profiling-based SPA attack. So we consider the percentage of correctly identified classes during the attack phase. For the DUT1 experiments, we use a batch size of 32 and 40 epochs, “Adadelta” optimizer with an initial learning rate of 1.0, adaptively reducing the learning rate with a factor of 10 if the validation loss is not decreased within 10 consecutive epochs.

For this target, in order to figure out the impact of misalignment with regards to the deep learning attacks, we conduct the deep learning attacks using the same ResNet model with two data sets: aligned traces (applying our synchronization method 6 times) and misaligned ones (applying our synchronization method 4 times). This impact of the alignment is visually illustrated by Figure 4.

We further calculate the SOST [7] trace per data set of DUT1 as shown in Figure 5 for better showing the impact of misalignment.

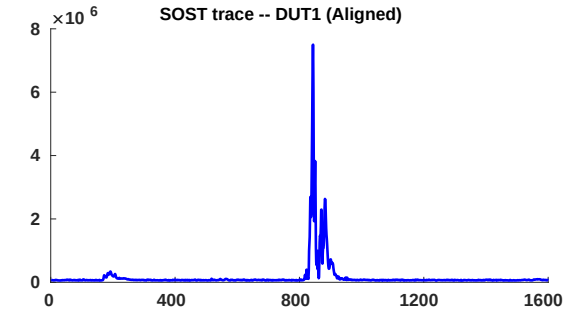


(a) Aligned

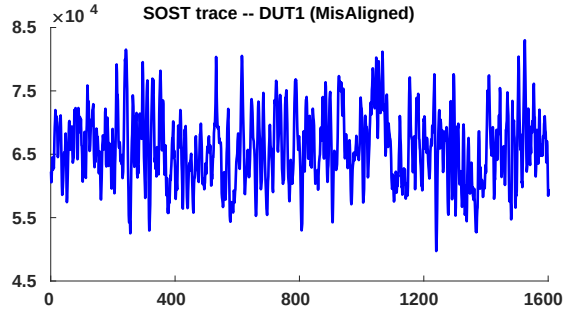


(b) MisAligned

Fig. 4: 256 overlapped aligned and misaligned EM traces of DUT1.



(a) Aligned



(b) MisAligned

Fig. 5: SOST traces of aligned and misaligned EM traces of DUT1.

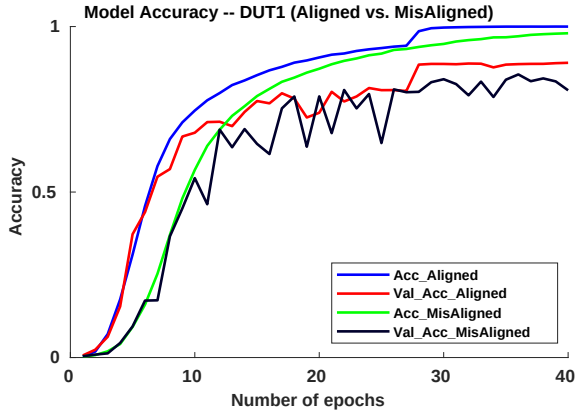


Fig. 6: ResNet Model Accuracy with DUT1 data sets.

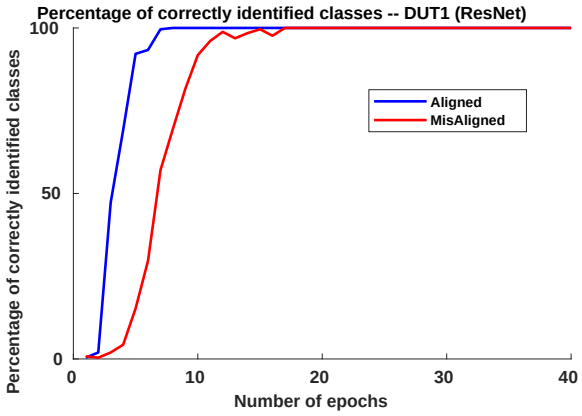
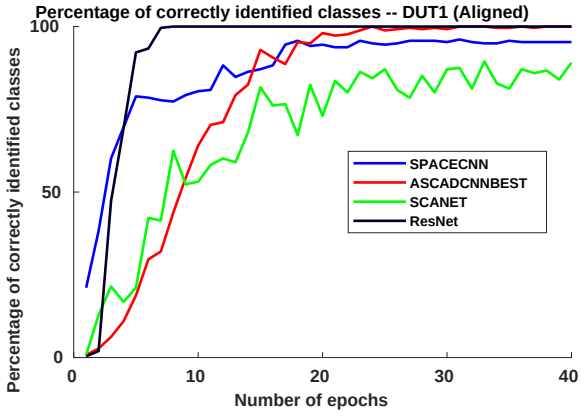
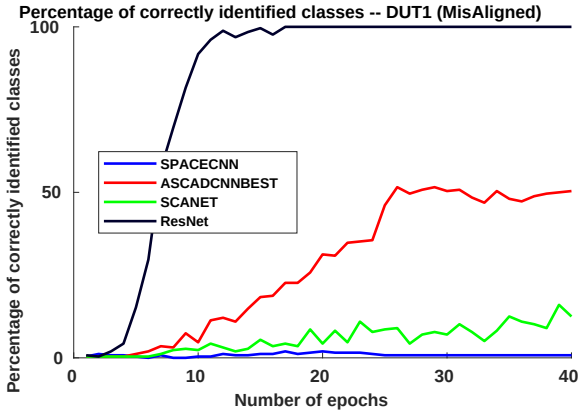


Fig. 7: Percentage of correctly identified classes with DUT1 data sets.

Both aligned and misaligned traces have 1,600 sample points to be fed into the ResNet model, and we are using the same number of profiling and attack traces and the same batch size for both cases. The training accuracy and validation accuracy (as described in Section 2.4) of the ResNet model are given in Figure 6. From the profiling perspective, the training accuracy using aligned traces is converging much faster and higher than the one using misaligned traces, although we are using the same ResNet model. Besides, from the attack perspective, as shown in Figure 7, the aligned traces also show much better percentage of correctly identified classes results. With aligned traces, the percentage of correctly identified classes already reaches 100% after 7 epochs and afterward the percentage of correctly identified classes is stabilized at 100%. On the other hand, with the misaligned traces, the percentage of correctly identified classes reaches 100% after 18 epochs.



(a) Using aligned traces.



(b) Using misaligned traces.

Fig. 8: Performance comparison of 4 NN models using DUT1 data sets.

4.1 Performance comparison with the state-of-the-art

Similarly, Figure 8 shows the percentage of correctly identified classes using our ResNet model and the other three models with both the aligned and misaligned DUT1 data sets. As can be seen for both aligned and misaligned traces, our ResNet model converges faster and gives better results from the attack perspective (less epochs of training are needed). In particular, considering the misaligned traces, our ResNet model outperforms the others a lot. It further confirms that the generalization of the ResNet model is can bing an interesting alternative to the other models.

5 Experimental results on DUT2 (AES S-box)

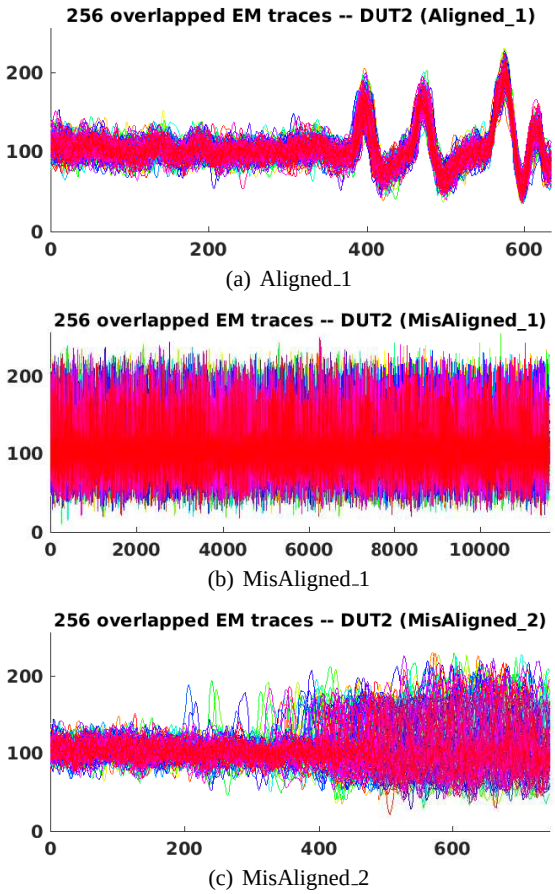
We now consider a more standard attack scenario where an adversary exploits the leakage of an S-box execution in the first round of the AES block cipher. In this case, for the profiling traces, we randomize all the 16 bytes of AES key and

AES input data to execute an AES encryption. And for the attack traces, we fixed a random 16-byte AES key and randomize the input data per trace. Similarly to the previous section, we train the ResNet model using 256 classes corresponding to the S-box outputs with the profiling traces, and we also use 20% profiling traces as validation data. For the DUT2 experiments, we use a batch size of 32 and 30 epochs, "Adadelta" optimizer with an initial learning rate of 1.0, adaptively reducing the learning rate with a factor of 10 if the validation loss is not decreased within 10 consecutive epochs.

To evaluate the impact of misalignment in this attack scenario, the same ResNet model is again used to perform deep learning attacks with aligned traces and misaligned ones, this time considering three different data sets. In the first one, next denoted as the *Aligned_1* data set (applying our synchronization method 9 times) and shown in the top graph of Figure 9, the traces are well aligned at the leakage part marked red and every trace has 634 sample points. The second data set (next denoted as the *misAligned_1* data set and shown in the middle graph of Figure 9) corresponds to the most misaligned traces. It is obtained by applying the alignment method only three times to the raw traces. In this case, to include the leakage part in the traces, every trace consists of 11,598 sample points. Finally, for the last data set (denoted as the *misAligned_2* data set and shown in the bottom graph of Figure 9), we apply only one alignment step less than what we do for the *Aligned_1* data set. As a result, the raw traces are aligned very close to the leakage part, but still the traces are not aligned at the leakage interval, and every trace contains 744 sample points.

To further demonstrate the impact of misalignment, we also calculate the SOST trace per data set of DUT2 as shown in Figure 10.

Based on this setup, first we launch the deep learning attacks using the *Aligned_1* data set. All the 16 S-boxes are getting similar results as depicted in the top graph of Figure 11. We then performed the same attacks as in the previous section against the *misAligned_1* data set. In this case, it took about one week per S-box for 30 epochs with a single NVIDIA GTX 1080Ti GPU card, hence highlighting the importance of alignment also for time complexity reasons. The attacks took about 4 hours per S-box for 30 epochs for the *Aligned_1* data set. Even after running it for a few weeks with different hyper-parameters (e.g., optimizer, number of residual blocks, number of filters, batch size, number of epochs, adding dropout layer [26] right before the last softmax layer in Figure 1), the rank of the correct subkey candidate remained very low and it was not possible to recover the subkeys. This result moderates the intuition that in a deep learning context, the best practice is to use the raw data without dimensional reduction, and the previous observation in [15, 22] that dimensional reduction methods lead to worse results in deep learning SCA context.



(a) Aligned_1

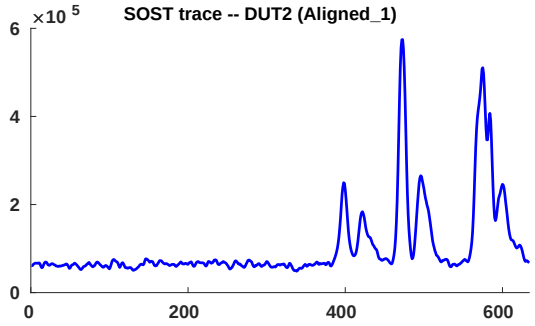
(b) MisAligned_1

(c) MisAligned_2

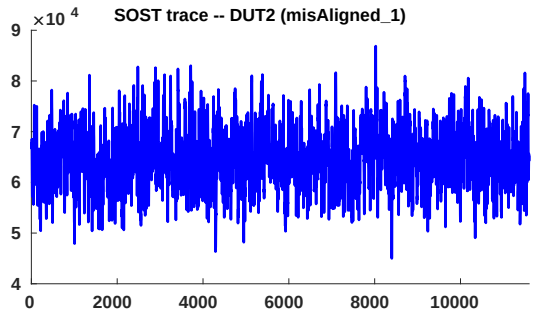
Fig. 9: 256 overlapped aligned and misaligned EM traces of DUT2.

Following, we further conduct the deep learning attacks on the *misAligned_2* data set. For this purpose, we first use the same ResNet model and the same amount of profiling traces as what we used in the *misAligned_1* data set case, which is 1,000 profiling traces per class of S-box output value including 20% of validation traces. In this case, the rank of the correct subkey remained stuck at high values. We next tweak the hyper-parameters of the ResNet model and still get similar results. Eventually, we increase the number of profiling traces to 1,200 and add a dropout layer (with a dropout factor of 0.3) right before the last softmax layer. To speed up the tests, we use dual NVIDIA GTX 1080Ti GPU cards. The best rank results are shown in the bottom graph of Figure 11. We also perform similar attacks (using the same ResNet model with the same amount of profiling traces and same dropout factor) on the other S-boxes using the same *misAligned_2* data set, this time with worse results.

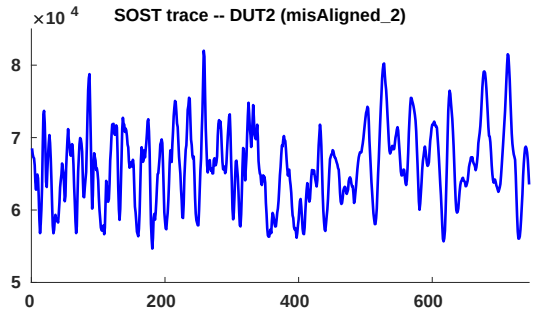
From the profiling perspective, no big difference could be observed between the training accuracy using aligned



(a) Aligned_1



(b) MisAligned_1

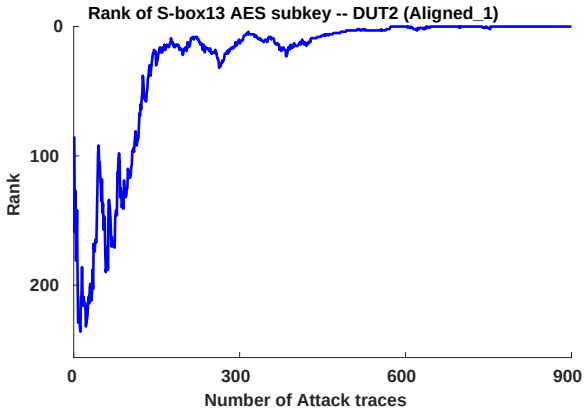


(c) MisAligned_2

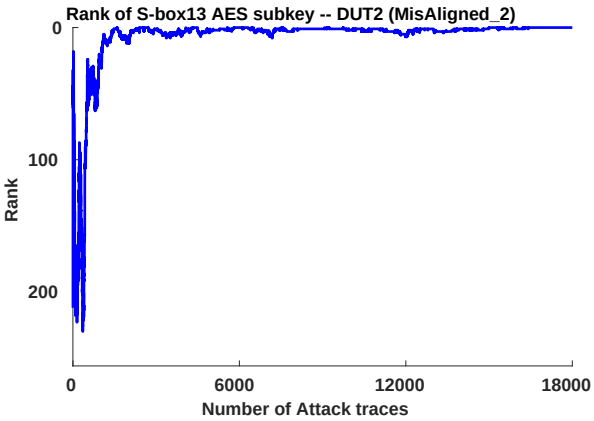
Fig. 10: SOST traces of aligned and misaligned EM traces of DUT2.

traces from the *Aligned_1* data set and the one using misaligned traces from the *misAligned_2* data set.

However, from the attack perspective, as can be seen from Figure 11, the aligned traces show much better key rank results. With aligned traces, the correct subkey is steadily ranked at the top position with more than 750 attack traces. The required number of attack traces for the misaligned case is about 16,294 (with more tweaks of the hyper-parameters and more profiling traces). We note again the larger ratio between the number of profiling and attack traces, that typically reflects a more challenging target.



(a) Using *Aligned_1* aligned traces.



(b) Using *misAligned_2* misaligned traces.

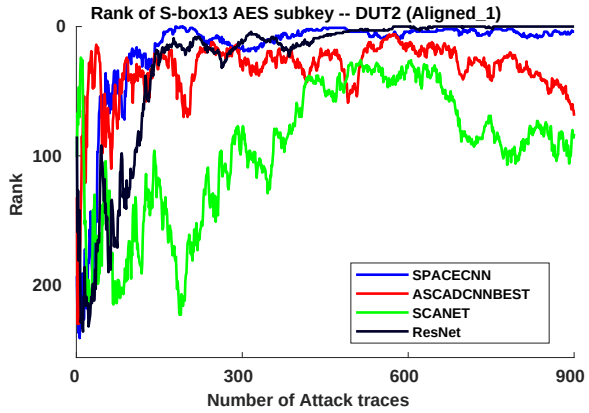
Fig. 11: Rank of correct subkey candidate with DUT2 data sets.

5.1 Performance comparison with the state-of-the-art

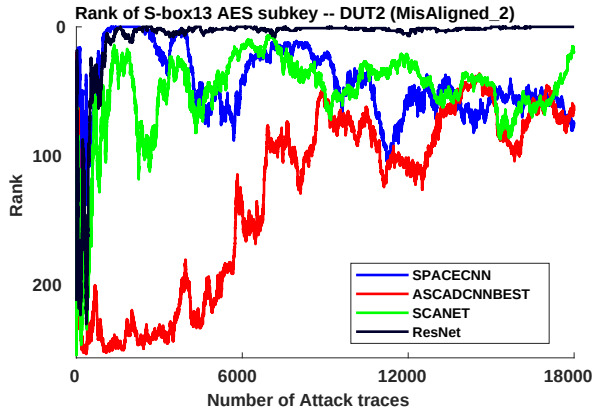
To further check the performance of our ResNet model, Figure 12 displays the rank of correct subkey candidates using our ResNet model and the other 3 models with both *Aligned_1* and *misAligned_2* data sets. For both aligned and misaligned traces, again, our ResNet model can recover the correct subkey byte faster than the others. Especially for the misaligned traces, our ResNet model outperforms the others a lot.

6 Comparison with ASCAD_CNN_Best model using ASCAD data set

To further demonstrate the generalization of our ResNet model, we compare the rank results of 16 S-boxes based on the published ASCAD data set using both the ResNet model and the published ASCAD_CNN_Best model. For both models, we use batch size of 100 and 200 epochs (the best choice for

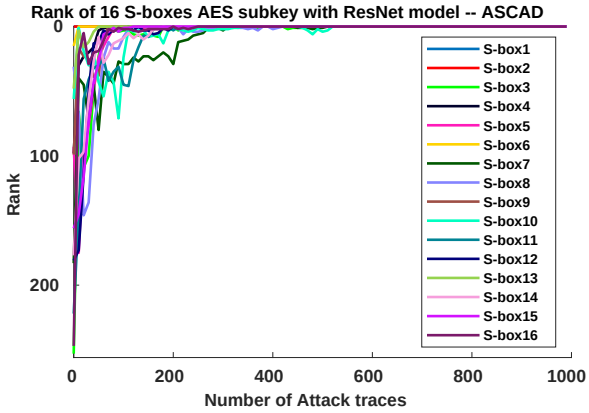


(a) Using aligned traces.

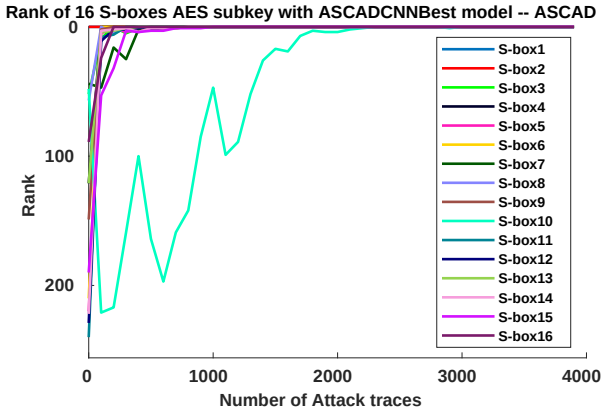


(b) Using misaligned traces.

Fig. 12: Performance comparison of 4 NN models using DUT2 data sets.



(a) Using ResNet model.



(b) Using ASCAD_CNN_Best model.

Fig. 13: Performance comparison of ResNet and ASCAD_CNN_Best models using ASCAD data set.

ASCAD_CNN_Best). For the ResNet model, we use "Adadelta" optimizer with an initial learning rate of 1.0, adaptively reducing the learning rate with a factor of 10 if the validation loss is not decreased within 10 consecutive epochs. For ASCAD_CNN_Best model, we use "RMSProp" optimizer with an initial learning rate of 10^{-5} and keep it fixed during the training (the best choice for ASCAD_CNN_Best).

Figure 13 shows the rank of correct subkey candidates of all 16 S-boxes using both models. For both models, all 16 subkey bytes can successfully be recovered using less than 4,000 traces. The first two S-boxes are not masked and the correct key candidates of them can be recovered with only 1 trace. For our ResNet model, all the subkey bytes can be recovered using less than 600 traces. For the ASCAD_CNN_Best model, 15 subkey bytes can be recovered using around 1,000 traces but S-box 10 requires 3,000 traces to get stable results. It further confirms the gppd generalization of our ResNet model.

7 Conclusions

Our results confirm that neural network models are powerful tools for black box leakage analysis and demonstrate the good generalization of the proposed ResNet model compared with the other state-of-the-art NN models in a side-channel context. Yet, even with such powerful tools, preprocessing the leakage traces with alignment/re-synchronization methods (and possibly filtering, ...) can be highly beneficial to the attacks/evaluations' success. This is true from the data complexity point-of-view, and data complexity is generally accounted for the most important complexity measure in side-channel analysis. But our second DUT shows that it is also true from the time complexity point-of-view. Indeed, alignment playing the role of data compressing due to much shorter interval being used then allows significantly reducing the number of samples in the traces to be fed to the ResNet, which may reduce its execution time from weeks to days (or even hours). The latter conclusions naturally become increasingly relevant for devices protected with a va-

riety of countermeasures. As stated in introduction, our conclusions are specific to the investigated datasets. In particular, deep ResNets only outperform other tested models in our case study (and are not claimed to be universally better, as popularized by the no-free-lunch theorem). We believe our concludion regarding the interest and sometimes need of preprocessing to be more general (since consistently observed with different models and datasets).

Acknowledgments. François-Xavier Standaert is a Senior Research Associate of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in parts by the EU through the ERC project SWORD (Consolidator Grant 724725) and the H2020 project REASSURE.

References

1. Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
2. Cédric Archambeau, Eric Peeters, François-Xavier Standaert, and Jean-Jacques Quisquater. Template attacks in principal subspaces. In *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*, pages 1–14, 2006.
3. Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing. In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 45–68, 2017.
4. Mathieu Carbone, Vincent Conin, Marie-Angela Cornelie, Francois Dassance, Guillaume Dufresne, Cecile Dumas, Emmanuel Prouff, and Alexandre Venelli. Deep learning to evaluate secure rsa implementations. Cryptology ePrint Archive, Report 2019/054, 2019. https://eprint.iacr.org/2019/054.
5. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.
6. François Chollet et al. Keras. https://github.com/fchollet/keras, 2015.
7. Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. Templates vs. stochastic methods. In *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*, pages 15–29, 2006.
8. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
9. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.
10. Benjamin Hettwer, Stefan Gehrer, and Tim Güneysu. Profiled power analysis attacks using convolutional neural networks with domain knowledge. In Carlos Cid and Michael J. Jacobson Jr., editors, *Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers*, volume 11349 of *Lecture Notes in Computer Science*, pages 479–498. Springer, 2018.
11. Annelie Heuser and Michael Zohner. Intelligent machine homicide - breaking cryptographic devices using support vector machines. In Werner Schindler and Sorin A. Huss, editors, *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings*, volume 7275 of *Lecture Notes in Computer Science*, pages 249–264. Springer, 2012.
12. Gabriel Hospodar, Benedikt Gierlichs, Elke De Mulder, Ingrid Verbauwhede, and Joos Vandewalle. Machine learning in side-channel analysis: a first study. *J. Cryptographic Engineering*, 1(4):293–302, 2011.
13. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.
14. Liran Lerman, Romain Poussier, Gianluca Bontempi, Olivier Markowitch, and François-Xavier Standaert. Template attacks vs. machine learning revisited (and the curse of dimensionality in side-channel analysis). In Stefan Mangard and Axel Y. Poschmann, editors, *Constructive Side-Channel Analysis and Secure Design - 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers*, volume 9064 of *Lecture Notes in Computer Science*, pages 20–33. Springer, 2015.
15. Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In Claude Carlet, M. Anwar Hasan, and Vishal Saraswat, editors, *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*, volume 10076 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2016.
16. Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814. Omnipress, 2010.
17. Christophe Pfeifer and Patrick Haddad. Spread: a new layer for profiled deep-learning side-channel attacks. Cryptology ePrint Archive, Report 2018/880, 2018. https://eprint.iacr.org/2018/880.
18. Stjepan Picek, Annelie Heuser, Alan Jovic, Shivam Bhasin, and Francesco Regazzoni. The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(1):209–237, 2019.
19. Stjepan Picek, Annelie Heuser, Alan Jovic, and Axel Legay. Climbing down the hierarchy: Hierarchical classification for machine learning side-channel attacks. In Marc Joye and Abderrahmane Nitaj, editors, *Progress in Cryptology - AFRICACRYPT 2017 - 9th International Conference on Cryptology in Africa, Dakar, Senegal, May 24-26, 2017, Proceedings*, volume 10239 of *Lecture Notes in Computer Science*, pages 61–78, 2017.
20. Stjepan Picek, Ioannis Petros Samiotis, Jaehun Kim, Annelie Heuser, Shivam Bhasin, and Axel Legay. *On the Performance of Convolutional Neural Networks for Side-Channel Analysis: 8th*

International Conference, SPACE 2018, Kanpur, India, December 15-19, 2018, Proceedings, pages 157–176. 12 2018.
21. Romain Poussier, Yuanyuan Zhou, and François-Xavier Standaert. A systematic approach to the side-channel analysis of ECC implementations with worst-case horizontal attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 534–554, 2017.
22. Emmanuel Prouff, Remi Strullu, Ryad Benadjila, Eleonora Cagli, and Cecile Dumas. Study of deep learning techniques for side-channel analysis and introduction to ascad database. Cryptology ePrint Archive, Report 2018/053, 2018. https://eprint.iacr.org/2018/053.
23. Pieter Robyns, Peter Quax, and Wim Lamotte. Improving CEMA using correlation optimization. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(1):1–24, 2019.
24. Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.
25. David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354 EP –, Oct 2017. Article.
26. Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
27. François-Xavier Standaert, François Koeune, and Werner Schindler. How to compare profiled side-channel attacks? In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *Applied Cryptography and Network Security, 7th International Conference, ACNS 2009, Paris-Rocquencourt, France, June 2-5, 2009. Proceedings*, volume 5536 of *Lecture Notes in Computer Science*, pages 485–498, 2009.
28. François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.
29. Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.
30. Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.
31. Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. The microsoft 2016 conversational speech recognition system. *CoRR*, abs/1609.03528, 2016.