Vulnerability Testing on the Key Scheduling Algorithm of PRESENT Using Deep Learning

Ming Duan[1,2], Rui Zhou[1,2(✉)], Chaohui Fu[1], Sheng Guo[1], and Qianqiong Wu[1]

[1] Information Engineering University, Zhengzhou, China
[2] Henan Key Laboratory of Network Cryptography, Zhengzhou, China

Abstract. PRESENT is a lightweight block cipher developed for extremely constrained environment such as RFID tags and IoT (Internet of Things). In 2020, Pareek et al. suggested a neural network to retrieving the 80-bit key of block cipher PRESENT from the last round subkeys and they arrived at a conclusion that the key scheduling algorithm is strong enough against its neural aided attack. While in this paper, we get a contradict result. We present two different experiments to test the vulnerability of key scheduling algorithm using deep learning. First, we build a 3-depth Fully Connected Neural Network to retrieve the master key bit by bit. The result is that we can predict about 80% bits of a random key with very high accuracy (above 0.9). Furthermore, we train a Residual Neural Network for classification. Compared with Fully Connected Network, we need less networks and the success rate is 100% . Finally, we combine the two networks to retrieve the whole 80-bit key from the last 64-bit round subkey. We think this method can be applied to the key schedule of other block ciphers or other similar cryptographic processes.

Keywords: Key scheduling algorithm · Deep learning · PRESENT

1 Introduction

PRESENT is an ultra-lightweight block cipher. There have been several attacks conducted to test its security but few researches have been done on the key scheduling algorithm (KSA) of PRESENT. A truncated differential attack was suggested to 26 rounds in 2014 [1] and several biclique cryptanalysis were done on the full round of 80-bit PRESENT cipher in [2] and [3]. As to the KSA, [4] showcases three factors to judge its strength.

Possible connections between cryptography and machine learning have been proposed since the neural network technology started flourishing [5]. It was not until 2019, Aron Gohr proposed a neural network based differential distinguisher for round-reduced Speck in [6] that first used deep learning in black-box cryptanalysis. Speck is also a lightweight block ciphers like PRESENT. And it shows a good transformation capability in many aspects [7–9]. In 2020, it inspired Pareek, Kohli and Mishra to apply the deep learning technique to the key scheduling process of PRESENT but they failed to

W. Shi et al. (Eds.): SPNCE 2021, LNICST 423, pp. 307–318, 2022.
https://doi.org/10.1007/978-3-030-96791-8_23

learn any features [10]. The failure does not mean the KSA is absolutely secure against deep learning attack, especially when we successfully find some features.

Contribution: In this paper, we use two types of neural networks to retrieve the key of PRESENT. First, we use Fully Connected Network to fit the key schedule process. It predicts 80% of master key at the accuracy above 0.9 bit by bit. For each bit, the offline data to train the model is $2^{15.97}$. Meanwhile, the bit positions that cannot be retrieved illustrate that this neural network fails to fit the S-box function in KSA. Next, we change our goal into solving a classification problem. We use Residual Neural Networks and improve the results, that we successfully predict n bits at one time under certain conditions. And the offline data that we use is $2^{13.77}$. Finally, we use both neural networks to finish the key recovery process and the minimum online data complexity is 2^6, when $n = 2$. This can be carried out within seconds on a modern computer. For brute force attack, it needs 2^{16} data and costs 2^{16} times PRESENT inverse KSA online theoretically. As a result, the deep learning method greatly surpasses the brute force method on online computation.

It is to be pointed out that by using more training/testing data, the accuracy will be improved. As the deep learning model can be pre-trained, it will not cost more online computation time.

Organization: The rest of the paper is divided into five parts. In Sect. 2, we introduce the key scheduling algorithm of the cipher PRESENT. Followed by a brief introduction of two neural network we used in this paper: Fully Connected Neural Network (or Multi-Layer Perception) and Residual Neural Network in Sect. 3. Section 4 presents our approach to retrieve the key using Fully Connected Neural Network, while Sect. 5 uses Residual Neural Network to identify n bits at a time and proposes our key recovery method. We conclude the paper and give probable direction of further study in the future in last section.

2 PRESENT Block Cipher

2.1 Encryption Process

PRESENT is an iterated block cipher proposed by Bogdanov, Knudsen, Leander, Paar, Poschmann, Robshaw, Seurin and Vikkelsoe in 2007 [11]. It employs an S/P (substitution and permutation) structure. The block length is 64 bits. This algorithm includes 31 rounds of encryption using key of 80 and 128 bits, and we only discuss about the 80-bit key version here.

In each round, the 64-bit input state exclusive-ores (XOR) the round key and is passed through 16 parallelly applied S-boxes and one permutation layer. S-box used in this cipher is a 4-bit to 4-bit mapping as a non-linear substitution. A block diagram showing the structure of PRESENT is given in Fig. 1.

2.2 Key Scheduling Algorithm

PRESENT cipher has two versions using the key size of 80-bit and 128-bit. In this paper, the former version is considered.

First, the initial 80-bit key is loaded in the key register, which is represented as $K_{79}K_{78}\ldots K_0$. For each round (i), the key register is rotated by 61-bit positions to the left, the left-most four bits are passed through the S-box, and bits $K_{19}K_{18}K_{17}K_{16}K_{15}$ of K is XORed with the *round_counter* value i on the right. After this the left-most 64-bits of the current key register are taken as the round key for that round (i). Figure 2 shows the key updating for 80-bit PRESENT variation.

Although PRESENT has 31-round encryption, the initial key needs to update 32 iterations and produce 32 round keys in one encryption.

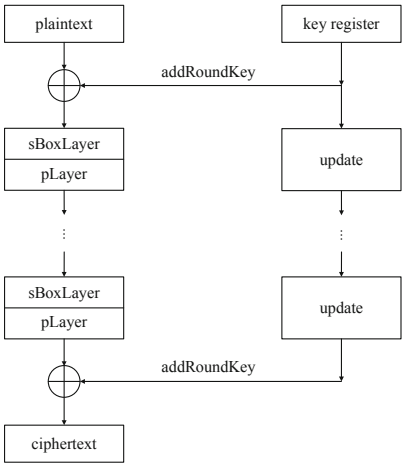

Fig. 1. Structure of PRESENT

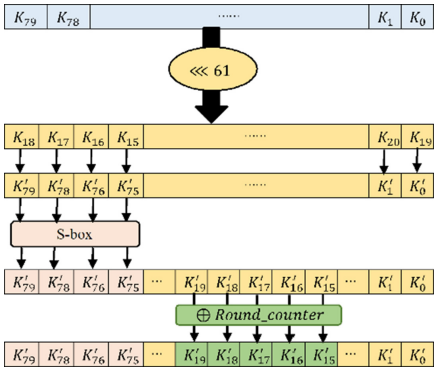

Fig. 2. Key updating in PRESENT

3 Convolutional Neural Network

Deep learning can be traced back to 40s, 50s of last century, which is a comprehensive field developing from biological science. Deep Learning is a more sophisticated format of Machine Learning, using artificial neural networks. Here in this paper, we solve the problem (to retrieve the seed key) using two types of neural networks and try to find how the structure of the network impacts the result.

3.1 Deep Learning Neural Network

Deep Learning technique need to create a Deep Neural Network (DNN) and train that DNN model on the provided dataset. MP model (proposed by McCulloch and Pitts) is a basic structure of a DNN [12]. It includes an input layer, a middle layer and an output layer. The input layer has several variables and a bias. The middle layer is only one neuron to do a sum of the inputs (as a linear operation) and one non-linear function (SIGMOID, ReLU, etc.). In the end it outputs one value. The basic structure is shown in Fig. 3.

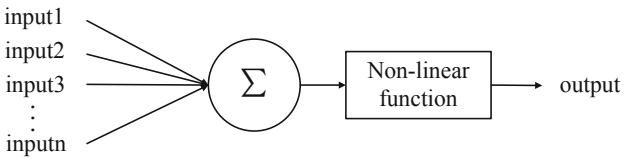

Fig. 3. MP model

The simplest and most basic DNN is "Fully Connected Neural Network", more often called Multi-Layer Perceptron (MLP). It has more than one middle layer so that we call it "deep". Those middle layers are called "hidden layers". For every neuron in a layer, it is connected with all the neurons forward, which is described as "fully connected". As a result, in a Fully Connected Neural Network, every output unit interacts with every input unit.

3.2 Residual Neural Network

Compared to the Fully Connected Neural Network, neurons in the Convolutional Neural Network (CNN) partially interact with each other which is called "sparse interactions" [12]. That is to say, all the input units are not connected with the output unit in latter layer.

The basic layer of CNN includes three stages: first, a convolutional stage which performs several convolutions in parallel; second, a detector stage using nonlinear activation function and, in this paper, we use the rectified function. Last is the pooling stage to modify the output of the layer.

Residual Neural Network (ResNet) we use in our experiment is one of the most commonly used CNNs. A basic unit in ResNet is a residual block. The output of a residual block can be described as a linear addition: $F(x) + x$, where x is input. That gives a connection between the input and output in a residual block, which is called skip connect. Figure 4 shows the structure clearly. It solves the vanishing gradients problem when a network gets deeper and deeper.

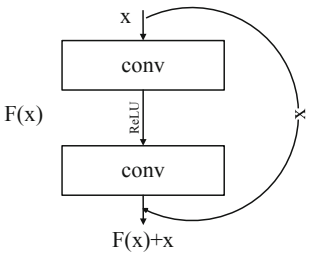

Fig. 4. Residual block

4 Key Recovery Using MLP

Pareek et al. tried to use Fully Connected Neural Network to retrieve the main key register from the final round key register of PRESENT, see [10]. That is, using 80 networks of same structure to predict each bit of the 80-bit-long master key separately. The result is that the prediction accuracy for a bit position of the master key is very close to 0.5, which means it failed to retrieve the master key. In my experiment, we first use the same type of neural network (MLP) and find that it is able to retrieve some specific bit positions of PRESENT master key. Next, we test different parameters and find that the result maintains its features.

4.1 Data Generation

We generate an 80-bit key randomly and then pass it through the key scheduling algorithm of PRESENT to obtain the final round 64-bit key as input data. The train dataset is composed of 600,00 samples and the number of samples in test data is 40,00.

4.2 Neural Network Structure

Input layer contains 64 neurons, each neuron for the individual bit of the final round key. 3 densely connected hidden layers of 32 neurons, 16 neurons, 8 neurons respectively each activated by the Rectified Linear Unit (ReLU) function. Output layer consists one neuron for the predicted bit (0 or 1), and is activated by the SIGMOID activation function. Figure 5 shows the structure of this Multi-Layer Perception.

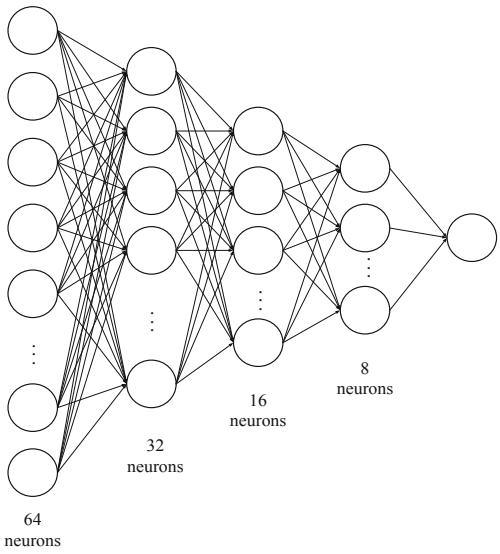

Fig. 5. MLP structure

4.3 Results

We use the batch size of 200. Take the 38^{th} bit of master key as an example in Fig. 6.

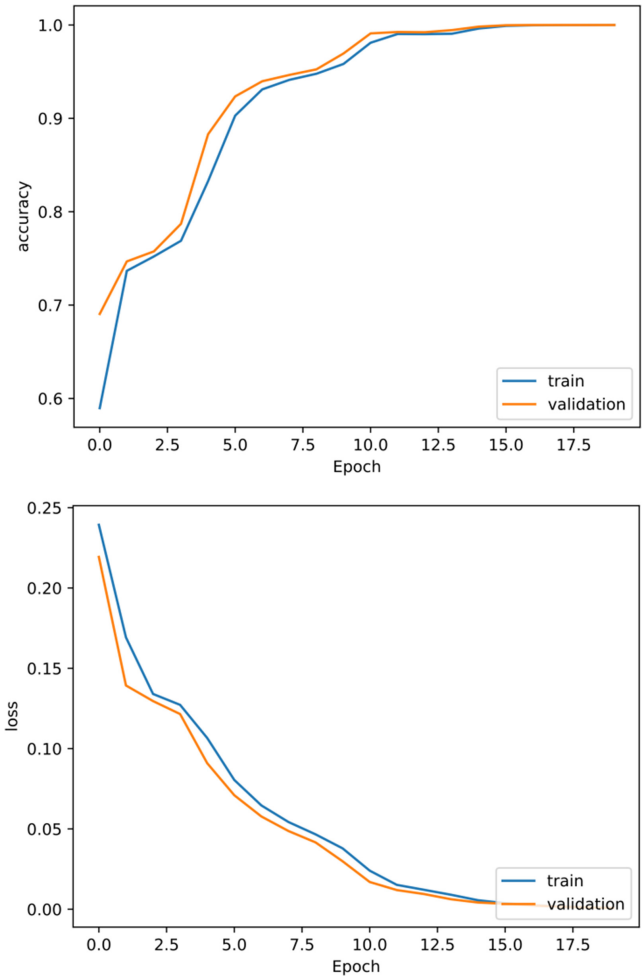

Fig. 6. The accuracy and loss rate of 38^{th} bit position change with epoch

Unlike the result given by Mann [10], we find the prediction accuracies on several bits are good enough and even reach 1.0. And it is not the upper bound of the accuracy we can reach because when we add our training data to 10^5, the accuracy rises for around 0.01. However, on some bits the accuracies of both test and train set are around 0.5, which means this network structure is not suitable for predicting these bits of the key because of underfitting. Several keys are chosen randomly and the results are similar. The best validation accuracy is shown in the following Table 1.

Table 1. Test accuracy of i^{th} bit

i^{th} bit	Accuracy	i^{th} bit	Accuracy	i^{th} bit	Accuracy	i^{th} bit	Accuracy
80	1.0000	60	0.4971	40	1.0000	20	1.0000
79	1.0000	59	0.4996	39	1.0000	19	0.9965
78	1.0000	58	0.4998	38	0.9804	18	0.9855
77	1.0000	57	0.4979	37	0.9840	17	0.9920
76	0.9805	56	0.4995	36	0.9958	16	0.9797
75	0.9843	55	0.4958	35	0.9402	15	0.9920
74	0.9657	54	0.4980	34	0.9760	14	0.9927
73	0.9803	53	0.4979	33	0.9998	13	0.9902
72	0.9860	52	0.4977	32	0.9967	12	0.9312
71	0.9908	51	0.4979	31	0.9905	11	0.9862
70	0.9955	50	0.5001	30	0.9905	10	0.9747
69	0.9910	49	0.6168	29	0.9737	9	0.9910
68	1.0000	48	0.7448	28	0.9870	8	0.9877
67	0.9768	47	0.6238	27	0.9530	7	0.9755
66	0.9902	46	0.7445	26	0.9845	6	0.9567
65	0.9912	45	0.9937	25	0.9998	5	0.9818
64	0.5046	44	1.0000	24	0.9937	4	0.9902
63	0.5005	43	0.9925	23	1.0000	3	1.0000
62	0.5045	42	0.9950	22	1.0000	2	1.0000
61	0.4981	41	1.0000	21	1.0000	1	1.0000

It does not matter when we change the one-bit output into one hot vector and the final activation function into SOFTMAX as a match. The basic neural network structure is the same with Mann but we get 80% of the master key retrievable by this fully connected network. Although we both use the same network structure, there could still exist some details that are different.

The result also shows that there are some features about the retrievable positions:

(1) The best validation accuracy of each bit is above 0.9, except (2).
(2) The 49^{th} bit to the 64^{th} bit can be hardly predicted (with accuracy around 0.5).

We think it results from the truncation operation in the KSA that every iteration, we truncate the 80-bit key in register to produce a 64-bit round key. If we ignore the S-box function in KSA of PRESENT, the 49^{th} to 64^{th} bit in master key is just the dropped-out bits from the 80-bit key register and has no relationship with the final 64-bit round key. To prove our guess, we repeat the experiment on reduced-round KSA, and the result

is the same: only the dropped out 16 bits cannot be predicted by the MLP. Further, the conclusion is that the MLP cannot fit S-box function (or substitution).

5 Key Recovery Using ResNet

In order to learn the feature of S-box in the KSA, we explore a new method. As the output of the previous neural network is discrete (0 or 1), we attempt to use a classification network to replace it. Our idea is to feed the neural network with both real master keys and random data and train the network to distinguish between the two. We choose a Residual Neural Network because it has been used to build a distinguisher already [6] and its residual structure makes sure we can add more layers to the network without vanishing gradients problem.

5.1 Data Generation

Instead of predicting one bit with one network, we choose n bit from the master key to train at a time. The input data is divided into two parts: one is labeled as 1. The front of one sample is the final round key generated through key scheduling algorithm of PRESENT and the rear is n-bit data chosen from n positions of the master key. Another labeled as 0 is $(64 + n)$-bit random data. Both are stored in the form of an array. We use 10^4 training data and 4×10^3 testing data.

5.2 ResNet Structure

The input layer contains $(64 + n)$ neurons as to be consistent with the $(64 + n)$-bit input data. The hidden layers are residual network: the first layer is a convolutional layer for 1 dimension. Batch normalization is applied to its output. Following is a depth-5 residual tower with 2 layers of residual network each block. The size of the convolutional kernels is 3. Other settings are the same as above. Last two densely connected layers has 64 neurons each and output in one neuron. The activation function for the hidden layers is ReLU and the output is activated by SIGMOID (Fig. 7).

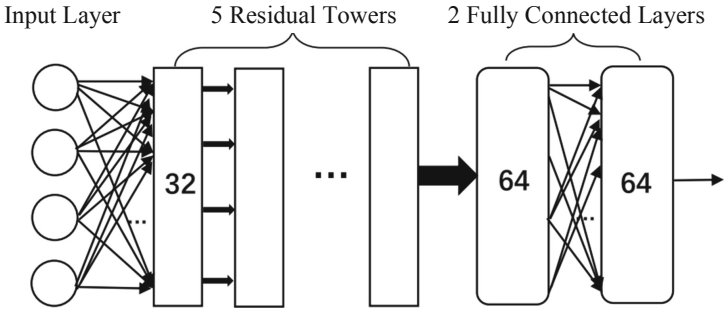

Fig. 7. Structure of Residual Neural Network

5.3 Results

We test every 16 bits ($n = 16$) in order (i.e., 8 groups). Both the accuracy of test and train set are above 0.9 for the 16 positions in sequence. That illustrates that the whole KSA of PRESENT is learnable by residual network. We can see from Fig. 8 that the neural network can reach a high accuracy at every beginning (take the last 16 bits as an example).

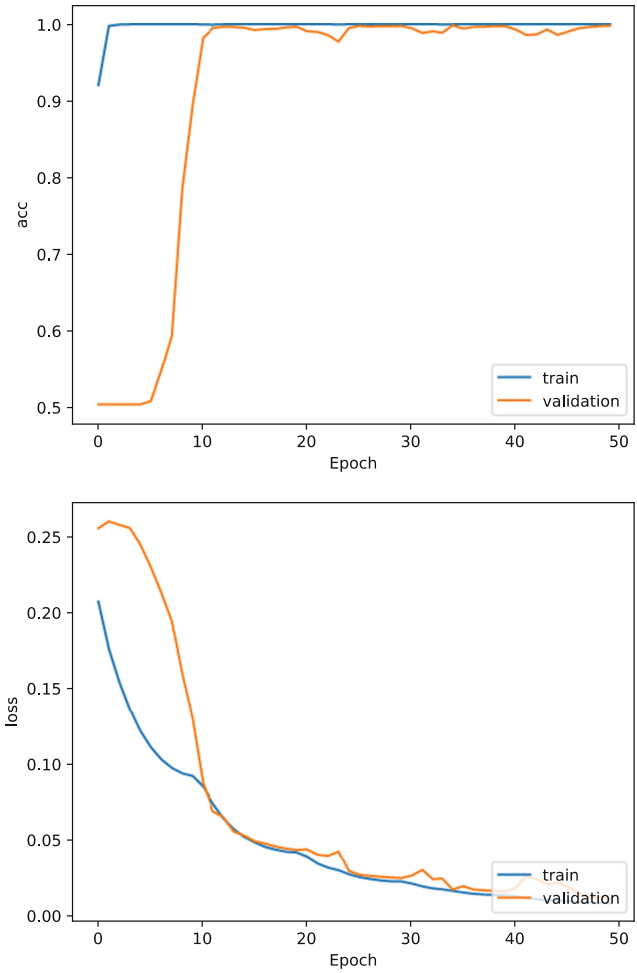

Fig. 8. The accuracy and loss rate of the last 16 bits

We specially pick those bit positions that failed to be retrieved by the previous Fully Connected Neural Network to be tested in the new network. Group A: From the 49^{th} bit to the 64^{th} bit position, among which only the 49^{th} bit was successfully predicted. The Residual Neural Network works out and gets a prediction accuracy just as high as other bit positions (Table 2).

Table 2. Test accuracy of i^{th} bit (n = 16).

Group	Bit position	Accuracy
1	1, 2, 3…, 14, 16	1.0000
2	17, 18, 19…, 32	1.0000
3	33, 34, 35…, 48	0.9998
4	49, 50, 51…, 64	0.9999
5	65, 67, 68…, 80	0.9655
6	1, 2, 3…, 14, 16	0.9998
7	17, 18, 19…, 32	0.9999
8	33, 34, 35…, 48	0.9655
A	16, 17, …, 31	0.9997

Let $n = 1$, the ResNet will predict bit by bit just like what we do in Sect. 3. And the result is the same, too: the accuracy from 49^{th} to 64^{th} is still around 0.5.

Let $n = 2$, if we choose both bits between 49^{th} to 64^{th}, it still fails. However, if we choose one bit between 49^{th} to 64^{th}, and one bit outside (i.e., one bit is dropped out from last round while another is not), the accuracy rate rises to above 0.7.

Similarly, if we want to predict these dropped-out bit successfully, we need to include at least one bit that is included in the last-round truncated key.

The result shows that the classification network can learn the relation between the truncated bits and the others, which means it can learn the feature of the S-box function (Table 3).

Table 3. Test accuracy of i^{th} bit (n = 2).

Group	Bit position	Accuracy
1	49, 50	0.5795
2	49, 48	0.7237
3	49, 1	0.7753
4	17, 1	0.8163

As this method cannot retrieve the master key one bit at a time, but just identify whether the n bits are from the real master key or not, we need one more step to search each bit.

In practical use, to retrieve the master key from the 32^{nd} round key. We can use MLP to retrieve the 1^{st} to 48^{th} and 65^{th} to 80^{th} bits, and identify the remaining 16 bits using ResNet with n bits at a time (including 1 bit that has been retrieved by the MLP already). Then we need to go through 2^{n-1} to retrieve each bit. The search complexity is $2^n \times \frac{16}{n-1}$. To minimize the complexity, we choose $n = 2 \, or \, 3$, and the search complexity is 2^6.

6 Conclusion

In summary, the Fully Connected Network can retrieve partially. The Residual Neural Network can identify the real from random. We combine the two neural networks to retrieve the total 80-bit master key with online data complexity of 2^6. Our method is also efficient that the pre-training processes can be carried out within 10 min and the online retrieving cost several seconds. But we get the online efficiency at the cost of more offline data. This drawback is also one significant feature of deep learning—relying on big data.

In terms of the vulnerability of KSA from deep learning perspective, we use Fully Connected Network to fit and Residual Neural Network to classify, and the result is that MLP fails to fit the S-box function but the classification network can identify it. We can use the features we found to dig out more in the future. And it is likely that we apply this method to similar cryptographic processes.

Acknowledgments. This work was supported by General Projects of Henan Natural Science Foundation (No. 212300410420).

References

1. Blondeau, C., Nyberg, K.: Links between truncated differential and multidimensional linear properties of block ciphers and underlying attack complexities. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 165–182. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_10
2. Lee, C.: Biclique cryptanalysis of PRESENT-80 and PRESENT-128. J. Supercomput. **70**(1), 95–103 (2014)
3. Sereshgi, M.H.F., Dakhilalian, M., Shakiba, M.: Biclique cryptanalysis of MIBS-80 and PRESENT-80 block ciphers. Secur. Commun. Netw. **1**(9), 27–33 (2016)
4. Hernandez-Castro, J.C., Peris-Lopez, P., Aumasson, J.-P.: On the key schedule strength of PRESENT. In: Garcia-Alfaro, J., Navarro-Arribas, G., Cuppens-Boulahia, N., de Capitani di Vimercati, S. (eds.) DPM/SETOP -2011. LNCS, vol. 7122, pp. 253–263. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28879-1_17
5. Rivest, R.L.: Cryptography and machine learning. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 427–439. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-57332-1_36
6. Gohr, A.: Improving attacks on round-reduced speck32/64 using deep learning. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 150–179. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_6
7. Su, H.-C., Zhu, X.-Y., Ming, D.: Polytopic attack on round-reduced Simon32/64 using deep learning. In: Wu, Y., Yung, M. (eds.) Inscrypt 2020. LNCS, vol. 12612, pp. 3–20. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-71852-7_1
8. Fu, C.H., Duan, M., Wei, Q., Wu, Q.Q., Zhou, R., Su, H.C.: Polytopic differential attack based on deep learning and its application. J. Cryptol. Res. **8**(4), 591–600 (2021)
9. Baski, A., Breier, J., Chen, Y., Dong, X.Y.: Machine learning assisted differential distinguishers for lightweight ciphers. In: 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 176–181. IEEE, Grenoble (2021)

10. Pareek, M., Kohli, V., Mishra, G.: Deep learning based analysis of the key scheduling algorithm of PRESENT cipher. https://eprint.iacr.org/2020/981. Accessed 11 May 2021
11. Bogdanov, A., et al.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74735-2_31
12. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**, 115–133 (1943)
13. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, pp. 326–352. MIT Press, Cambridge (2016)