Improving Differential-Neural Cryptanalysis with Inception

Liu Zhang[1][0000−0001−6106−3767], Zilong Wang[1][0000−0002−1525−3356], Baocang Wang[2][0000−0002−2554−4464], and Boyang Wang[3][0000−0001−8973−2328]

[1] School of Cyber Engineering, Xidian University, Xi'an, China
liuzhang@stu.xidian.edu.cn, zlwang@xidian.edu.cn
[2] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China bcwang79@aliyun.com
[3] Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, USA
boyang.wang@uc.edu

Abstract. In CRYPTO'19, Gohr proposed a new cryptanalysis method by building differential-neural distinguishers with neural networks. Gohr combined a differential-neural distinguisher with a classical differential path and achieved a 12-round (out of 22) key recovery attack on Speck32/64. Chen and Yu improved the accuracy of differential-neural distinguisher considering derived features from multiple-ciphertext pairs. Bao *et al.* enhanced the classical differential path by generalizing the concept of neutral bits, thus launching key recovery attacks for 13-round Speck32/64 and 16-round (out of 32) Simon32/64.

Our focus is on improving the accuracy of the distinguisher and training the distinguisher for more rounds using deep learning methods. To capture more dimensional information, we use multiple parallel convolutional layers with kernels of different sizes placed in front of the Residual Network to train differential-neural distinguisher inspired by the Inception in GoogLeNet. For Speck32/64, we obtain a 9-round differential-neural distinguisher and significantly improve the accuracy of distinguishers on 6-,7-, 8-round. For Simon32/64, we obtain a 12-round differential-neural distinguisher and significantly improve the accuracy of the distinguishers on 9-, 10-, and 11-round. In addition, we use neutral bits to solve the same distribution of data required to successfully launch a key recovery attack when using multiple-ciphertext pairs as the input of the neural network.

Under the combined effect of multiple improvements, the time complexity of our 11-, 12-, and 13-round key recovery attacks of Speck32/64 is decreased. Also, the success rate of our 12-round key recovery attack reaches 100% in 98 trials. For Simon32/64, we are able to implement a 17-round key recovery attack using the deep learning method for the first time. Also, we decrease the time complexity of the 16-round key recovery attack. [4]

[4] The source codes are available in https://github.com/CryptAnalystDesigner/NeuralDistingsuisherWithInception.git.

Keywords: Differential-Neural Distinguisher · Inception · SPECK · SIMON

1 Introduction

In CRYPTO 2019, Gohr proposed the idea of differential-neural cryptanalysis [8]. The differential-neural distinguisher model, a trained neural network, is introduced as the underlying distinguisher, and Bayesian search is used to speed up key recovery attacks compared to the classical differential cryptanalysis. The differential-neural distinguisher is to distinguish ciphertext from random numbers. If the accuracy of the differential-neural distinguisher is greater than 0.5, it is considered as an effective distinguisher. However, the current differential-neural distinguisher seems only effective for limited rounds for ciphertext. In order to increase the number of rounds for key recovery attacks, a short classical high-probability differential path $\Delta S \rightarrow \Delta P$ is prepended before the differential-neural distinguisher.

Gohr [8] showed that the Residual Network [9] could be trained to capture the non-randomness of the distribution of output pairs when the input pairs of round-reduced SPECK32/64 meet a specific difference. As a result, differential-neural distinguishers on 6-, 7-, 8-round were trained, and 11-,12-round key recovery attacks for SPECK32/64 were achieved by combining 2-round differential path. There may be two directions to improve the differential-neural cryptanalysis proposed by Gohr [8]. One is to increase the length of the differential path suspended on top of the differential-neural distinguisher. Bao *et al.* [2] generalized the concept of neutral bits and searched for (conditional) simultaneous neutral bit-set with a higher probability for more rounds of the differential path. Thus, Bao *et al.* devised a new 13-round and improved the 12-round key recovery attack for SPECK32/64 with the same differential-neural distinguisher proposed in [8]. Note that the improvement in [2] came from the classical differential cryptanalysis. The other is to study the effective differential-neural distinguisher with more rounds. To launch key recovery attacks with more rounds, better differential-neural distinguishers were also studied recently. Chen and Yu [6] proposed multiple-ciphertext pairs instead of single-ciphertext pairs (in Gohr's work) as the input of the neural network and improved the accuracy of the 6-, 7-round differential-neural distinguisher of SPECK32/64. Bao *et al.* [2] used Dense Network [11] and Squeeze-and-Excitation Network [10] with existing deep architectures to train differential-neural distinguisher, and obtained 9-, 10-, 11-round differential-neural distinguisher and devised a 16-round key recovery attack for SIMON32/64.

In EUROCRYPT 2021, Benamira [4] indicated that Gohr's differential-neural distinguisher builds a good approximation of the differential distribution table of the cipher during the learning phase and learns additional information. Based on the principle that the purpose of the differential-neural distinguisher is to obtain more information from the ciphertext, we train a better differential-neural

distinguisher by modifying the network structure. The main improvements for differential-neural cryptanalysis are listed as follows.

First, we trained a better differential-neural distinguisher. Specifically, we used multiple-ciphertext pairs as the input of the neural network and added the Inception composed of the multiple-parallel convolutional layers before the Residual Network. The purpose of the Inception is to capture more dimensional information in the ciphertext pairs. Some minor modifications have also been made according to the round functions of the cipher. As a result, we improved the accuracy of 6-, 7-, 8-round and trained an effective 9-round differential-neural distinguisher for SPECK32/64. In addition, we improved the accuracy of 9-, 10-, and 11-round distinguishers and trained an effective 12-round differential-neural distinguisher for SIMON32/64. The result and comparison on differential-neural distinguisher of SPECK32/64 and SIMON32/64 are presented in Table 1 and 2, where N is the number of the training instances and m is the number of the ciphertext pairs in every instance.

Table 1. Summary of distinguish accuracy on SPECK32/64 using N instances

R	Gohr [8]	m=2		m=4		m=8		m=16	
		Chen [6]	Sect. 3	Chen [6]	Sect. 3	Chen [6]	Sect. 3	Chen [6]	Sect. 3
6	0.788	0.8613	0.8773	0.931	0.9497	0.9562	0.9895	0.9802	0.9992
7	0.614	0.6393	0.6649	0.6861	0.7283	0.7074	0.8106	0.6694	0.8963
8	0.514	-	-	-	0.5428	-	0.5562	-	0.5853
9	-	-	-	-	-	-	0.5024	-	0.5050

Table 2. Summary of distinguish accuracy on SIMON32/64 using N instances

R	Bao [2]	Sect. 6			
		m=2	m=4	m=8	m=16
9	0.6532	0.7240	0.8095	0.8958	0.9630
10	0.5629	0.5907	0.6339	0.6900	0.7608
11	0.5173	0.5240	0.5387	0.5591	0.5878
12	-	-	-	0.5152	0.5225

Second, we use some neutral bits with probability one to generate multiple-plaintext pairs to launch a key recovery attack successfully. Not all plaintext pairs that satisfy the differential ΔS follow the differential ΔP after propagating through the differential path. In order to resolve the requirement of the same distribution of multiple-ciphertext pairs, inspired by the combined response of differential-neural distinguisher in Gohr's work, we use some neutral bit with probability one to generate multiple-plaintext pairs and then encrypt multiple-plaintext pairs to get multiple-ciphertext pairs.

Third, we successfully implemented several rounds of key recovery attacks using the new differential-neural distinguisher with a differential path. We improved 11-, 12-, and 13-round key recovery attacks for SPECK32/64 and reduced the time complexity. Surprisingly, when a 3-round differential path is combined with our 7-round differential-neural distinguisher, the success rate of the 12-round key recovery attack launched is 100%. This shows that our 7-round differential-neural distinguisher can effectively distinguish ciphertext and random numbers. For SIMON32/64, we are able to implement a 17-round key recovery attack using deep learning methods for the first time. Also, we decrease the time complexity of the 16-round key recovery attack. Detailed experimental comparison are shown in Table 3.

The rest of the paper is organized as follows. Section 2 gives the preliminary on the previous differential-neural distinguisher model, key recovery attack process, and generalized neutral bits. Section 3 introduces the training method and accuracy of our differential-neural distinguisher for SPECK32/64. we introduce data generation, experimental environment and complexity calculation in Section 4. Section 5 exhibits the details of our 11-, 12-, 13-round key recovery attacks for SPECK32/64. Section 6 introduces the training method and accuracy of our differential-neural distinguisher for SIMON32/64. Section 7 exhibits the details of our 16-, 17-round key recovery attacks for SIMON32/64. In Section 8, our results and possible extensions are discussed.

2 Preliminary

2.1 Brief Description of SPECK32/64 and SIMON32/64

Notations. Let n be the word size (i.e., the number of bits of a word), and the state size can be denoted as $2n$ bits. Let (x_r, y_r) be the left and right branches of a state after the encryption of r rounds, k_r the subkey of r rounds. Denote the bit-wise XOR by $\oplus$, the addition modulo 2^n by $\boxplus$, the bit-wise AND by $\cdot$, the bit-wise right rotation by $\gg$, the bit-wise left rotation by $\ll$.

SPECK32/64 and SIMON32/64 are members in the lightweight block cipher family SPECK and SIMON [3]. The round function of SPECK32/64 takes a 16-bit subkey k_i and a state consisting of two 16-bit words (x_i, y_i) as input. The state of the next round (x_{i+1}, y_{i+1}) is computed as follows:

$$x_{i+1} := ((x_i \gg 7) \boxplus y_i) \oplus k_i, y_{i+1} := (y_i \ll 2) \oplus x_{i+1}.$$

The round function is repeated 22 times for SPECK32/64.

The round function of SIMON32/64 takes a 16-bit subkey k_i and a state consisting of two 16-bit words (x_i, y_i) as input. The next round state (x_{i+1}, y_{i+1}) is computed as follows:

$$x_{i+1} := (x_i \ll 1) \cdot (x_i \ll 8) \oplus (x_i \ll 2) \oplus y_i \oplus k_i, y_{i+1} := x_i.$$

The round function is repeated 32 times for SIMON32/64.

Table 3. Summary of key recovery attacks on SPECK32/64 and SIMON32/64

Target	r	Dist.	Conf.	Time	Data	Succ. Rate	Key Space	Ref.
SPECK32/64	11	$\mathcal{DD}$	1+6+4	2^{46}	2^{14}	–	2^{64}	[7]
		$\mathcal{ND}$	1+2+7+1	$2^{39.62}$	$2^{13.64}$	52%	2^{64}	[8]
		$\mathcal{ND}$	1+2+7+1	$2^{36.82}$	$2^{15.64}$	97%	2^{64}	Exp. 1
	12	$\mathcal{DD}$	1+7+4	2^{51}	2^{19}	–	2^{64}	[7]
		$\mathcal{ND}$	1+2+8+1	$2^{46.57}$	$2^{22.97}$	40%	2^{64}	[8]
		$\mathcal{ND}$	1+2+8+1	$2^{46.11}$	2^{22}	86%	2^{64}	[2]
		$\mathcal{ND}$	1+2+8+1	$2^{44.75}$	$2^{22.97}$	36%	2^{64}	Exp. 2
		$\mathcal{ND}$	1+3+7+1	$2^{44.82}$	$2^{18.58}$	81%	2^{63}	[2]
		$\mathcal{ND}$	1+3+7+1	$2^{42.32}$	2^{25}	100%	2^{63}	Exp. 3
		$\mathcal{ND}$	1+3+7+1	$2^{41.32}$	2^{24}	–	2^{63}	Ideal
	13	$\mathcal{DD}$	1+8+4	2^{57}	2^{25}	–	2^{64}	[7]
		$\mathcal{ND}$	1+3+8+1	$2^{53.97}$	2^{30}	75%	2^{63}	[2]
		$\mathcal{ND}$	1+3+8+1	$2^{53.85}$	2^{29}	61%	2^{63}	[2]
		$\mathcal{ND}$	1+3+8+1	$2^{52.68}$	2^{31}	49%	2^{63}	Exp. 4
		$\mathcal{ND}$	1+3+8+1	$2^{51.68}$	2^{30}	–	2^{63}	Ideal
SIMON32/64	16	$\mathcal{DD}$	2+12+2	$2^{26.48}$	$2^{29.48}$	62%	2^{64}	[1]
		$\mathcal{ND}$	1+3+11+1	$2^{46.98}$	2^{21}	49%	2^{64}	[2]
		$\mathcal{ND}$	1+3+11+1	$2^{43.19}$	2^{22}	72%	2^{64}	Exp. 5
	17	$\mathcal{ND}$	1+4+11+1	$2^{54.65}$	2^{28}	8%	2^{64}	Exp. 6

1. All time complexities of key recovery attack using $\mathcal{ND}$ are recalculated using the new time complexity calculation formula proposed in Sect. 4.2.
2. $\mathcal{DD}$: differential distinguisher; $\mathcal{ND}$: differential-neural distinguisher;
3. $-$: Not available.
4. Ideal: Assuming two classical differential paths share the same output differences, the time complexity and data complexity of the attack can be halved.

2.2 Overview of Gohr's and Chen's Differential-Neural Distinguisher Model for SPECK32/64

The structure of the neural network in Gohr [8] and Chen [6] is almost identical except input format. Thus, we introduce these two models uniformly. The differential-neural distinguisher is a supervised model which distinguishes ciphertext and random numbers. Ciphertexts and random numbers was artificially generated with corresponding labels. Given m plaintext pairs $\{(P_{i,0}, P_{i,1}), i \in [0, m-1]\}$ and target cipher SPECK32/64, the resulting ciphertext pairs $\{(C_{i,0}, C_{i,1}), i \in [0, m-1]\}$ is regarded as a instance. Note that $m = 1$ in [8], and $m \in \{2, 4, 8, 16\}$ in [6]. Each instance will be attached with a label Y:

$$Y = \begin{cases} 1, & \text{if } P_{i,0} \oplus P_{i,1} = \Delta, i \in [0, m-1] \\ 0, & \text{if } P_{i,0} \oplus P_{i,1} \neq \Delta, i \in [0, m-1] \end{cases}$$

where $\Delta = (0x0040, 0x0000)$. If Y is 1, this instance is sampled from the target distribution and defined as a positive example. Otherwise, this instance is sampled from a uniform distribution and defined as a negative example. A large number of instances need to be put into neural network training. If the neural network can obtain a stable accuracy higher than 0.5 on a test set, it can effectively distinguish ciphertext and random numbers. The differential-neural distinguisher model can be described as:

$$\begin{gathered} \Pr(Y = 1 \mid X_0, \ldots, X_{m-1}) = F\left(f(X_0), \ldots, f(X_{m-1}), \varphi(f(X_0), \ldots, f(X_{m-1}))\right) \\ X_i = (C_{i,0}, C_{i,1}), i \in [0, m-1] \\ \Pr(Y = 1 \mid X_0, \ldots, X_{m-1}) \in [0, 1] \end{gathered}$$

where $f(X_i)$ represents the basic features of a ciphertext pair X_i, and $\varphi(\cdot)$ is the derived features, and $F(\cdot)$ is the new posterior probability estimation function.

The network structure contains several modules described in Fig. 1. The input layer of the neural network consisting of multiple-ciphertext pairs is arranged in a $m \times \omega \times \frac{2L}{\omega}$ array, where L represents the block size of the target cipher, and ω is the size of a basic unit. For example, L is 32 and ω is 16 for SPECK32/64. Module 1 is the initial with-1 convolution layer that intends to make learning simple bit-sliced functions such as bitwise addition easier. Module 2 is the Residual Network. *Conv* stands for one-dimensional convolution $Conv1D$ with N_f filters, and the size of the convolution kernel is $k_s \times k_s$. The number of module 2 is determined by experiment. The prediction head consists of modules 3, 4, and output layer. FC is a fully-connected layer which has d_1 or d_2 neurons. BN is the batch normalization layer. *Relu* and *Sigmoid* are two different activation functions. The output of *Sigmoid* ranges from 0 to 1.

2.3 Differential-neural cryptanalysis

The master key can be recovered if the last two round subkeys of SPECK and SIMON are known according to the key extension algorithm. The differential-neural cryptanalysis, a chosen-plaintext attack, aims to recover the subkey of

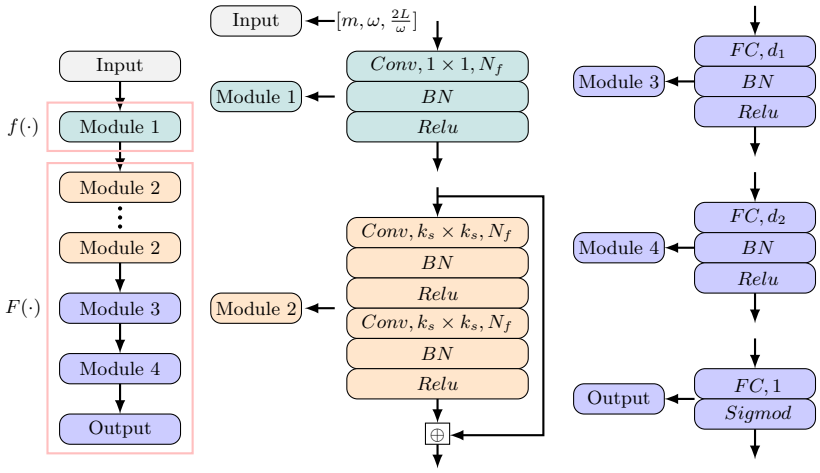


Fig. 1. The network structure of Gohr's and Chen's model

the last two rounds. We decrypt the ciphertext using guessed subkey and use the differential-neural distinguisher to estimate the distance between the guessed subkey and the correct key.

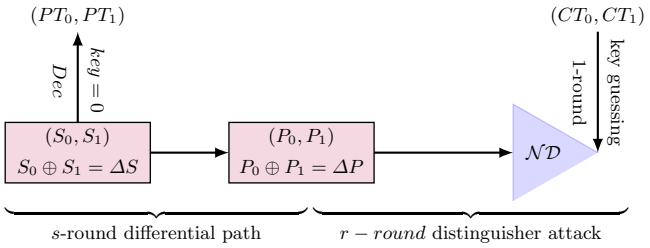


Fig. 2. $(1 + s + r + 1)$-round key recovery attack of differential-neural cryptanalysis

The overall processing of a key recovery attack based on differential-neural distinguisher is shown in Fig. 2, where $\mathcal{ND}$ is the trained differential-neural distinguisher, (PT_0, PT_1) is plaintext paris, and (CT_0, CT_1) is ciphertext paris. The $(1 + s + r + 1)$-round key recovery attack employs an r-round main and $(r - 1)$-round helper differential-neural distinguisher trained using input pairs with difference ΔP. A short s-round differential path $(\Delta S \rightarrow \Delta P)$ with probability denoted by 2^{-p} is prepended on top of the differential-neural distinguisher to increase the number of rounds of key recovery attack. In order to ensure the existence of data pairs satisfying difference ΔP after s-round encryption, about $c \cdot 2^p$ (denoted by n_{cts}) data pairs with difference ΔS are required according to difference propagation probability, where c is a small constant. Neutral bits of the

s-round differential path are used to expand each data pair to a structure of n_b data pairs. The n_{cts} structures of data pairs are decrypted by one round with 0 as the subkey to get plaintext structures because nonlinear operation happens before key addition for SPECK and SIMON. All plaintext structures are encrypted to obtain the corresponding ciphertext structures. Each ciphertext structure is used to select a candidate of the subkey by the r-round main differential-neural distinguisher based on a variant of Bayesian optimization. The usage of ciphertext structures is also highly selective by using a standard exploration-exploitation technique, namely *Upper Confidence Bounds* (UCB). Each ciphertext structure is assigned with a priority according to the score of the recommended subkeys and the visited times. Without exhaustively performing trail decryption, the key search policy depends on the expected response of the differential-neural distinguisher upon wrong-key decryption. The *wrong key response profile* is used to recommend new candidate values from previous candidate values though minimizing the weighted Euclidean distance in a BAYESIANKEYSEARCH Algorithm [8].

2.4 Combined Response and Neutral Bits

As the number of encryption rounds increases, the accuracy of differential-neural distinguisher decreases. A differential-neural distinguisher with an accuracy higher than 0.5 means some distinguisher advantage over a random distinguisher. To reduce the impact of the misjudgment of the single prediction of the distinguisher, Gohr used the combined response of the differential-neural distinguisher over large amounts of samples of the same distribution, which can be satisfied by neutral bits [8]. The primary notion of neutral bits can be interpreted as follows.

Definition 1 (Neutral bits of a differential, NB[5]). *Let $\Delta_{in} \rightarrow \Delta_{out}$ be a differential with input difference Δ_{in} and output difference Δ_{out} of an r-round encryption F^r. Let (P, P') be the input pair, and $(C, C' \mid C = F^r(P), C' = F^r(P'))$ be the output pair, where $P \oplus P' = \Delta_{in}$. If $C \oplus C' = \Delta_{out}$, (P, P') is said to be conforming the differential $\Delta_{in} \rightarrow \Delta_{out}$. Let $e_0, e_1, \ldots, e_{n-1}$ be the standard basis of $\mathbb{F}_2^n$. Let i be an index of a bit (starting from 0). The i-th bit is a neutral bit for the differential $\Delta_{in} \rightarrow \Delta_{out}$, if $(P \oplus e_i, P' \oplus e_i)$ is also a confirming pair for any confirming pair (P, P').*

The responses $v_{i,k}$ from the differential-neural distinguisher on ciphertext pairs in the ciphertext structure (of size n_b) are combined using the Formula $s_k = \sum_{i=0}^{n_b-1} \log_2 \left(\frac{v_{i,k}}{1-v_{i,k}} \right)$ and used as the score of the recommended subkey. The score s_k plays a decisive role in the attack's execution time and success rate. For a combined-response-distinguisher built on top of a weak differential-neural distinguisher to reach its most potential for distinguishability, the number of samples of the same distribution should be sufficiently large. In general, neutral bits of the non-trivial differential path are scarce. Therefore, probabilistic neutral bits (PNB) are exploited in [8]. Some probabilistic neutral bits with a higher probability, simultaneous-neutral bit-sets (SNBS), and conditional (simultaneous-)

neutral bit(-set)s (CSNBS) was found in [2]. The explicit definitions of PNB, SNBS, and CSNBS (collectively referred to as generalized neutral bits) refer to [2].

3 Differential-Neural Distinguishers for Round-Reduced Speck32/64

3.1 Network Structure

The overall structure of our neural network for training the differential-neural distinguisher is shown in Fig. 3. Our neural network consists of four parts: an input layer consisting of multiple-ciphertext pairs, an initial convolutional layer consisting of four parallel convolutional layers, a residual tower of multiple two-layer convolutional neural networks, and a prediction head consisting of multiple fully connected layers (according to the color to distinguish in the Fig 3).

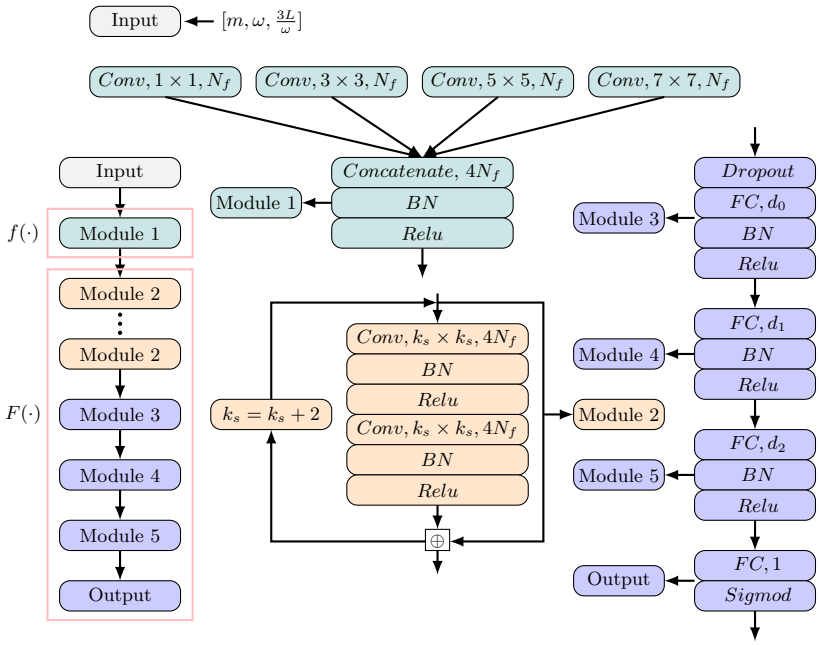


Fig. 3. The network structure of our distinguisher model for Speck32/64

Input Representation. Once the output of the r-th round $(C, C') = (x_r||y_r, x'_r||y'_r)$ is known, one can directly compute (y_{r-1}, y'_{r-1}) without knowing the $(r-1)$-th subkey according to the round function of Speck. Thus, the neural network accept data of the form $(x_r, x'_r, y_r, y'_r, y_{r-1}, y'_{r-1})$. The input layer has

m group ciphertext pairs consisting of $3L$ units likewise arranged in a $[m, \omega, \frac{3L}{\omega}]$ array, where $L = 32, \omega = 16$ for SPECK32/64. The detailed data generation method of the input layer is shown in Fig. 4, where $C_{i,0} = (x_r, y_r), C_{i,1} = (x'_r, y'_r), C_{i,2} = (y_{r-1}, y'_{r-1}), i \in (0, m)$.

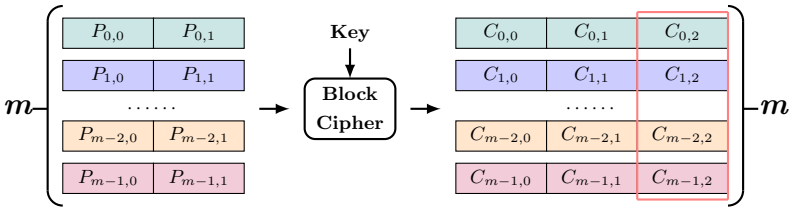


Fig. 4. The format of input data

Initial Convolution. The input layer is connected to the initial convolutional layer, which comprises four convolutional layers with N_f channels of different kernel sizes. The four convolution layers are concatenated at the channel dimension, similar to the Inception [13] in GoogLeNet. Batch normalization is applied to the output of concatenate layers. Finally, rectifier nonlinearity is applied to the output of batch normalization, and the resulting $[m, \omega, 4N_f]$ matrix is passed to the convolutional blocks layer. The structure of the initial convolution layer can be seen in Fig. 5.

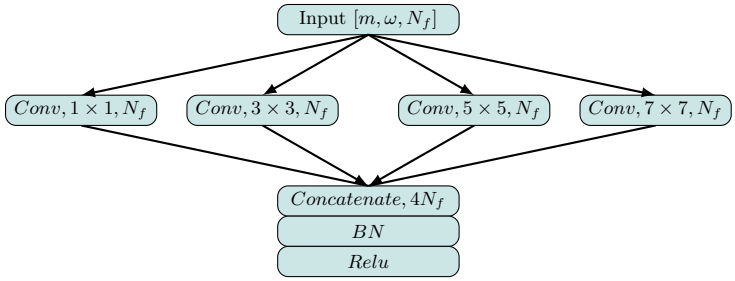


Fig. 5. The initial convolution layer

Convolutional Blocks. Each convolutional block consists of two layers of $4N_f$ filters. Each block applies first the convolution with kernel size k_s, then a batch normalization, and finally a rectifier layer. At the end of the convolutional block, a skip connection is added to the output of the final rectifier layer of the block to the input of the convolutional block and transfers the result to the next block.

After each convolutional block, the kernel size increases by 2. The number of convolutional blocks is 5 in our model (determined by experiment). The structure of the convolutional blocks layer can be seen in Fig. 6.

Prediction Head. The prediction head consists of three hidden layers and one output unit. Before the first hidden layer, we add a dropout layer to prevent model overfitting. The four fully connected layers comprise 512, 64, 64, and 1 unit, followed by the batch normalization and rectifier layers. The final layer consists of a single output unit using a *Sigmoid* activation function. The structure of the prediction head is shown in Fig. 7.

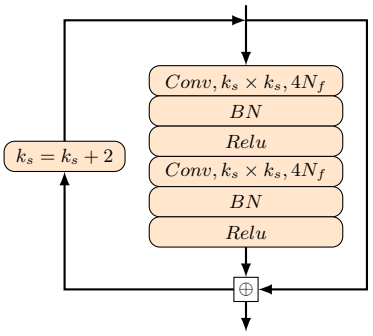


Fig. 6. The convolutional blocks

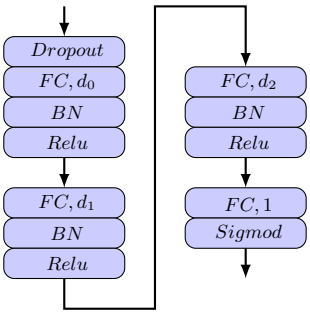


Fig. 7. The prediction head

Rationale. First, we modify the format of input data for the model to use more ciphertext. Given the ciphertext of the last round, the right half of the ciphertext of the penultimate round can be calculated according to the round function of SPECK32/64 without knowing the (r-1)-th subkey. Second, considering the circular shift operation in the round function of SPECK and some relationship in the adjacent bits generated by the modular addition operation, we add the convolution operation of widths 3, 5, and 7 to capture the feature from more dimensions which is inspired by the Inception [13] in GoogLeNet to capture more dimensional information. Third, to increase the convolution's receptive field, the convolution kernel's size increases by 2 with the increase of the depth of the Residual Network. Also, Our network structure has a strong fitting ability, and it is easy to overfit on the training set, so we add a dropout layer before the fully connected layer to improve the network's generalization ability.

3.2 The Training of Differential-Neural Distinguisher

The distinguish accuracy is the most critical indicator which reflects the performance of the differential-neural distinguisher. The following training was carried out to verify the performance of our differential-neural distinguisher.

Data Generation. Training and test data were generated by using the Linux random number generator to obtain uniformly distributed keys K_i and m plaintext pairs $(P_{i,0}, P_{i,1})$ with the input difference $\Delta = (0x0040, 0x0000)$ as well as a vector of binary-valued labels Y_i. During producing training set or test set for r-round SPECK32/64, the m plaintext pair $(P_{i,0}, P_{i,1})$ was then encrypted for r rounds if $Y_i = 1$, while otherwise the second plaintext of the pairs was replaced with a freshly generated random plaintext and then encrypted for r rounds.

Remark 1. We conduct two sets of experiments with different numbers of datasets. In [8], the training set and test set include N and M instances, which consist of a ciphertext pair, that is, total N and M ciphertext pairs, respectively. In [6], the training set and test set include N/m and M/m instances, and each instance includes m ciphertext pairs; that is, the total numbers of ciphertext pairs used are N and M. To ensure a fair comparison, We used the same amount of data as [6] in the first set of experiments. However, Using N/m and M/m instances as training and test sets may lead to overfitting (See *Remark 2* for details). In order to overcome this problem, we also use N and M instances as training test and test set in the second set of experiments, which consists of m ciphertext pair, that is, total $N \times m$ and $M \times m$ ciphertext pairs, respectively.

Basic Training Scheme. We run the training for 20 epochs on the dataset for $N = 10^7$ and $M = 10^6$. The batch size processed by the dataset is adjusted according to the parameter m to maximize GPU performance. Optimization was performed against mean square error loss plus a small penalty based on L2 weights regularization parameter $c = 10^{-5}$ using the Adam algorithm [12]. A cyclic learning rate schedule was applied, setting the learning rate l_i for epoch i to $l_i = \alpha + \frac{(n-i) \mod (n+1)}{n}.(\beta - \alpha)$ with $\alpha = 10^{-4}, \beta = 2 \times 10^{-3}$ and $n = 9$. The networks obtained at the end of each epoch were stored, and the best network by validation loss was evaluated against a test set.

Training using the Staged Train Method. We use several stages of pre-training to train an 8-round differential-neural distinguisher for SPECK32/64. First, we use our 7-round distinguisher to recognize 5-round SPECK32/64 with the input difference (0x8000,0x804a) (the most likely difference to appear three rounds after the input difference (0x0040,0x0000)). The training was done on 10^7 instances for 20 epochs with cyclic learning rates. Then we trained the distinguisher obtained to recognize 8-round SPECK32/64 with the input difference $(0x0040, 0x0000)$ by processing 10^7 freshly generated instances for 10 epochs with a learning rate of 10^{-4}. Finally, the learning rate was dropped to 10^{-5} after processing another 10^7 fresh instances for 10 epochs. For the 9-round distinguisher, the overall training method is the same as the 8-round distinguisher. We used an 8-round distinguisher to identify 5-round SPECK32/64 with the input difference (0x850a,0x9520) (the most likely difference to appear four rounds after the input difference (0x0040,0x0000).

3.3 Result

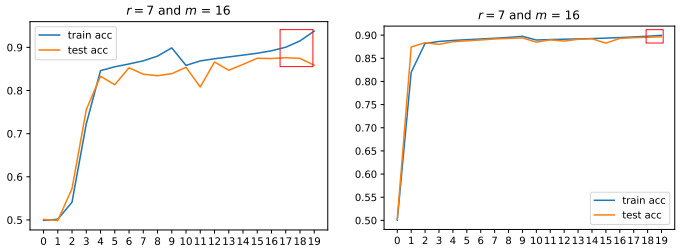
Test Accuracy. We summarize the experimental results of 6-, 7-, 8-, 9-round differential-neural distinguisher in Table 4 compared to [8,6]. The 6-, 7-round distinguishers were trained using the basic training method, while the 8-, 9-round distinguishers were trained using the staged training method. The role of the differential-neural distinguisher is to distinguish between ciphertext and random numbers, so if the accuracy is greater than 0.5, it is considered effective. The "N/m" column is the result of accuracy using N/m and M/m instances as the training and test sets, respectively. The "N" column results from accuracy using N and M instances as the training and test sets, respectively. From Table 4, the accuracy of our differential-neural distinguisher was significantly improved both the "N/m" column and the "N" column compared to [8,6]. Under the two experiments, the accuracy of the distinguisher is almost same except for $m = 16$. This exception is caused by model overfitting, see *Remark 2* for details. Furthermore, even using N instance data in [8,6], the improvement in the accuracy of the distinguisher is very small. Moreover, we obtained a 9-round differential-neural distinguisher while [8,6] cannot be trained.

Table 4. Summary of distinguish accuracy on SPECK32/64 using different number of instances

R	Gohr [8]	m=2			m=4		
		Chen [6]	N/m	N	Chen [6]	N/m	N
6	0.788	0.8613	0.8771	0.8773	0.931	0.9468	0.9497
7	0.614	0.6393	0.6663	0.6649	0.6861	0.7194	0.7283
8	0.514	-	-	-	-	0.5347	0.5428
R	Gohr [8]	m=8			m=16		
		Chen [6]	N/m	N	Chen [6]	N/m	N
6	0.788	0.9562	0.9868	0.9895	0.9802	0.9969	0.9992
7	0.614	0.7074	0.7991	0.8106	0.6694	0.8759	0.8963
8	0.514	-	0.5412	0.5590	-	0.5597	0.5854
9	-	-	-	0.5024	-	-	0.5050

Remark 2. Why do we train a differential-neural distinguisher with two different numbers of data? For the sake of fairness of comparison, we must use N/m and M/m instances as training test and test set (refer to *Remark 1*). From Fig. 8a, we can see that the difference between training accuracy and test accuracy is relatively large. Therefore, using N/m and M/m instances as a training and test sets to train a neural network will suffer from overfitting, especially when the number of rounds r and the group size m is large. However, training accuracy and test accuracy are almost equal in Fig. 8b. Therefore, using N and M instances as training set and test set to train the differential-neural distinguisher

can avoid overfitting, speed up the model convergence and improve the model accuracy to a certain extent. Due to the overfitting phenomenon, the accuracy of the distinguisher will be low, which also affects our training of more rounds of differential-neural distinguisher. Therefore, we train the differential-neural distinguisher for various reasons using two different numbers of data.



(a) Using N/m instances (b) Using N instances

Fig. 8. Overfitting with different number of instances.

Wrong Key Response Profile. In [8], an improved key recovery attack was presented by using *Upper Confidence Bound* and *Bayesian Optimization*. More specifically, the key search policy depends on an important observation that the expected response of the differential-neural distinguisher upon wrong-key decryption will rely on the bitwise difference between the guessed key and the real key. The wrong key response profile, which can be precomputed, is used to recommend new candidate values for the key from previous candidate values by minimizing the weighted Euclidean distance as the criteria in a BayesianKey-Search Algorithm. It recommends a set of subkeys and provides their scores without exhaustively performing trial decryption. If the Hamming distance between the correct and wrong keys is small, the distinguisher will score high; otherwise, the score will be low. From Fig. 9, we know the score of our differential-neural distinguisher is higher than that of Gohr's distinguisher when the Hamming distance of key difference is small. When the Hamming distance is large, we can obtain a lower score. In other words, our distinguisher makes the difference between the score of the correct key and the wrong key even more significant, i.e., our differential-neural distinguisher is better.

4 From Differential-Neural Distinguisher to Key Recovery Attack

4.1 Generation of Same Distributed Data

During the training of the differential-neural distinguisher, we take multiple-ciphertext pairs as input to the neural network. The data in multiple-ciphertext

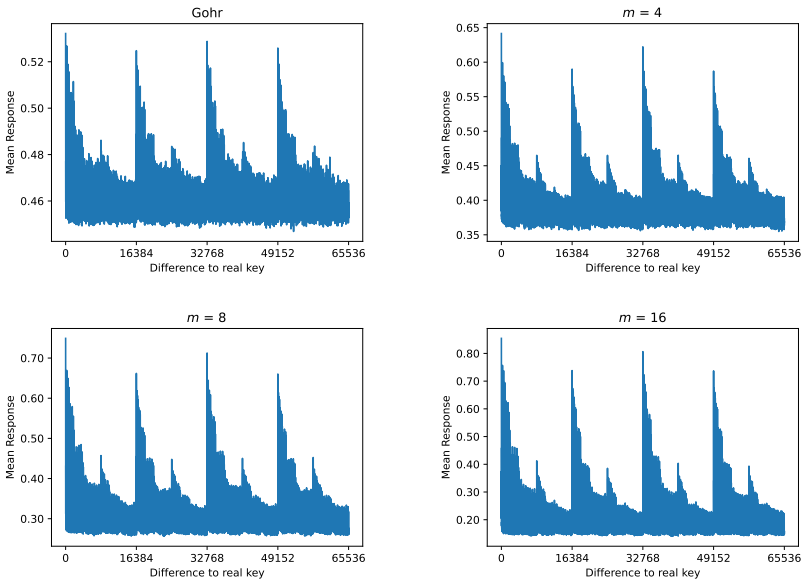


Fig. 9. Wrong key response profile for 7-round SPECK32/64 using N instances

pair has the same distribution, i.e., if it is a positive example, the data in multiple-ciphertext pair is encrypted by multiple-plaintext pair satisfying a fixed difference; if it is a negative example, the data in multiple-ciphertext pair is encrypted by a randomly generated multiple-plaintext pair. After obtaining a differential-neural distinguisher, we can launch a key recovery attack. To increase the number of rounds for the key recovery attack, we add a differential path before the differential-neural distinguisher. However, the differential path is probabilistic. Even if multiple-plaintext pairs have the same difference, after propagation through the differential path, the difference of output could not be the same. Therefore, the initial input(multiple-plaintext pairs) cannot be randomly generated during a key recovery attack. Neutral bits can resolve the problem of obtaining ciphertext that satisfies the same distribution, ensuring that we can launch key recovery attacks. We randomly generate a plaintext pair that meets the initial difference and use $\log_2 m$ neutral bits to obtain m plaintext pairs. Moreover, to choose neutral bits with high probability, the optimal choice is to use neutral bits with probability one. The m ciphertext pairs obtained by propagation of the m plaintext pairs through the differential path have the same differential ΔP with a substantial probability. After r rounds of encryption, the obtained multiple-ciphertext pairs have the same distribution.

4.2 Experimental Environment and Complexity Calculation

The Core of the key recovery attack was examined using a server with multiple GeForce GTX 2080-Ti GPUs. When a fast graphics card is used, the performance of our implementation is not limited by the speed of neural network evaluation but by the total number of iterations on the ciphertext structures. We count a key guess as successful if the subkey of the last round was guessed correctly and guessed subkey of the second round is at the hamming distance at most two of the real key.

Data Complexity. The data complexity of the experiment is calculated by the formula $n_{kg} \times n_b \times n_{ct} \times m \times 2$. During a key recovery attack, the differential path used may need to satisfy some conditions involving the key. n_{kg} indicates the number of times to guess the key involved in the condition The data complexity is calculated as theoretical values. $n_b = 2^{|NB|}$, where $|NB|$ instead of the number of neutral bits. n_{ct} represents the number of ciphertext structures used, and m is the number of ciphertext pairs in each instance. In the actual experiment, when the accuracy of the differential-neural distinguisher is high, the key can be recovered quickly and successfully. Not all the data is used, so the actual data complexity is lower than the theoretical.

Time Complexity. In [8], a highly optimized implementation of SPECK32/64 could perform brute force key search at a speed of about 2^{28} key per second per core. A correct key guess takes an average of 500 seconds, and the single success rate of a key recovery attack of 11-round SPECK32/64 is 52.1%. Adjusting for the empirically measured success rate of the key recovery attack, about 1000 seconds is needed to execute the key recovery attack until a correct key is found on a single core. This yields an estimated computational attack complexity of $2^{28} \times 1000 \approx 2^{38}$ SPECK32/64 encryptions until a solution is found. The formula in [8] for calculating the time complexity does not take into account the success rate of the key recovery attack. To better evaluate the effect of key recovery attacks, we give a new calculation method and update the previous result accordingly. To reduce the experimental error, we conduct multiple key recovery attacks, take the average running time rt as the running time of an experiment, and divide the number of successful key recovery experiments by the total number of experiments as the success rate sr of a single key recovery attack. We calculate how many experiments need to be performed to ensure that at least one experiment is successful. When the overall success rate is 99%, we consider the experiment to be successful, and the number of experiments ne is: $1-(1-sr)^{ne} = 0.99$, i.e., $\log_{1-sr} 0.01$. For 11-round SPECK32/64 in [8], this yields an estimated computational attack complexity of $2^{28} \times 500 \times \log_{1-0.52} 0.01 \approx 2^{39.6}$, and the average running time is 225s in 300 trials in our equipment. To facilitate comparison, the time complexity of experiment in [8] is taken as the benchmark for conversion. The new time complexity calculation formula:

$$n_{kg} \times 2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01.$$

5 Key Recovery Attack on Round-Reduced Speck32/64

This section shows that our differential-neural distinguisher can improve the performance of key recovery attacks. Our attack follows the same framework in [8,2], only replacing the differential-neural distinguisher trained using N instances. For a more detailed procedure, refer to Appendix B. We count a key guess as successful if the last round key was guessed correctly and the second round key is at the hamming distance at most two of the real key. In the following, we significantly decreased the time complexity of 11-, 12-, and 13-round key recovery attacks for Speck32/64 by employing an improved differential-neural distinguisher.

5.1 Key Recovery Attack on 11-round Speck32/64

To demonstrate the superiority of our differential-neural distinguisher, we conducted a comparative experiment with Gohr's differential-neural distinguisher. To ensure the fairness of the comparative experiment, except for the difference of differential-neural distinguisher, other parameters should be the same.

Experiment 1: The components of key recovery attack $\mathcal{A}^{\text{Speck11}R}$ of 11-round Speck32/64 are shown as follows.

1. 2-round differential path $(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$;
2. neutral bits set NBS are $\{[20], [21], [22], [14], [15], [23], [7], [30], [0], [8]\}$; generalized neutral bits of generating multiple-ciphertext pairs: the $\log_2 m$ neutral bit with the largest probability from NBS; generalized neutral bits of combined response of differential-neural distinguisher: 6 remaining neutral bits from NBS (refer to Appendix A.1 Table 7).
3. 7-round differential-neural distinguisher $\mathcal{ND}^{\text{Speck}_{7R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{Speck}_{7R}} \cdot \mu$ and $\mathcal{ND}^{\text{Speck}_{7R}} \cdot \delta$.
4. 6-round differential-neural distinguisher $\mathcal{ND}^{\text{Speck}_{6R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{Speck}_{6R}} \cdot \mu$ and $\mathcal{ND}^{\text{Speck}_{6R}} \cdot \delta$.

The parameters for recovery the last two subkeys are denoted as follows.

1. n_{kg}: the number of possible values for the bits of k_0.
2. n_{cts}: the number of ciphertext structures.
3. n_b: the number of ciphertext pairs in each ciphertext structures, i.e., $2^{|NB|}$.
4. n_{it}: the total number of iterations on the ciphertext structures.
5. c_1 and c_2: the cutoffs with respect to the scores of the recommended last subkey and second to last subkey, respectively.
6. n_{byit1}, n_{cand1} and n_{byit2}, n_{cand2}: the number of iterations and number of key candidates within each iteration in the BayesianKeySearch Algorithm for guessing each of the last and the second to last subkeys, respectively.

In the experimental verification of the attack $\mathcal{A}^{\text{Speck11}R}$, the 7-round and 6-round differential-neural distinguisher was provided in Sect. 3. Concrete parameters used in our 11-round key recovery attack $\mathcal{A}^{\text{Speck11}R}$ are listed as follows.

$n_b = 2^6$ $n_{ct} = 100/200$ $n_{it} = 500$
$c_1 = 8, c_2 = 10$ $n_{byit1} = n_{byit2} = 5$ $n_{cand1} = n_{cand2} = 32$

In Experiment 1, several experiments were done for different values of m and n_{ct}, and the specific experimental results are shown in Table 5. The data complexity is $n_b \times n_{ct} \times m \times 2$. The core of these attacks were examined in 35 trials. The time complexity is $2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01$.

Table 5. Experiment result of 11-round key recovery attack

	m	Conf.	$n_{ct} = 100$				$n_{ct} = 200$			
			Succ. Rate	Run Time	Data	Time	Succ. Rate	Run Time	Data	Time
Gohr [8]	1		52.1%	225	$2^{13.64}$	$2^{39.61}$	80%	194	$2^{14.64}$	$2^{38.27}$
Our	2	1+2+7+1	57.14%	178	$2^{14.64}$	$2^{39.07}$	97.14%	158	$2^{15.64}$	$2^{36.82}$
	4		77.14%	165	$2^{15.64}$	$2^{38.16}$	94.28%	137	$2^{16.64}$	$2^{36.93}$
	8		85.71%	234	$2^{16.64}$	$2^{38.26}$	91.43%	243	$2^{17.64}$	$2^{37.98}$
	16		77.14%	464	$2^{17.64}$	$2^{39.65}$	94.29%	137	$2^{18.64}$	$2^{36.93}$

From Table 5, we know that the success rate is significantly improved, and the time complexity is decreased compared to the result in [8]. This phenomenon shows that our distinguisher can very accurately determine whether the guessed key is correct, thereby speeding up the process of key recovery attacks. Although the data complexity of key recovery attacks increases in this case, the time complexity is generally lower than [8].

In addition, the success rate of the key recovery attack can still be significantly improved when n_{ct} is 200. Because the probability of 2-round difference path $(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$ is 2^{-6}, the average number of correct ciphertext structures are only 1.56 when n_{ct} is 100. Therefore, correct ciphertext structures maybe not exist, so the success rate of the key recovery attack is low. Thus, two aspects affect the success rate of key recovery attacks. First, the accuracy of the differential-neural distinguisher should be higher. Second, there should be enough plaintext pairs to ensure that after differential propagation, a certain number of ciphertext pairs still meets the distribution of input data of the neural network.

5.2 Key Recovery Attack on 12-round Speck32/64

To verify the performance of our 8-round differential-neural distinguisher, the following experiments were carried out in this subsection.

Experiment 2: The components of key recovery attack $\mathcal{A}_I^{\text{Speck12}R}$ of 12-round Speck32/64 are shown as follows.

1. 2-round differential path $(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$;
2. generalized neutral bits of generating multiple-ciphertext pairs: $\{[9, 16], [2, 11, 25], [6, 29]\}$; generalized neutral bits of combined response of differential-neural distinguisher:$\{[20], [21], [22], [14], [15], [23], [30], [7], [0]\}$ (refer to Appendix A.1 Table 8).
3. 8-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{8R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{8R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{8R}} \cdot \delta$.
4. 7-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{7R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \delta$.

In the experimental verification of the attack $\mathcal{A}_I^{\text{SPECK}12R}$, the 8-round and 7-round differential-neural distinguisher was provided in Sect. 3. Concrete parameters used in our 12-round key recovery attack $\mathcal{A}_I^{\text{SPECK}12R}$ are listed as follows.

$$\begin{array}{llll} m = 8 & n_b = 2^9 & n_{cts} = 500 & n_{it} = 2000 \\ c_1 = 3 & c_2 = 8 & n_{byit1} = n_{byit2} = 5 & n_{cand1} = n_{cand2} = 32 \end{array}$$

The data complexity is $n_b \times n_{ct} \times m \times 2 = 2^9 \times 500 \times 8 \times 2 = 2^{22.97}$ plaintexts. The core of the attack was examined in 70 trials. There are 25 succeeded trials, the average run times of every trails in our server is 4785s. Thus, we count the success rate is $^{25}/_{70}$, which is 0.357. The time complexity is $2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01 = 2^{28} \times 4785 \times {}^{500}/_{225} \times \log_{1-0.357} 0.01 = 2^{44.75}$.

In pursuit of better key recovery attack, we considered combining the 3-round differential path and the more powerful 7-round differential-neural distinguisher. But in order for the 3-round differential path to hold, 3 conditions need to be met (refer to Appendix A.1 Table 9). Because of the extended one round on top of these 3-round differentials, these conditions cannot be fulfilled by chosen data without guessing corresponding bits of k_0. To make the experimental verification economic, we tested the core of the attack with the three conditions being fulfilled only. Because the prepended differential path are valid to keys fulfilling $k_2[12] \neq k_2[11]$, thus we tested for these valid keys only, and the presented attack works for 2^{63} keys.

Experiment 3: The components of key recovery attack $\mathcal{A}_{II}^{\text{SPECK}12R}$ of 12-round SPECK32/64 are shown as follows.

1. 3-round differential path $(0x8020, 0x4101) \rightarrow (0x0040, 0x0000)$;
2. generalized neutral bits of generating multiple-ciphertext pairs: $\{[22], [20], [13]\}$; generalized neutral bits of combined response of differential-neural distinguisher:$\{[12, 19], [14, 21], [6, 29], [30], [0, 8, 31], [5, 28]\}$ (refer to Appendix A.1 Table 10).
3. 7-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{7R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \delta$.
4. 6-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{6R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{6R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{6R}} \cdot \delta$.

In the experimental verification of the attack $\mathcal{A}_{II}^{\text{SPECK}12R}$, the 8-round and 7-round differential-neural distinguisher was provided in Sect. 3. At the beginning, we guess three key bits of k_0, that is $k_0[7]$, $k_0[5] \oplus k_0[14]$, and $k_0[15] \oplus k_0[8]$

because the 3-round differential(refer to Appendix A.1 Table 9) and one key bits $k_0[12] \oplus k_0[5]$ for employing one conditional neutral bits (refer to Appendix A.1 Table 10). Thus, n_{kg} is 16, and there are 2^4 loops. Concrete parameters used in our 12-round key recovery attack $\mathcal{A}_{II}^{\text{SPECK12}R}$ are listed as follows.

$n_{kg} = 2^4$	$m = 8$	$n_b = 2^6$	$n_{cts} = 2^{11}$
$n_{it} = 2^{12}$	$c_1 = 7, c_2 = 10$	$n_{byit1} = n_{byit2} = 5$	$n_{cand1} = n_{cand2} = 32$

The data complexity is $n_{kg} \times n_b \times n_{ct} \times m \times 2 = 2^4 \times 2^6 \times 2^{11} \times 8 \times 2 = 2^{25}$ plaintexts. The core of the attack was examined in 98 trials. There are 98 succeeded trials, the average run times of every trails in our server is 577s. Thus, we count the success rate is 100%. The time complexity is $2^{28} \times rt \times {}^{500}/_{225} = 2^{28} \times 577 \times {}^{500}/_{225} = 2^{42.32}$.

It is the first time that a key recovery attack has a 100 percent success rate. When we do not use the weak key and modify the ciphertext structure to 2^{12}, the success rate of the key recovery attack is only 51%. The main reason for the reduced success rate is that no correct ciphertext structures exists. Notably, if we exclude the absence of the correct ciphertext structure, our success rate is still 100%. This shows that our differential-neural distinguisher has a powerful ability to distinguish between ciphertext and random numbers.

5.3 Key Recovery Attack on 13-round Speck32/64

Combining a 3-round differential path with an 8-round differential-neural distinguisher, we examine how far a practical attack can go on 13-round Speck32/64 in this subsection.

Experiment 4: The components of key recovery attack $\mathcal{A}^{\text{SPECK13}R}$ of 13-round Speck32/64 are shown as follows.

1. 3-round differential path $(0x8020, 0x4101) \rightarrow (0x0040, 0x0000)$;
2. generalized neutral bits of generating multiple-ciphertext pairs: $\{[22], [20], [13]\}$; generalized neutral bits of combined response of differential-neural distinguisher:$\{[5, 28], [15, 24], [12, 19], [6, 29], [6, 11, 12, 18], [4, 27, 29], [14, 21], [0, 8, 31], [30]\}$ (refer to Appendix A.1 Table 10).
3. 8-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{8R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{8R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{8R}} \cdot \delta$.
4. 7-round differential-neural distinguisher $\mathcal{ND}^{\text{SPECK}_{7R}}$ under difference $(0x0040, 0x0000)$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \mu$ and $\mathcal{ND}^{\text{SPECK}_{7R}} \cdot \delta$.

In the experimental verification of the attack $\mathcal{A}^{\text{SPECK13}R}$, the 8-round and 7-round differential-neural distinguisher was provided in Sect. 3. At the beginning, we guess three key bits of k_0, that is $k_0[7]$, $k_0[5] \oplus k_0[14]$, and $k_0[15] \oplus k_0[8]$ because the 3-round differential(refer to Appendix A.1 Table 9) and four key bits $k_0[12] \oplus k_0[5]$, $k_0[1]$, $k_0[2] \oplus k_0[11]$, $k_0[11] \oplus k_0[4]$ for employing four conditional neutral bits (refer to Appendix A.1 Table 10). Thus, n_{kg} is 2^7, and there are 2^7 loops. Concrete parameters used in our 13-round key recovery attack $\mathcal{A}^{\text{SPECK13}R}$ are listed as follows.

$n_{kg} = 2^7$	$m = 8$	$n_b = 2^9$	$n_{cts} = 2^{11}$
$n_{it} = 2^{12}$	$c_1 = 5, c_2 = -86$	$n_{byit1} = n_{byit2} = 5$	$n_{cand1} = n_{cand2} = 32$

To make the experimental verification economic, we tested the core of the attack with the seven conditions being fulfilled only, including three conditions for confirming the 3-round differential path (refer to Appendix A.1 Table 9) and four conditions for increasing probability of neutral bits (refer to Appendix A.1 Table 10). Because the prepended differential path is valid when the keys fulfilling $k_2[12] \neq k_2[11]$, we tested for these valid keys only, and the presented attack works for 2^{63} keys.

The data complexity is $n_{kg} \times n_b \times n_{ct} \times m \times 2 = 2^7 \times 2^9 \times 2^{11} \times 8 \times 2 = 2^{31}$ plaintexts. The core of the attack was examined in 35 trials. There are 17 succeeded trials, the average run times of every trails in our server is 13690s. Thus, we count the success rate is $^{17}/_{35}$, which is 0.4857. The time complexity is $n_{kg} \times 2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01 = 2^7 \times 2^{28} \times 13690 \times {}^{500}/_{225} \times \log_{1-0.4857} 0.01 = 2^{52.68}$.

5.4 Using Multiple Differential paths

Bao *et al.* found that the output of the differential path matters to the differential-neural distinguisher, but not the input difference. Hence, more than one differential path can be prepended to a differential-neural distinguisher, as long as they share the same output difference. Multiple such differential paths can share some neutral bits. Using such differential paths might enable data reuse, thus slightly reducing data complexity. Bao launched the $R_{1+3+7+1}$-round key recovery attack using four sub-optimal differential paths and the $R_{1+3+8+1}$-round key recovery attack using two sub-optimal differential paths (refer to Appendix A.1 Table 9). However, we only use one differential path in our experiment when we launch the key recovery attack by the prepended 3-round differential path on top of the differential-neural distinguisher. When using two sub-optimal differential paths to generate multiple-ciphertext pairs, we only need to use two conditions. So, we can halve both the data complexity and the time complexity of the 12-round and 13-round key recovery attacks in the ideal situation.

6 Differential-Neural Distinguishers on Round-Reduced Simon32/64

In CRYPTO'19, Gohr used the Residual Network to train differential-neural distinguisher for Speck32/64. Bao *et al.* modified the structure of Gohr's neural network using the Dense Network and Squeeze-and-Excitation Network, training 9-, 10-, and 11-round differential-neural distinguishers for Simon32/64 [2]. This section presents our differential-neural distinguisher on Simon32/64. Compared to Bao's results in [2], we improve the accuracy of the 9-, 10-, 11-round and obtain a 12-round differential-neural distinguishers for Simon32/64.

6.1 Network Structure

In Sect. 3, we detailed the network structure of the differential-neural distinguisher for the SPECK32/64 but found that the trained differential-neural distinguisher model had a deficiency. In the network structure of SPECK32/64, the prediction head uses four fully connected layers, resulting in too many model parameters, making the model training time too long and the model file too large. When designing the network structure of SIMON32/64, we solved this problem by modifying the structure of the prediction head and replacing the fully connected layer with a GlobalAveragePooling layer. The network structure of SIMON32/64 is similar to SPECK32/64 overall, with only minor modifications based on the round function of SIMON32/64. The overall framework of our neural network for training the differential-neural distinguisher for SIMON32/64 is shown in Fig 10.

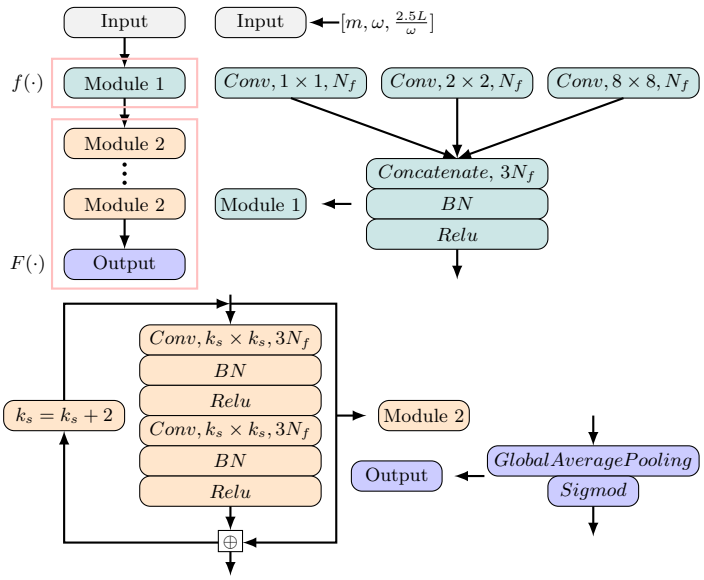


Fig. 10. The network structure of our distinguisher model for SIMON32/64

Input Representation. Based on the round function, $(y_{r-1} \oplus y'_{r-1})$ can be got without knowing the $(r-1)$-th subkey for SIMON32/64. Thus, the neural network accept data of the form $(x_r, x'_r, y_r, y'_r, y_{r-1} \oplus y'_{r-1})$. The input layer has m group ciphertext pairs consisting of $2.5L$ units likewise arranged in a $[m, \omega, \frac{2.5L}{\omega}]$ array, where $L = 32, \omega = 16$ for SIMON32/64.

Initial Convolution. The input layer is connected to the initial convolutional layer, which comprises three convolution layers with N_f channels of kernel sizes 1, 2, and 8. The three convolution layers are concatenated at the channel dimension. Batch normalization is applied to the output of concatenate layers. Finally, rectifier nonlinearity is applied to the output of batch normalization, and the resulting $[m, \omega, 3N_f]$ matrix is passed to the Convolutional Blocks layer.

Convolutional Blocks. The convolutional blocks layer of the differential-neural distinguisher model is the same as SPECK32/64, except that the shape of the input is different.

Prediction Head. The prediction head consists of a GlobalAveragePooling layer and an output unit using a *Sigmoid* activation function. The structure of the prediction head is shown in Fig. 11.

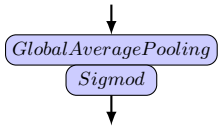


Fig. 11. The prediction head

6.2 The Training of Differential-Neural Distinguisher

Training using the basic scheme. We used two different numbers of training and test sets to train the neural network. The first situation of experiments uses N/m and M/m instances as the training and test set. The second situation of experiments uses N and M instances as the training set and test set. For training parameters, refer to the basic training scheme of SPECK32/64 in Sect. 3. By modifying the network structure and using the basic training scheme, we trained the differential-neural distinguisher to recognize output pairs of 9-, 10-, 11-round SIMON32/64 with the input difference $(0x0000, 0x0040)$.

Training using the Staged Training Method. A 12-round differential-neural distinguisher of SIMON32/64 was trained using several pre-training stages. First, we use our 11-round distinguisher to recognize 9-round SPECK32/64 with the input difference (0x0440,0x0100) (the most likely difference to appear three rounds after the input difference (0x0000,0x0040). The training was done on 10^7 instances for 20 epochs with cyclic learning rates. Then we trained the distinguisher so obtained to recognize 12-round SPECK32/64 with the input difference $(0x0000, 0x0040)$ by processing 10^7 freshly generated instances for ten epochs with a learning rate of 10^{-4}. Finally, the learning rate was dropped to 10^{-5} after processing another 10^7 fresh instances each.

6.3 Result

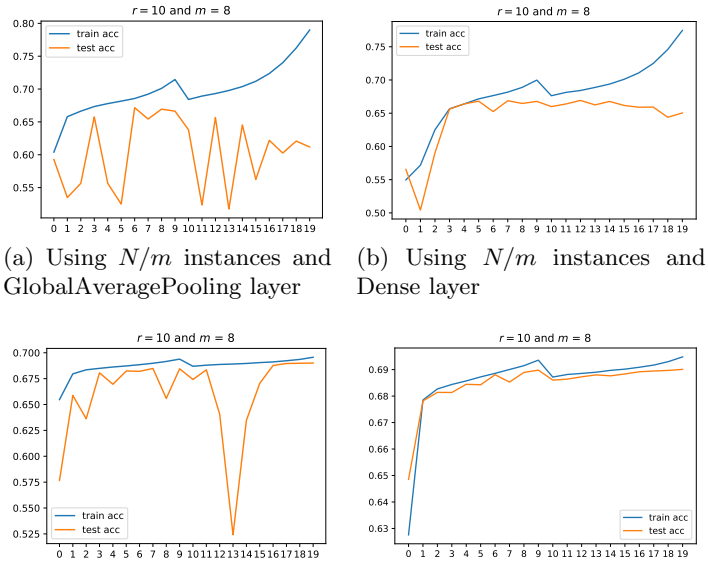
Test Accuracy. We summarize the experimental results of 9-, 10-, 11-,12-round of the differential-neural distinguisher in Table 6. The 9-, 10-, and 11-round distinguisher was trained using the basic training method. A 12-round distinguisher was derived from an 11-round distinguisher using the staged training method. We also use two different numbers of instances to train the differential-neural distinguisher for SIMON32/64, both for the fair comparison of the experiment and to solve the problem of overfitting (refer to Appendix C). Compared to Bao's, the accuracy of our differential-neural distinguisher was significantly improved when the group size m took different values. Also, we trained the differential-neural distinguisher for one more round.

Table 6. Summary accuracy of distinguisher on SIMON32/64 using different number of instances

R	Bao [2]	m=2		m=4		m=8		m=16	
		N/m	N	N/m	N	N/m	N	N/m	N
9	0.6532	0.7251	0.7240	0.7991	0.8095	0.8774	0.8958	0.9344	0.9630
10	0.5629	0.5917	0.5907	0.6239	0.6339	0.6716	0.6900	0.7230	0.7608
11	0.5173	0.5193	0.5240	0.5343	0.5387	0.5441	0.5591	0.5339	0.5878
12	-	-	-	-	-	-	0.5152	-	0.5225

Accuracy fluctuations. The neural network designed for SPECK32/64 to train the differential-neural distinguisher uses multiple fully connected layers as prediction heads, resulting in too many model parameters, making the training time too long. To address this issue, we use a GlobalAveragePooling layer instead of the fully connected layer as the prediction head in the neural network of the SIMON32/64. Using different prediction heads does not significantly impact the accuracy of the differential-neural distinguisher, but it will lead to different fluctuations in the accuracy. In order to study whether the reason for the fluctuation of the accuracy of the distinguisher is the difference in the prediction heads, we used a different number of instances and prediction heads to conduct multiple sets of comparative experiments. From Fig. 12a and 12b, when using N/m instances to train the differential-neural distinguisher, the accuracy of the distinguisher using the fully connected layer fluctuates less, while the accuracy of the distinguisher using the GlobalAveragePooling layer fluctuates more. From Fig. 12c and 12d, when using N instances to train the differential-neural distinguisher, as well as using different prediction heads and still observe fluctuations in accuracy.

Wrong Key Response Profile. The wrong key response profile for our 9-, 10-, 11-, 12-round differential-neural distinguisher is shown in Fig. 13. As we can see from the figure, when the Hamming Distance between the correct key



(a) Using N/m instances and GlobalAveragePooling layer

(b) Using N/m instances and Dense layer

(c) Using N instances and GlobalAveragePooling layer

(d) Using N instances and Dense layer

Fig. 12. Using different number of instances with different prediction head

and the wrong key is smaller, the score of the differential-neural distinguisher is higher, and vice versa, it is smaller. This makes it easier to judge how far the guessed key deviates from the correct key, which shows that our differential-neural distinguisher can effectively distinguish between ciphertext and random numbers.

7 Key Recovery Attack on Round-Reduced Simon32/64

Under a similar framework to the key recovery attack on Speck32/64, the trained differential-neural distinguishers can be prepended with a differential path to perform key recovery attacks for Simon32/64. We improved 16-round and devised the first 17-round differential-neural-distinguisher-based key recovery attacks on Simon32/64. Sadly, due to the low accuracy of the 12-round differential-neural distinguisher and the lack of a sufficient number of generalized neutral bits, we were unable to successfully carry out the 18-round key recovery attack for Simon32/64.

7.1 Key Recovery Attack on 16-round Simon32/64

Experiment 5: The components of key recovery attack $\mathcal{A}^{\text{Simon16}R}$ of 16-round Simon32/64 are shown as follows.

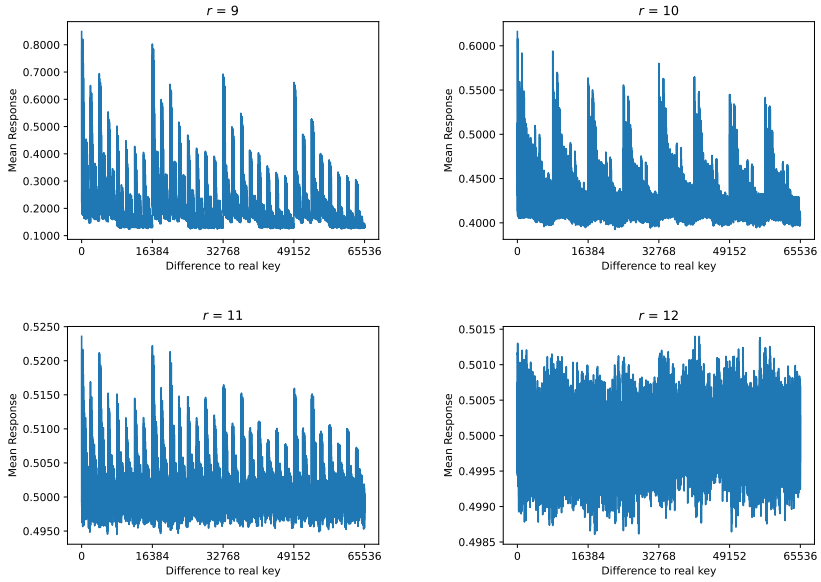


Fig. 13. Wrong key response profile for SIMON32/64 using N instances where $m = 8$

1. 3-round differential path $(0x0440, 0x1000) \rightarrow (0x0000, 0x0040)$;
2. generalized neutral bits of generating multiple-ciphertext pairs: $\{[2], [3], [4]\}$; generalized neutral bits of combined response of differential-neural distinguisher:$\{[6], [8], [9], [10], [18], [22], [0, 24], [12, 26]\}$ (refer to Appendix A.2 Table 11).
3. 11-round differential-neural distinguisher $\mathcal{ND}^{\text{SIMON}_{11R}}$ under difference $(0x00\,00, 0x0040)$ and wrong key response profiles $\mathcal{ND}^{\text{SIMON}_{11R}} \cdot \mu$ and $\mathcal{ND}^{\text{SIMON}_{11R}} \cdot \delta$.
4. 10-round differential-neural distinguisher $\mathcal{ND}^{\text{SIMON}_{10R}}$ under difference $(0x00\,00, 0x0040)$ and wrong key response profiles $\mathcal{ND}^{\text{SIMON}_{10R}} \cdot \mu$ and $\mathcal{ND}^{\text{SIMON}_{10R}} \cdot \delta$.

In the experimental verification of the attack $\mathcal{A}^{\text{SIMON}16R}$, the 11-round and 10-round differential-neural distinguisher was provided in Sect. 6. The goal is to recover the last two subkeys k_{15} and k_{14}. At the beginning, we guess two key bits of k_0, that is $k_0[1]$ and $k_0[3]$, because of the 3-round differential, the conditions for correct pairs are $x_1[1] = x_1'[1] = 0$ and $x_1[3] = x_1'[3] = 0$. Thus, n_{kg} is 2^2. Concrete parameters used in our 16-round key recovery attack $\mathcal{A}^{\text{SIMON}16R}$ are listed as follows.

$n_{kg} = 2^2$	$m = 8$	$n_b = 2^8$	$n_{cts} = 2^8$
$n_{it} = 2^9$	$c_1 = 35, c_2 = 70$	$n_{byit1} = n_{byit2} = 5$	$n_{cand1} = n_{cand2} = 32$

The data complexity is $n_{kg} \times m \times n_{cts} \times n_b \times 2 = 2^{22}$ plaintexts. In total, 25 trials are run, and there are 18 successful trials, for which the returned last two subkeys have a hamming distance to real subkeys of at most two. Thus, the success rate is computed as $^{18}/_{25}$, i.e., 0.72. The average running time of

the experiment is 1168.7s in 25 trials. The time complexity is $n_{kg} \times 2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01 = 2^2 \times 2^{28} \times 1168 \times {}^{500}/_{225} \times \log_{1-0.72} 0.01 = 2^{43.19}$.

7.2 Key Recovery Attack on 17-round Simon32/64

Experiment 6: The components of key recovery attack $\mathcal{A}^{\text{Simon17}R}$ of 17-round Simon32/64 are shown as follows.

1. 4-round differential path $(0x1000, 0x4440) \rightarrow (0x0000, 0x0040)$;
2. generalized neutral bits of generating multiple-ciphertext pairs: $\{[2], [6], [12, 26]\}$; generalized neutral bits of combined response of differential-neural distinguisher:$\{[10, 14, 28]\}$ and five neutral bits conditioned on $x[1, 15], x[15, 13], x[13, 11], x[11, 9], x[9, 7]$ (refer to Appendix A.2 Table 12).
3. 11-round differential-neural distinguisher $\mathcal{ND}^{\text{Simon}_{11}R}$ under difference $(0x0000, 0x0040)$ and wrong key response profiles $\mathcal{ND}^{\text{Simon}_{11}R} \cdot \mu$ and $\mathcal{ND}^{\text{Simon}_{11}R} \cdot \delta$.
4. 10-round differential-neural distinguisher $\mathcal{ND}^{\text{Simon}_{10}R}$ under difference $(0x0000, 0x0040)$ and wrong key response profiles $\mathcal{ND}^{\text{Simon}_{10}R} \cdot \mu$ and $\mathcal{ND}^{\text{Simon}_{10}R} \cdot \delta$.

In the experimental verification of the attack $\mathcal{A}^{\text{Simon17}R}$, the 11-round and 10-round differential-neural distinguisher was provided in Sect. 6. The goal is to recover the last two subkeys k_{16} and k_{15}. At the beginning, we guess two key bits of k_0, that is $k_0[3]$ and $k_0[5]$, because of the 4-round differential, the conditions for correct pairs are $x_1[5] = x_1'[5] = 0$ and $x_1[3] = x_1'[3] = 0$; and six key bits $k_0[1], k_0[15], k_0[13], k_0[11], k_0[9], k_0[7]$ for employing five conditional neutral bits (refer to Appendix A.2 Table 12). Thus, n_{kg} is 2^{2+6}. Concrete parameters used in our 17-round key recovery attack $\mathcal{A}^{\text{Simon17}R}$ are listed as follows.

$n_{kg} = 2^8$	$m = 8$	$n_b = 2^6$	$n_{cts} = 2^{10}$
$n_{it} = 2^{11}$	$c_1 = 15, c_2 = 65$	$n_{byit1} = n_{byit2} = 5$	$n_{cand1} = n_{cand2} = 32$

The data complexity is $n_{kg} \times m \times n_{cts} \times n_b \times 2 = 2^{28}$ plaintexts. In total, 50 trials are running, and there are 4 successful trials, for which the returned last two subkeys have a hamming distance to real subkeys of at most two. Thus, the success rate is computed as ${}^4/_{50}$, i.e., 0.08. The average running time of the experiment is 3356.4s in 50 trials. The time complexity is $n_{kg} \times 2^{28} \times rt \times {}^{500}/_{225} \times \log_{1-sr} 0.01 = 2^8 \times 2^{28} \times 3356 \times {}^{500}/_{225} \times \log_{1-0.08} 0.01 = 2^{54.65}$.

8 Conclusion and Discussions

In this paper, we designed a new network structure to train differential-neural distinguisher, which uses multiple-parallel convolution layers to capture the features of different dimensions of cryptographic algorithms. As a result, we improved the accuracy and obtained the distinguisher with more rounds. Focusing on the problem under the same data distribution, we propose a solution using neutral bits with probability one to generate multiple-plaintext pairs. A combination of improvements has increased success rates and decreased time complexity for key recovery attacks.

Limit to More Rounds of Key Recovery Attack. The differential-neural cryptanalysis mainly consists of the differential path and differential-neural distinguisher. Generalized neutral bits with higher probabilistic and the same differential-neural distinguisher was applied in [2], getting an improved result than [8] on key recovery attack. Now, we have improved the performance of the differential-neural distinguisher and enhanced the key recovery attack. However, it does not reach the full potential of differential-neural distinguisher. On the one hand, for key recovery attacks of 13 rounds SPECK32/64, we only used differential-neural distinguisher when $m = 8$, where the accuracy of distinguisher is only 55.9%. When $m = 16$, the accuracy of our 8-round distinguisher reaches 58.53%. However, we do not have enough generalized neutral bits to launch a key recovery attack when $m = 16$. On the other hand, the accuracy of our 12-round differential-neural distinguisher of SIMON32/64 is lower, and we cannot use it for a key recovery attack. Therefore, we need to make in-depth use of the nature of the structure of the cryptographic algorithm and the more effective structure of the neural network to improve the accuracy of differential-neural distinguisher and search for more generalized neutral bits. More rounds of key recovery attacks can be launched by combining the two aspects.

References

1. AlKhzaimi, H., Lauridsen, M.M.: Cryptanalysis of the SIMON family of block ciphers. IACR Cryptol. ePrint Arch. p. 543 (2013), http://eprint.iacr.org/2013/543
2. Bao, Z., Guo, J., Liu, M., Ma, L., Tu, Y.: Conditional differential-neural cryptanalysis. IACR Cryptol. ePrint Arch. p. 719 (2021)
3. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK families of lightweight block ciphers. IACR Cryptol. ePrint Arch. p. 404 (2013), http://eprint.iacr.org/2013/404
4. Benamira, A., Gérault, D., Peyrin, T., Tan, Q.Q.: A deeper look at machine learning-based cryptanalysis. Lecture Notes in Computer Science, vol. 12696, pp. 805–835. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_28
5. Biham, E., Chen, R.: Near-collisions of SHA-0. In: CRYPTO. Lecture Notes in Computer Science, vol. 3152, pp. 290–305. Springer (2004)
6. Chen, Y., Yu, H.: A new neural distinguisher model considering derived features from multiple ciphertext pairs. IACR Cryptol. ePrint Arch. p. 310 (2021), https://eprint.iacr.org/2021/310
7. Dinur, I.: Improved differential cryptanalysis of round-reduced speck. In: Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 8781, pp. 147–164. Springer (2014)
8. Gohr, A.: Improving attacks on round-reduced speck32/64 using deep learning. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 11693, pp. 150–179. Springer (2019)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778. IEEE Computer Society (2016)
10. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City,

UT, USA, June 18-22, 2018. pp. 7132–7141. Computer Vision Foundation / IEEE Computer Society (2018). https://doi.org/10.1109/CVPR.2018.00745
11. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. pp. 2261–2269. IEEE Computer Society (2017). https://doi.org/10.1109/CVPR.2017.243
12. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (Poster) (2015)
13. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. pp. 1–9. IEEE Computer Society (2015). https://doi.org/10.1109/CVPR.2015.7298594

A Used Generalized Neutral Bit-sets for Key Recovery Attack

A.1 Generalized Neural Bit-sets for Speck32/64

Neutral bit-sets (NB) Used in [8] for 2-round Classical Differential: The signal from distinguisher will rather weak. Gohr boosts it by using $|NB|$ probabilistic neutral bits to create from each plaintext pair. A plaintext structure consisting of $2^{|NB|}$ plaintext pairs that are expected to pass the initial 2 round classical differential together. Concretely, neutral bits are probabilistically neutral are summarized as follows.

Table 7. (Probabilistic) single-bit neutral bit for 2-round Classical Differential $(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$ of Speck32/64 [8]

NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.
[20]	1	[21]	1	[22]	1	[14]	0.965	[15]	0.938	[23]	0.812	[7]	0.806
[30]	0.809	[0]	0.763	[8]	0.664	[24]	0.649	[31]	0.644	[1]	0.574		

Simultaneous-neutral bit-sets (SNBS) used in [2] for 2-round classical differential: for the prepended 2-round differential on top of differential-neural distinguisher, Bao *et al.* [2] can experimentally obtain 3 complete NB and 2 SNBS using exhaustive search. Concretely, for the 2-round differential$(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$, bit and bit-sets that are (probabilistically) (simultaneous-)neutral are summarized in Table 8.

Table 8. (Probabilistic) SNBS for 2-round Classical Differential $(0x0211, 0x0a04) \rightarrow (0x0040, 0x0000)$ of Speck32/64 [2]

NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.	NB	Pr.
[20]	1	[21]	1	[22]	1	[9,16]	1	[2,11,25]	1	[14]	0.965	[15]	0.938
[6,29]	0.91	[23]	0.812	[30]	0.809	[7]	0.806	[0]	0.754	[11,27]	0.736	[8]	0.664

Conditional simultaneous-neutral bit-sets (CSNBS) used in [2] for 3-round classical differential: Bao *et al.* found that there are the three sufficient conditions for a pair $(x, y), (x', y')$ to conform the 3-round differential $(0x8020, 0x4101) \rightarrow (0x0040, 0x0000)$, summarized in Table 9. Concretely, for the 3-round four sub-optimal differential, bit and bit-sets that are (probabilistically) conditional simultaneous-neutral are summarized in Table 10.

Table 9. Three sufficient conditions conform the 3-round sub-optimal differential [2]

$(0x8020, 0x4101)$ $\downarrow$ $(0x0040, 0x0000)$	$(0x8060, 0x4101)$ $\downarrow$ $(0x0040, 0x0000)$	$(0x8021, 0x4101)$ $\downarrow$ $(0x0040, 0x0000)$	$(0x8061, 0x4101)$ $\downarrow$ $(0x0040, 0x0000)$
$x[7] = 0$	$x[7] = 0$	$x[7] = 0$	$x[7] = 0$
$x[5] \oplus y[14] = 1$	$x[5] \oplus y[14] = 0$	$x[5] \oplus y[14] = 1$	$x[5] \oplus y[14] = 0$
$x[15] \oplus y[8] = 0$	$x[15] \oplus y[8] = 0$	$x[15] \oplus y[8] = 1$	$x[15] \oplus y[8] = 1$

Table 10. (Probabilistic) (simultaneous-)neutral bit-sets for 3-round differential $(0x8020, 0x4101) \rightarrow (0x0040, 0x0000)$, $(0x8060, 0x4101) \rightarrow (0x0040, 0x0000)$, $(0x8021, 0x4101) \rightarrow (0x0040, 0x0000)$, $(0x8061, 0x4101) \rightarrow (0x0040, 0x0000)$ of Speck32/ 64 [2]

	(8020,4101)		(8060,4101)		(8021,4101)		(8061,4101)		
Bit-set	Pre.	Post.	Pre.	Post.	Pre.	Post.	Pre.	Post.	Condition
[22]	0.995	1.000	0.995	1.000	0.996	1.000	0.997	1.000	-
[20]	0.986	1.000	0.997	1.000	0.996	1.000	0.995	1.000	-
[13]	0.986	1.000	0.989	1.000	0.988	1.000	0.992	1.000	-
[12,19]	0.986	1.000	0.995	1.000	0.993	1.000	0.986	1.000	-
[14,21]	0.855	0.860	0.874	0.871	0.881	0.873	0.881	0.876	-
[6,29]	0.901	0.902	0.898	0.893	0.721	0.706	0.721	0.723	-
[30]	0.803	0.818	0.818	0.860	0.442	0.442	0.412	0.407	-
[0,8,31]	0.855	0.859	0.858	0.881	0.000	0.000	0.000	0.000	-
[5,28]	0.495	1.000	0.495	1.000	0.481	1.000	0.469	1.000	$x[12] \oplus y[5] = 1$
[15,24]	0.482	1.000	0.542	1.000	0.498	1.000	0.496	1.000	$y[1] = 0$
[6,11,12,18]	0.445	0.903	0.456	0.906	0.333	0.701	0.382	0.726	$x[2] \oplus y[11] = 0$
[4,27,29]	0.672	0.916	0.648	0.905	0.535	0.736	0.536	0.718	$x[11] \oplus y[4] = 1$

A.2 Generalized Neural Bit-sets for Simon32/64

For an input pair $((x, y), (x', y'))$ to conform the 3-round differential $(0x0440, 0x1000) \rightarrow (0x0000, 0x0040)$, one has conditions that

$$\begin{cases} x[1] = x'[1] = 0 \\ x[3] = x'[3] = 0 \end{cases}$$

Table 11. NB and SNBS for 3-round Classical Differential $(0x0440, 0x1000) \rightarrow (0x0000, 0x0040)$ of Simon32/64 [2]

[2]	[3]	[4]	[6]	[8]	[9]
[10]	[18]	[22]	[0,24]	[12,26]	

For an input pair $((x, y), (x', y'))$ to conform the 4-round differential $(0x1000, 0x4440) \to (0x0000, 0x0040)$, one has conditions that

$$\begin{cases} x[5] = x'[5] = 0 \\ x[3] = x'[3] = 0 \end{cases}$$

Table 12. CSNBS for 4-round Classical Differential $(0x1000, 0x4440) \to (0x0040, 0x0000)$ of SIMON32/64 [2]

Bit-set	C.	Bit-set	C.	Bit-set	C.	Bit-set	C.	Bit-set	C.
$x[1, 15]$		$x[15, 13]$		$x[13, 11]$		$x[11, 9]$		$x[9, 7]$	
[24,10]	00	[22,8]	00	[20]	00	[18,4]	00	[16,8]	00
[24,10,9]	10	[22,8,7]	10	[20,5]	10	[18,4,3]	10	[16,8,1]	10
[24,10,0]	01	[22,8,14]	01	[20,12]	01	[18,4,10]	01	[16]	01
[24,10,9,0]	11	[22,8,7,14]	11	[20,12,5]	11	[18,4,10]	11	[16,1]	11

C.: Condition on $x[i, j]$, e.g., $x[i, j] = 10$ means $x[i] = 1$ and $x[j] = 0$.

B procedure of $(1 + s + r + 1)$-round key recovery attack

The attack procedure is as follows.

1. Initialize variables $Gbest_{key} \leftarrow$ (None, None), $Gbest_{score} \leftarrow -\infty$.
2. For each of the n_{kg} guessed key bits, on which the conditions depend,
   (a) Generate n_{cts} random data with difference ΔP, and satisfying the conditions being conforming pairs (refer to Appendix A).
   (b) Using n_{cts} random data and $\log_2 m$ neutral bit with probability one to generate n_{cts} data pairs. Every data pairs have m data.
   (c) From the n_{cts} random data pairs, generate n_{cts} structures using n_b generalized neutral bit.
   (d) Decrypt one round using zero as the subkey for all data in the structures and obtain n_{cts} plaintext structure.
   (e) Query for the ciphertexts under $(1+s+r+1)$-round SPECK or SIMON of the $n_{cts} \times n_b \times 2$ plaintext structures, thus obtain n_{cts} ciphertext structures, denoted by $\{\mathcal{C}_1, \ldots, \mathcal{C}_{n_{cts}}\}$.
   (f) Initialize an array $\omega_{\max}$ and an array n_{visit} to record the highest distinguisher score obtained so far and the number of visits have received in the last subkey search for the ciphertext structures.
   (g) Initialize variables $best_{score} \leftarrow -\infty$, $best_{key} \leftarrow$ (None, None), $best_{pos} \leftarrow$ None to record the best score, the corresponding best recommended values for the two subkeys obtained among all ciphertext structures and the index of this ciphertext structures.
   (h) For j from 1 to n_{it}:
      i. Compute the priority of each of the ciphertext structures as follows: $s_i = \omega_{\max i} + \alpha \cdot \sqrt{\log_2 j / n_{\text{visit}i}}$, for $i \in \{1, \ldots, n_{cts}\}$, and $\alpha = \sqrt{n_{cts}}$; The formula of priority is designed according to a general method in reinforcement learning for achieving automatic exploitation versus exploration trade-off based on *Upper Confidence Bounds*. It is motivated to focus the key search on the most promising ciphertext structures [8].
      ii. Pick the ciphertext structure with the highest priority score for further processing in this j-th iteration, denote it by $\mathcal{C}$, and its index by idx, $n_{\text{visit}idx} \leftarrow n_{\text{visit}idx} + 1$.
      iii. Run BAYESIANKEYSEARCH Algorithm [8] with $\mathcal{C}$, the r-round differential-neural distinguisher $\mathcal{ND}^r$ and its wrong key response profile $\mathcal{ND}^r \cdot \mu$ and $\mathcal{ND}^r \cdot \sigma$, n_{cand1}, and n_{byit1} as input parameters; obtain the output, that is a list L_1 of $n_{byit1} \times n_{cand1}$ candidate values for the last subkey and their scores, i.e., $L_1 = \{(g_{1i}, v_{1i}) : i \in \{1, \ldots, n_{byit1} \times n_{cand1}\}\}$.
      iv. Find the maximum $v_{1\max}$ among v_{1i} in L_1, if $v_{1\max} > \omega_{\max idx}$, $\omega_{\max idx} \leftarrow v_{1\max}$.
      v. For each of recommended last subkey $g_{1i} \in L_1$, if the score $v_{1i} > c_1$,
         A. Descrypt the ciphertext in $\mathcal{C}$ using the g_{1i} by one round and obtain the ciphertext structures $\mathcal{C}'$ of $(1+s+r)$-round SPECK or SIMON.

B. Run BayesianKeySearch Algorithm with $\mathcal{C}'$, the differential-neural distinguisher $\mathcal{ND}^{r-1}$ and its wrong key response profile $\mathcal{ND}^{r-1} \cdot \mu$ and $\mathcal{ND}^{r-1} \cdot \sigma$, n_{cand2}, and n_{byit2} as input parameters; obtain the output, that is a list L_2 of $n_{byit2} \times n_{cand2}$ candidate values for the last subkey and their scores, i.e., $L_2 = \{(g_{2i}, v_{2i}) : i \in \{1, \ldots, n_{byit2} \times n_{cand2}\}\}$.

C. Find the maximum v_{2i} and the corresponding g_{2i} in L_2, and denote them by $v_{2\max}$ and $g_{2\max}$.

D. If $v_{2\max} > best_{score}$, update $best_{score} \leftarrow v_{2\max}$, $best_{key} \leftarrow (g_{1i}, g_{2\max})$, $best_{pos} \leftarrow idx$.

vi. If $best_{score} > c_2$, go to Step 2i.

(i) Make a final improvement using VerifierSearch [8] on the value of $best_{key}$ by examining whether the scores of a set of keys obtained by changing at most 2 bits on top of the incrementally updated $best_{key}$ could be improved recursively until no improvement obtained, update $best_{score}$ to the best score in the final improvement; If $best_{score} > Gbest_{score}$, update $Gbest_{score} \leftarrow best_{score}$, $Gbest_{key} \leftarrow best_{key}$.

3. Return $Gbest_{key}, Gbest_{score}$.

C Overfitting in training differential-neural distinguisher with N/m and N instances

C.1 train differential-neural distinguisher for 6-round Speck32/64

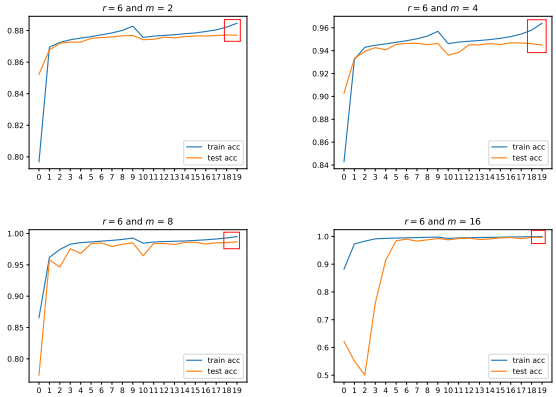


Fig. 14. Train differential-neural distinguisher with N/m instances for 6-round Speck32/64

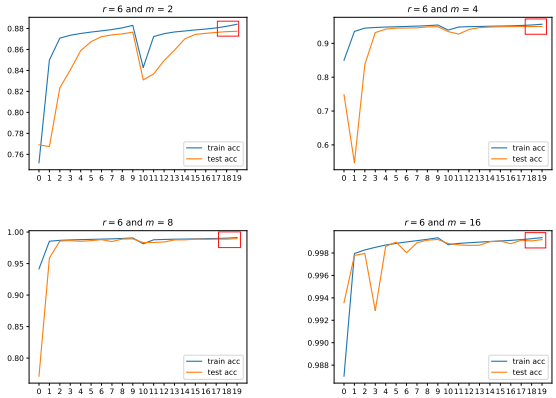


Fig. 15. Train differential-neural distinguisher with N instances for 6-round Speck32/64

C.2 train differential-neural distinguisher for 7-round SPECK32/64

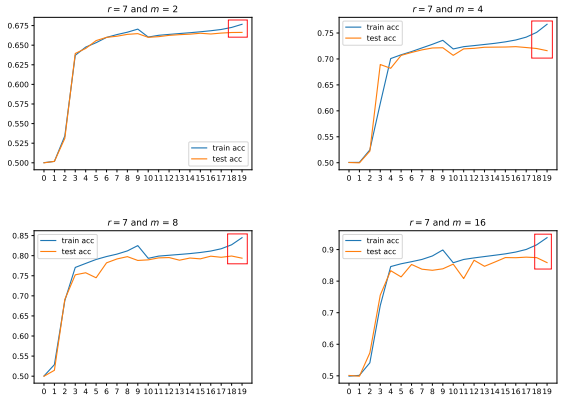


Fig. 16. Train differential-neural distinguisher with N/m instances for 7-round SPECK32/64

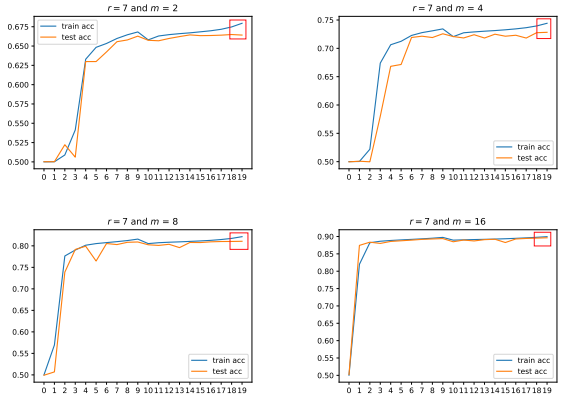


Fig. 17. Train differential-neural distinguisher with N instances for 7-round SPECK32/64

C.3 train differential-neural distinguisher for 9-round SIMON32/64

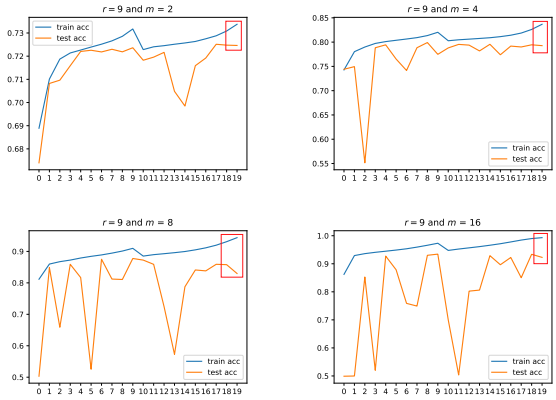


Fig. 18. Train differential-neural distinguisher with N/m instances for 9-round SIMON32/64

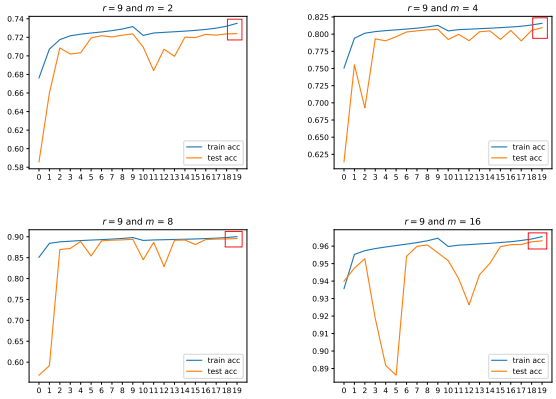


Fig. 19. Train differential-neural distinguisher with N instances for 9-round SIMON32/64

C.4 train differential-neural distinguisher for 10-round SIMON32/64

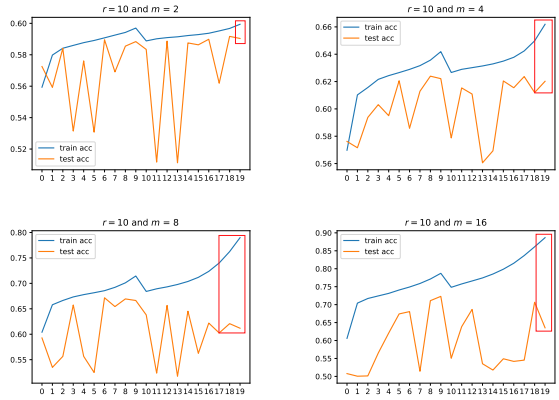


Fig. 20. Train differential-neural distinguisher with N/m instances for 10-round SIMON32/64

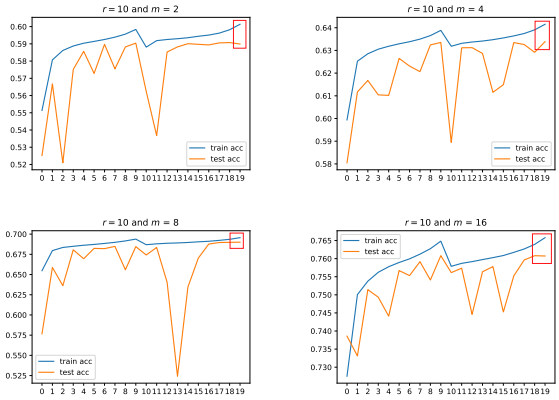


Fig. 21. Train differential-neural distinguisher with N instances for 10-round SIMON32/64

C.5 train differential-neural distinguisher for 11-round SIMON32/64

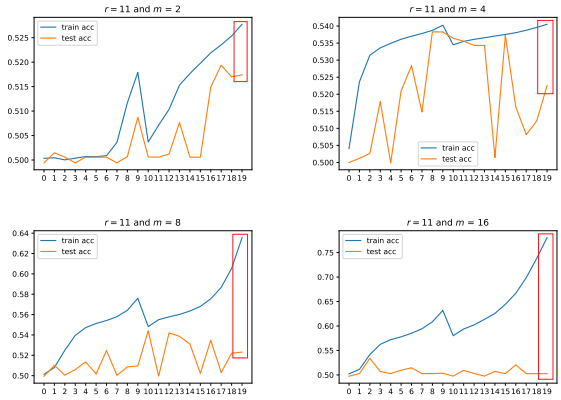


Fig. 22. Train differential-neural distinguisher with N/m instances for 11-round SIMON32/64

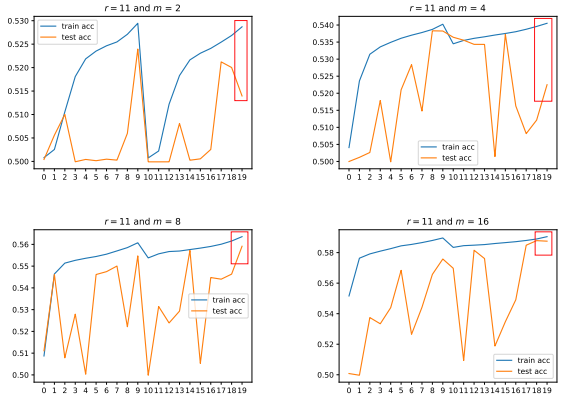


Fig. 23. Train differential-neural distinguisher with N instances for 11-round SIMON32/64