A New Neural Distinguisher Model Considering Derived Features from Multiple Ciphertext Pairs *

Yi Chen[1] and Hongbo Yu[2]

[1] Department of Computer Science and Technology, Tsinghua University
chenyi19@mails.tsinghua.edu.cn
[2] Department of Computer Science and Technology, Tsinghua University
yuhongbo@mail.tsinghua.edu.cn

Abstract. Gohr has proposed the only deep learning-based distinguisher model at Crypto 2019, which is used to distinguish reduced Speck32/64 and a pseudorandom permutation. This distinguisher model can be applied to many symmetric ciphers. Given a plaintext differential, Gohr's distinguisher model can learn differences between two distributions from adequate single ciphertext pairs.
In this paper, we propose a new neural distinguisher model which takes $k \geqslant 2$ ciphertext pairs as the analysis object. A non-uniform distribution can produce many derived features that will not appear in a single ciphertext pair. Our neural distinguisher model can exploit these derived features from k ciphertext pairs. Taking Gohr's distinguisher model as the baseline model, we firstly construct strong baseline distinguishers for five reduced ciphers. Then our neural distinguishers for five ciphers are also constructed using the new distinguisher model proposed in this paper. Experiments show our neural distinguishers can always obtain distinguishing accuracy promotions under various settings of k. When combining k samples incorrectly classified by baseline distinguishers into one group, our neural distinguishers can still distinguish correctly with a non-negligible probability. It indicates that derived features have been successfully captured by our neural distinguishers. The distinguishing accuracy promotion also comes from derived features. Our neural distinguishers can also be used to improve the key recovery attack on 11-round Specck32/64.
Besides, compared with the raw attack scheme provided by Gohr, we propose a new key recovery attack scheme which can further reduce the time complexity.

Keywords: Neural distinguisher · Differential cryptanalysis · Deep learning · Data reuse

1 Introduction

Target of this paper. In the field of cryptanalysis, neural distinguisher is a newly introduced deep learning-based tool [12]. Given a specific plaintext differ-

* This paper was first submitted to Eurocrypt 2021 on October 8, 2020.

ential α and a target cipher $Enc_{key}(P)$, if $(C_1, C_2) = (Enc_{key}(P), Enc_{key}(P \oplus \alpha))$ obey nonuniform distribution on some features such as the differential, a well trained neural network can distinguish this distribution from the uniform distribution. The neural network can be used as a neural distinguisher.

Except for the differential feature, neural distinguisher can capture some unknown features of the ciphertext pair, which has been proved in [12]. It ensures that neural distinguishers are more powerful than pure differential distinguisher under the same number of rounds. As a deep learning based distinguisher, its distinguishing accuracy plays a vital role in attacks on a cipher [12]. Therefore, improving the performance of neural distinguishers is an important topic in deep learning based cryptanalysis.

In this paper, our target is to propose a new neural distinguisher model which can also be applied to any symmetric ciphers. It's expected that neural distinguishers based on our distinguisher model can always obtain better performance than baseline distinguishers.

Inspiration from existed differential cryptanalysis and artificial intelligence community. In the last decades, various distinguishers such as single differential distinguisher [5], linear distinguisher [22] are proposed to analyze the security bound of ciphers. All these distinguishers are based on a specific analysis unit and a certain feature. For example, a ciphertext pair is the analysis unit and differential is the feature for the single differential distinguisher.

In order to enhance the capability of these basic distinguishers, derived features of the distribution of the chosen feature can be exploited to design novel distinguishers. Multiple differential distinguisher [29], truncated differential distinguisher[26], high order differential distinguisher [20] are all the representatives that exploit features derived from multiple samples drawn from the same differential distribution.

Such a strategy above can also be applied to the neural distinguisher. Although features learned by the neural distinguisher are unknown, we are sure that derived features can be obtained from the nonrandomness of these features. Here a simplified example can well explain our opinion. Take the differential between a ciphertext pair $\beta = C_0 \oplus C_1$ as an example. Denote the differential distribution resulted from a cipher as the target distribution, a uniform distribution is the reference distribution.

Assuming that the target distribution is a Gaussian distribution as Figure 1 shows, there are some derived features that don't arise in a single sample. The average distance of two randomly generated samples is representative:

$$\begin{cases} d_p = |\beta_1^p - \beta_2^p| \\ d_n = |\beta_1^n - \beta_2^n| \end{cases} \tag{1}$$

where β_1^p, β_2^p, β_1^n, β_2^n are random samples from two distributions. It's expected that $d_p < d_n$. Similar derived features can be captured by the neural network, which are very useful for helping classification. The extreme scenario is that even k samples are all wrongly classified based on features of a ciphertext pair, we can recognize the right distribution with the help of derived features from k

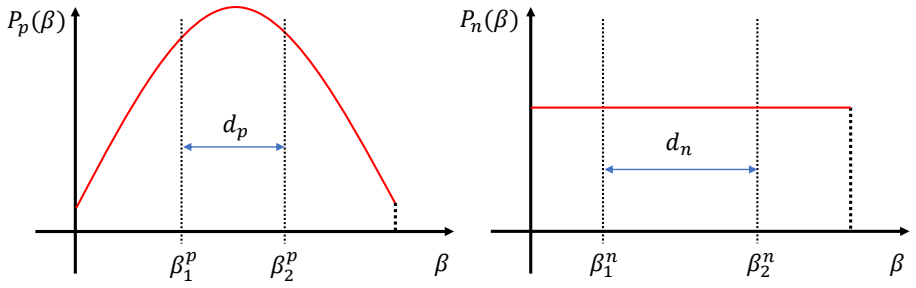

Fig. 1. $P_p(\beta)$ is the target difference probability distribution which is a Gaussian distribution. $P_n(\beta)$ is reference difference probability distribution which is a uniform distribution. d_p stands for the distance between two random samples β_1^p, β_2^p. d_n stands for the distance between two random samples β_1^n, β_2^n.

samples. This is verified through two tests denoted as *False Positive Test* and *False Negative Test*, which are proposed in Section 3.2.

Deep learning has been applied to various fields, such as computer vision [17][19], natural language processing [4][25], and health medical [8][10]. An important target of these topics is promoting the performance of neural networks. Providing more features that are related to the task is one of the most powerful methods to achieve this target.

For example, depth map estimation [21] is a regression problem, which aims to estimate the true distance of the object to the camera from the image. Except for raw color images, related features such as traditional stereo knowledge [30], motion features [23] are extracted and fed into the neural network together. Action recognition [28] is a classification problem, which needs to recognize what action the actor is performing from images or videos. Similarly, extra depth maps [31] or abstract representations [9] of postures are input to neural networks together.

The same mechanism behind various methods above is adjusting the input to the neural networks. If there are more derived features that are related to the task, neural networks usually can successfully capture them for promoting the performance significantly. Our new neural distinguisher model is also based on this common property.

Our contributions. Proposing a new neural distinguisher model that can always obtain performance promotion compared to Gohr's distinguisher model is the main contribution of this paper. According to the introduction of related research about traditional differential cryptanalysis, derived features can be generated from k ciphertext pairs as long as the distribution of a ciphertext pair isn't a uniform distribution. Research in the artificial intelligence community also proves that neural networks are likely to learn extra features from k samples. Thus our new neural distinguisher model aims at exploiting derived features from k ciphertext pairs. Our work in this paper contains the following aspects:

Propose a new neural distinguisher model and construct neural distinguishers for five reduced symmetric ciphers. Taking Gohr's dis-

tinguisher model[12] as the strong baseline, we have constructed neural distinguishers for five reduced symmetric ciphers using the baseline and our new distinguisher model respectively. Experiments show that our new neural distinguishers can always promote the distinguishing accuracy under various settings of the group size k.

Design false negative test and false positive test. In order to figure out whether derived features have been captured by our neural distinguishers, we have designed false negative test and false positive test. We focus on samples that are wrongly classified by baseline distinguishers. By taking k false negative or false positive samples as a group, we can observe whether our new neural distinguishers can correctly classify these samples with a non-negligible probability. Experiments show that our neural distinguishers have successfully captured derived features, which result in the distinguishing accuracy promotion.

Perform key recovery attacks on 11-round Speck32/64. With the help of a data reuse strategy proposed in Section 5, we have performed the same key recovery attack on 11-round Speck32/64 in [12] using our neural distinguishers. A success rate of 54.4% that is higher than 52% in [12] is obtained. In fact, if the distinguishing accuracy is greatly promoted, the complexity of key search cal also be reduced.

Except for these works, We also find an output signal clustering phenomenon that is widespread in deep learning based distinguishers. It can directly visualize the ability of neural distinguishers to distinguish the correct key from the wrong keys. We have applied it to neural distinguishers for reduced Speck32/64. Besides, we propose a new key recovery attack scheme based on this phenomenon. It can further reduce the time complexity of the attack.

Organization. In section 2 and section 3, the overviews of the baseline model and our new distinguisher model are given respectively. In section 4, neural distinguishers for five symmetric ciphers are constructed using two distinguisher models. False positive tests and false negative tests are also performed. In section 5, a data reuse strategy for reducing data complexity is proposed. In section 6, we first describe the output signal clustering phenomenon. Then key recovery attacks on 11-round Speck32/64 using two attack schemes are given.

2 Notations and overview of Gohr's Distinguisher Model

To make it easier to read this paper, we first list the major notations. Then an overview of Gohr's distinguisher model proposed in [12] is given.

2.1 Notations

P, C, X	Plaintext, Ciphertext, A ciphertext pair Even there is a subscript, the meaning doesn't change
α	Plaintext differential
N, M	The number of ciphertext pairs
BD	Baseline distinguishers built using Gohr's distinguisher model
$ND_{k=?}$	Our new neural distinguishers. The group size is $k =?$
Z	The output of a neural distinguisher
c_i	The key rank score threshold of the i_{th} round for key recovery
R	The number of reduced rounds

2.2 overview of Gohr's Distinguisher Model

Given a plaintext pair (P_1, P_2) and a target cipher, the resulting ciphertext pair (C_1, C_2) is regarded as a sample. Each sample will be attached a label Y:

$$Y(C_1, C_2) = \begin{cases} 1, if\ P_1 \oplus P_2 = \alpha \\ 0, if\ P_1 \oplus P_2 \neq \alpha \end{cases} \tag{2}$$

If Y is 1, it means this sample is sampled from the target distribution. If Y is 0, it means this sample is sampled from a uniform distribution. For convenience, a sample with label 1 is named a positive sample. Otherwise, we call it a negative sample. A neural network is trained over enough positive and negative samples. If the neural network can obtain a stable distinguishing accuracy higher than 0.5 on a testing set, we say this cipher is not a pseudorandom function. The neural network can be used as a powerful neural distinguisher. Using the model above, Gohr has obtained distinguishers on Speck32/64 reduced to 5/6/7 rounds.

There is only one neuron in the output layer. And its output is regarded as the posterior probability that the ciphertext pair is drawn from the target distribution. Thus, Gohr's neural distinguisher model can be described as

$$\begin{gathered} Pr\left(Y = 1 \,|X\right) = F_1\left(f\left(X\right)\right) \\ X = (C_1, C_2) \\ Pr\left(Y = 1 \,|X\right) \in [0, 1] \end{gathered} \tag{3}$$

where $f\left(X\right)$ stands for features captured from the ciphertext pair X, and $F_1\left(\cdot\right)$ is the posterior probability estimation function. It's worth noticing that only basic features of a single ciphertext pair are exploited in Gohr's model. In this paper, we take Gohr's model as a strong baseline model. Neural distinguishers built using the baseline model are used as baseline distinguishers, which can help assess our new neural distinguisher model.

3 Our Neural Distinguisher Model

In this section, our new neural distinguisher model is described firstly. Then *false negative test* and *false positive test* are designed for the model verification.

At last, the architecture of the neural network for implementing our neural distinguisher model is introduced. Parameters of this architecture can be adjusted adaptively.

3.1 New Distinguisher Model

Our new distinguisher model takes a ciphertext group consisting of multiple ciphertext pairs as the input, which can be described as:

$$\begin{gathered} Pr\left(Y=1\left|X_1,\cdots,X_k\right.\right)=F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) \\ X_i=\left(C_{i,1},C_{i,2}\right),i\in[1,k] \\ Pr\left(Y=1\left|X_1,\cdots,X_k\right.\right)\in[0,1] \end{gathered} \tag{4}$$

where $f\left(X_i\right)$ represents the basic features of a ciphertext pair X_i, $\varphi\left(\cdot\right)$ is the derived features, and $F_2\left(\cdot\right)$ is the new posterior probability estimation function. Derived features from k ciphertext pairs are extracted from the distribution of basic features.

Neural distinguishers based on our new distinguisher model can be obtained by following three processes:

1. **Data Generation:** A ciphertext group consisting of k ciphertext pairs $\left(C_{1,1}^i,C_{1,2}^i,\cdots,C_{k,1}^i,C_{k,2}^i\right)$ is regarded as a sample. Given a plaintext differential α, each sample will be attached a label Y according corresponding plaintexts:
$$Y=\begin{cases}1,if\ P_{j,1}\oplus P_{j,2}=\alpha,j\in[1,k]\\0,if\ P_{j,1}\oplus P_{j,2}\neq\alpha,j\in[1,k]\end{cases} \tag{5}$$
Similarly, if the label is 1, the ciphertext group is denoted as a positive sample. Otherwise it's denoted as a negative sample. A training set is composed of $\frac{N}{2k}$ positive samples and $\frac{N}{2k}$ negative samples. A testing set is composed of $\frac{M}{2k}$ positive samples and $\frac{M}{2k}$ negative samples.
2. **Training:** Train a neural network on the training dataset. If the training accuracy is not larger than 0.5, choose a different α and start from the data generation process again. Or save the trained neural network and perform the testing process.
3. **Testing:** Test the distinguishing accuracy of the trained neural network on the testing dataset. If the test accuracy is larger than 0.5, return the neural network as a successful neural distinguisher. Or choose a different α and start from the data generation process again.

3.2 Model Verification Tests

In order to verify whether our neural distinguisher based on the new distinguisher model can learn derived features from k ciphertext pairs. Two auxiliary tests named as false negative test and false positive test are designed as follows.

False Negative Test (FNT): If k ciphertext pairs with label 1 are all wrongly classified by the baseline distinguisher:

$$\begin{aligned} p\left(Y=1\left|X_1\right.\right) &= F_1\left(f\left(X_1\right)\right) < 0.5 \\ &\vdots \\ p\left(Y=1\left|X_k\right.\right) &= F_1\left(f\left(X_k\right)\right) < 0.5 \end{aligned} \tag{6}$$

such ciphertext pairs are false negative samples. It means that the features of a single ciphertext pair $f\left(X_i\right)$ can't help correct classification. These k samples are combined into a ciphertext group and fed into our neural distinguisher. The output of our distinguisher is:

$$p\left(Y=1\left|X_1,\cdots,X_k\right.\right) = F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) \tag{7}$$

Generate a large number of such ciphertext groups and feed them to our neural distinguisher. What we care about is the following pass ratio

$$F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) \geqslant 0.5 \tag{8}$$

The classification is totally determined by $\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)$ now. The final pass ratio under such a setting can show whether derived features have been learned and their effects. If our neural distinguisher can obtain a non-negligible pass ratio, then $\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)$ can offset the negative influence of $f\left(X_i\right), i \in [1,k]$. If the pass ratio is high, derived features from k ciphertext pairs play a vital role in classification for this kind of ciphertext pairs.

False Positive Test (FPT): Similarly, if k ciphertext pairs with label 0 are wrongly classified by the baseline model

$$\begin{aligned} p\left(Y=1\left|X_1\right.\right) &= F_1\left(f\left(X_1\right)\right) \geqslant 0.5 \\ &\vdots \\ p\left(Y=1\left|X_k\right.\right) &= F_1\left(f\left(X_k\right)\right) \geqslant 0.5 \end{aligned} \tag{9}$$

such ciphertext pairs are false positive samples. These k samples are combined into a ciphertext group and fed into our neural distinguisher. Now what we care about is the following pass ration

$$F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) < 0.5 \tag{10}$$

The pass ratios of **FNT** and **FPT** are another two important indicators that can show the superiority of our distinguisher model. In the following experiments, the distinguishing accuracy and these two pass ratios are used to assess our neural distinguishers together.

3.3 Generic Network Architecture

Figure 2 shows the neural network architecture for implementing the new distinguisher model. The network architecture contains several modules that are

described in Figure 2. The input consisting of k ciphertext pairs is arranged in a $k \times w \times \frac{2L}{w}$ array. L represents the block size of the target cipher and w is the size of a basic unit. For example, L is 32 and w is 16 for speck32/64. Based on this input, the new distinguisher model can be implemented in two parts. First, the learning of a single ciphertext pair's basic features is accomplished by module 1. The kernel size is 1×1, which can capture basic features efficiently. Second, the learning of derived features and posterior probability estimation functions are combined in a part. The two-dimensional filters with a size of $K_s \times K_s$ can learn derived features from k ciphertext pairs. Such an architecture obeys the model of our neural distinguisher completely.

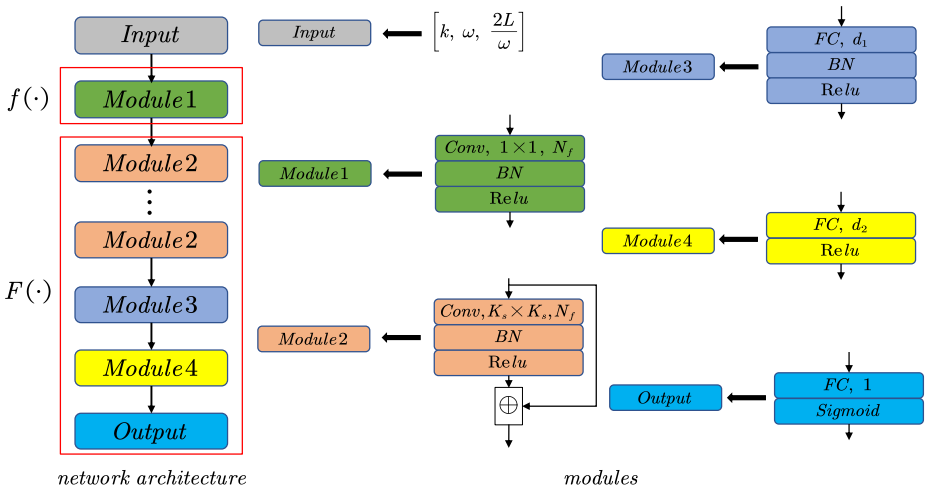

Fig. 2. The network architecture of our neural distinguishers. *Conv* stands for a convolution layer with N_f filters. The size of each filter is $K_s \times K_s$. Module 2 also adopts the skip connection [13]. *FC* is a fully-connected layer which has d_1 or d_2 neurons. *BN* is batch normalization. *Relu* and *Sigmoid* are two different activation functions. The output of *Sigmoid* ranges from 0 to 1.

The number of module 2 is set 1. Such a setting can greatly reduce the number of neural network parameters. From the perspective of deep learning, it can limit the neural network's capability. This allows a fair comparison between our new distinguisher model and the baseline model.

Training Pipeline: The neural network is trained for E_s epochs with a batch size of B_s. The cyclic learning rate scheme in [12] is adopted. Optimization is performed against the following loss function:

$$loss = \sum_{i=1}^{\frac{N}{k}} (Y_{i,p} - Y_i)^2 + \lambda \times \|W\| \tag{11}$$

where $Y_{i,p}$ is the output of the neural distinguisher, Y_i is the true label, W is the parameters of the neural network, and λ is the penalty factor. The Adam algorithm [18] with default parameters in Keras [11] is applied to the optimization.

4 Neural Distinguishers for Five Reduced Symmetric Ciphers

The distinguishing accuracy is the most important indicator which reflects the performance of neural distinguishers. To verify whether our distinguisher model can achieve performance improvements on as many types of symmetric ciphers as possible, neural distinguishers for five different ciphers are constructed in this section. The concrete properties of these ciphers are shown in Table 1.

Table 1. Five tested ciphers in this paper

Ciphers	Nonlinear Components			Types		
	Sbox	modulo addition	logic operation	Block Cipher	Hash	Mac
Speck		✓		✓		
Chaskey		✓				✓
Present	✓			✓		
Des	✓			✓		
Keccak			✓		✓	

The distribution of a feature such as the differential feature is also mainly affected by nonlinear components. Almost all the non-linear components currently used in symmetric ciphers are included in Table 1. So this is enough to verify whether the new distinguisher model is generic and effective.

For a cipher reduced to T rounds, a specific plaintext differential α is set firstly. Then a training set and a test set are randomly generated. After sufficient training, we will observe the testing accuracy of the obtained new neural distinguisher. An important parameter related to our neural distinguisher is the group size k. In subsequent experiments, the group size k has four options $\{2, 4, 8, 16\}$. Other parameters related to the training and network architecture of our neural distinguisher are posted in Table 2. Baseline distinguishers based on Gohr's distinguisher model are constructed with parameters given in [12].

Table 2. Related parameters for constructing neural distinguishers based on our distinguisher model

N_f	d_1	d_2	K_s	B_s
32	64	64	3	500
λ	L_r	E_s	N	M
10^{-5}	$0.02 \rightarrow 0.001$	10	10^7	10^6

4.1 Experiments on Speck32/64

Neural distinguishers for reduced Speck32/64: Distinguishers for reduced speck32/64 [3] built based on the baseline model have been already reported. Related distinguishing accuracies will be posted directly. In order to assess our new distinguisher model, new neural distinguishers are also built for Speck32/64 reduced to 5, 6, and 7 rounds respectively.

Let $N = 10^7$ and $M = 10^6$. The plaintext differential is $\alpha = (0x0040, 0)$ introduced in [1]. The accuracy comparison is presented on Table 3.

Table 3. Distinguishing accuracy of neural distinguishers for 5/6/7 rounds speck32/64.

R	BD	Our neural distinguishers			
		$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
5	0.929	0.9738	0.991	0.9992	0.9999
6	0.788	0.8613	0.931	0.9562	0.9802
7	0.616	0.6393	0.6861	0.7074	0.6694

Compared with baseline distinguishers, our neural distinguishers can obtain significant accuracy promotion despite the value of k. When Speck32/64 is reduced to 6 or 5 rounds, the distinguishing accuracy of our distinguishers can be further promoted by increasing k. This phenomenon will be discussed in later tests.

Although the distinguishing accuracy comparisons are sufficient to verify the superior performance of our new distinguishers, we can still have higher expectations about our neural distinguishers. The number of training samples for baseline distinguishers is N while it is $\frac{N}{k}$ for our neural distinguishers. Thus the performance of our neural networks may be likely improved by training on more ciphertext groups.

False negative/positive test: With the help of baseline distinguishers, $10000\times k$ false negative and false positive samples are randomly generated respectively. Then **FNT** and **FPT** are performed on 10000 ciphertext groups respectively. Table 4 has shown corresponding test results.

Table 4. Pass ratios of false negative test and false positive test.

R	False Negative Test				False Positive Test			
	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
5	0.0112	0.0013	0.0001	0	0.3068	0.6268	0.6748	0.7228
6	0.0331	0.0143	0.0081	0.0048	0.1519	0.1432	0.3723	0.4375
7	0.0511	0.0212	0.0283	0.0917	0.0659	0.0233	0.0157	0.0691

Even if all the k ciphertext pairs are regarded as the opposite class by baseline distinguishers, our neural distinguishers can correctly classify these false negative/positive samples with a non-negligible or high probability. It proves that our neural distinguishers have captured derived features from k ciphertext pairs. These derived features can offset the negative influence of a single ciphertext pair's features.

If the target distribution is very different from the uniform distribution, there would be more derived features among ciphertext pairs by increasing k. Due to such an advantage, the distinguishing accuracy of our distinguishers can be further promoted by increasing the group size k.

Besides, the two tests can provide more information. For example, the success rate of **FPT** is far higher than the success rate of **FNT** when $R = 5, 6$. Thus derived features from k ciphertext pairs learned by our neural distinguishers are mainly helpful for recognizing the uniform distribution.

4.2 Experiments on Chaskey

Neural distinguishers for reduced Chaskey: Based on the best differential path searched in [27], baseline distinguishers are built for reduced Chaskey firstly. Given the plainext difference $\alpha = (0x8400, 0x0400, 0, 0)$, the baseline distinguisher can distinguish chaskey up to 4 rounds.

Our neural distinguishers are also built for Chaskey reduced to 3, 4 rounds. All related parameters are the same with Table 2, except for the penalty factor λ is increased to 10^{-4}. A larger penalty factor can reduce the complexity of the neural network, which can help obtain a stable distinguisher when the ratio of sampled data to the entire ciphertext pair space is too small. The distinguishing accuracy comparisons are presented on Table 5.

Table 5. Distinguishing accuracy of neural distinguishers for 3/4 rounds Chaskey

R	BD	Our neural distinguishers			
		$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
3	0.8608	0.8958	0.9583	0.9887	0.9986
4	0.6161	0.6589	0.6981	0.7603	0.7712

Compared with baseline distinguishers, our neural distinguishers can obtain much higher accuracy even if the ciphertext group size k is 2. Besides, the distinguishing accuracy of our neural distinguishers can be further improved by increasing k.

Speck is a block cipher while Chaskey is a Message Authentication Code (MAC) algorithm. These two ciphers have the same nonlinear components. Our neural distinguishers can both achieve the target of promoting distinguishing accuracy despite the concrete encryption process. Thus it can be inferred that our distinguisher model can be applied to more symmetric ciphers which are constructed based on modulo addition.

False negative/positive test: Similarly, $10^4 \times k$ false negative and false positive samples are randomly generated first. Table 6 shows the pass ratios of **FNT** and **FPT** on reduced Chaskey.

Table 6. Pass ratios of false negative test and false positive test

R	False Negative Test				False Positive Test			
	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
3	0.1156	0.0635	0.0373	0.0087	0.4027	0.4032	0.3976	0.4705
4	0.1412	0.1749	0.1481	0.1675	0.8369	0.7439	0.7298	0.5591

Based on these two tests above, our neural distinguishers can correctly classify these false negative/positive samples with a non-negligible or high probability. Thus we can obtain the same conclusion which has been proved in tests for reduced Speck32/64. Our neural distinguishers can successfully capture derived features that can help promote the distinguishing accuracy. Compared with the baseline distinguisher for Chaskey reduced to 4 rounds, the accuracy of our neural distinguisher with $k = 2$ is only a little higher. However, our neural distinguisher can obtain 0.8369 and 0.1412 accuracies on two tests respectively. Based on our test setting, only derived features among k ciphertext pairs can make

positive influence on the classification. Thus, it further proves the superiority of our distinguisher model.

If we observe Table 4 and Table 6 at the same time, we can find a similar property of our neural distinguishers for reduced Speck32/64 and Chaskey. Compared with **FNT**, our neural distinguisher can both obtain higher accuracy on **FPT**. It indicates that the distribution resulting from Speck32/64 is similar to the distribution resulting from Chaskey. These two symmetric ciphers are both built on modulo addition which may be the cause of this phenomenon. This guess can be further verified in the following experiments.

4.3 Experiments on Present64/80

Neural distinguishers for reduced Present64/80: Present [7] is a block cipher that is based on a 4×4 Sbox. Based on the plaintext difference $\alpha = (0, 0, 0, 0x9)$ provided in [24], we have built baseline distinguishers for Present64/80 reduced up to 7 rounds.

Our neural distinguishers are also built for Present64/80 reduced to 6, 7 rounds respectively. The penalty factor is 10^{-4} and other related parameters are the same with Table 2. The distinguishing accuracy comparisons are presented in Table 7.

Table 7. Distinguishing accuracy of neural distinguishers for 6/7 rounds Present64/80

R	BD	Our neural distinguishers			
		$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
6	0.6584	0.7198	0.7953	0.8308	0.8259
7	0.5486	0.5503	0.5853	0.5786	0.5818

Compared with baseline distinguishers, our neural distinguishers can obtain distinguishing accuracy promotion under all settings. Additionally, the distribution resulted from the Present64/80 reduced to 6 rounds is very different from the uniform distribution, which can be inferred from the distinguishing accuracy. Then we can obtain high accuracy promotion by directly increasing the value of k as expected.

The nonlinear component of Present is Sbox, which is different from the modulo addition. Our neural distinguishers can still achieve the target of accuracy promotion as long as the distribution resulted from the target cipher is different from the uniform distribution. This further proves that our distinguisher model is applicable to symmetric ciphers based on Sbox which the size of the input is the same with the size of the output.

False negative/positive test: Based on baseline distinguishers, $10^4 \times k$ false positive and false negative samples are randomly generated. Table 8 shows the pass ratios of **FNT** and **FPT**.

Table 8. Pass ratios of false negative test and false positive test

R	False Negative Test				False Positive Test			
	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
6	0.0277	0.0097	0.0258	0.0751	0.0147	0.0046	0.0068	0.0183
7	0.1796	0.0587	0.1214	0.1488	0.0533	0.0126	0.0324	0.0302

The false positive and false negative test confirm the same conclusion, but also allow us to know more about our neural distinguishers. Our neural distinguishers have captured derived features from k ciphertext pairs. However, there is a different phenomenon that doesn't occur in the same tests for reduced Speck32/64 and Chaskey. Compared with **FPT**, our neural distinguishers can obtain higher accuracy on **FNT**. Besides, if we look at Table 6 and Table 8 at the same time, we can find another unusual phenomenon. Compared with the distribution generated by the T-round cipher, the distribution generated by the $(T+1)$-round cipher is more similar to the uniform distribution. However, our neural distinguishers can obtain higher accuracy on two tests for the $T+1$-round cipher. At last, our neural distinguishers with lower accuracy can obtain higher accuracy in the false negative test, such as our neural distinguishers with $k = 2$.

These phenomenons are unusual but totally obey the properties of neural networks. Since there are no limitations about the neural network except for the optimization loss function, The neural network can autonomously choose which features to learn, and can also autonomously assign weights to the learned features. If the distribution resulted from the target cipher is too complex, the performance of our neural distinguishers may vary differently.

At the same time, these phenomenons also prove that our distinguisher model is very reasonable. No matter how complex the target distribution is, our neural distinguishers can always capture useful derived features from k ciphertext pairs, which can promote the distinguishing accuracy.

4.4 Experiments on DES

Neural distinguishers for reduced DES: DES [14] is a block cipher that is built on a 6×4 Sbox. Based on the analysis of DES in [6], the plaintext difference adopted in this paper is $\alpha = (0x40080000, 0x04000000)$. We have built baseline distinguishers for DES reduced up to 6 rounds.

Our neural distinguishers are also built for DES reduced to 5, 6 rounds. The batch size is adjusted to 5000. The penalty factor is increased to 8×10^{-4}. Other related parameters are the same as Table 2. Corresponding distinguishing accuracy comparisons are presented on Table 9.

Table 9. Distinguishing accuracy of neural distinguishers for 5/6 rounds DES

R	BD	Our neural distinguishers			
		$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
5	0.6261	0.7209	0.8382	0.9318	0.9585
6	0.5493	0.5653	0.5568	0.5507	0.5532

Compared with baseline distinguishers, our neural distinguishers can obtain distinguishing accuracy promotion under all settings. When $R = 5$, the baseline distinguisher can well classify two kinds of ciphertext samples. Thus there are significant differences between the target distribution and the uniform distribution, which can make our neural distinguisher capture more derived features by increasing k. When $R = 6$, the two distributions are very similar. Then it is unlikely to keep getting more useful information by increasing k. On the contrary, it

may make two ciphertext groups sampled from two different distributions more similar. Corresponding influence is that the accuracy of our distinguisher may decrease.

False negative/positive test: Based on baseline distinguishers, $10^4 \times k$ false negative and false positive samples are randomly generated. Table 10 shows the pass ratios of **FNT** and **FPT**.

Table 10. Pass ratios of false negative test and false positive test

R	False Negative Test				False Positive Test			
	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
5	0.0046	0.0034	0.0132	0.0131	0.0594	0.0627	0.0566	0.0518
6	0.0802	0.2348	0.2526	0.3207	0.0462	0.0598	0.0921	0.0809

The **FNT** and **FPT** prove that the accuracy promotion of our neural distinguishers mainly comes from derived features. When $R = 6$, our neural distinguishers can also obtain higher accuracy on **FNT** than that on **FPT**. Although this rule doesn't hold when $R = 5$, it doesn't affect our conclusions. As long as the distribution generated by the reduced cipher is not a uniform distribution, our neural distinguishers can obtain accuracy promotion by capturing derived features from k samples. Besides, the distributions generated by symmetric ciphers which have the same nonlinear component are likely to be similar.

4.5 Experiments on SHA3-256

Neural distinguishers for reduced SHA3-256: SHA3-256 [15] is a hash function that is exactly different from the other four ciphers from the perspective of the nonlinear component. When one message block is fed into reduced SHA3-256, we collect the first 32 bytes of the output process after T-rounds permutation is applied to this message block. Given a message differential $\alpha = 1$, we have built baseline distinguishers for SHA3-256 reduced up to 4 rounds.

Our neural distinguishers are also built for SHA3-256 reduced to $3, 4$ rounds. Limited by the computer memory, the number of ciphertext pairs is adjusted to $N = 2 \times 10^6$. The batch size is 500, and the penalty factor is 10^{-5}. The distinguishing accuracy comparisons are presented in Table 11.

Table 11. Distinguishing accuracy of neural distinguishers for 3/4 rounds SHA3-256

R	Gohr	Our neural distinguishers			
		$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
3	0.7228	0.8149	0.9241	0.971	0.9904
4	0.5844	0.6409	0.8441	0.8748	0.8775

Compared with baseline distinguishers, our neural distinguishers can obtain accuracy promotion under all settings. SHA3-256 is built on nonlinear logic operations. As long as the resulting distribution is different from the uniform distribution, our neural distinguishers can obtain accuracy promotion as expected.

False negative/positive test: With the help of baseline distinguishers, $10^4 \times k$ false negative and false positive samples are randomly generated. Table 12 shows the pass ratios of **FNT** and **FPT**.

Table 12. Pass ratios of false negative test and false positive test

R	False Negative Test				False Positive Test			
	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$	$ND_{k=2}$	$ND_{k=4}$	$ND_{k=8}$	$ND_{k=16}$
3	0.2249	0.2347	0.3336	0.2711	0.1045	0.0961	0.0171	0.0088
4	0.8834	0.1425	0.1406	0.8646	0.4619	0.4933	0.4918	0.5044

Test results above can also prove that our neural distinguishers have successfully captured derived features from k ciphertext pairs. These derived features are very useful for promoting distinguishing accuracy as Table 11 shows. Since the nonlinear component is different from the other four ciphers, the test results are also very different. These complex but interesting test results above can still prove that our neural distinguishers can learn derived features no matter how complex the target distribution is.

4.6 Short Conclusion

Our neural distinguishers for five reduced symmetric ciphers can always obtain distinguishing accuracy promotion by regarding k ciphertext pairs as the analysis object. The **FNT** and **FPT** also prove that the accuracy promotion comes from derived features that have been successfully learned by our neural distinguishers. Since there are similar properties in the distributions generated by symmetric ciphers built on the same nonlinear component, experiments on these five different symmetric ciphers can prove the generic superiority of our distinguisher model.

5 Data Reuse Strategy for Reducing Data Complexity

Previous experiments have shown the superiority of our distinguisher model. But there is a potential problem that may make it unpractical being applied to attacks in real scenarios. Assuming the baseline distinguisher and our neural distinguisher have the same performance, and a certain attack requires M random inputs. If we directly reshape $M \times k$ ciphertext pairs into M ciphertext groups, the data complexity of our neural distinguisher is k times as much as the data complexity of the baseline model.

Given N ciphertext pairs $X_i = (C_{i,0}, C_{i,1}), i \in [1, N]$, there are a total of C_N^k options for composing a ciphertext group, which is much larger than $\frac{N}{k}$. Thus we can randomly select M ciphertext groups from C_N^k options. Such a strategy can help reduce the data complexity. In fact, it is equivalent to attach more importance to derived features from k ciphertexts. However, the subsequent key recovery attacks using this naive strategy don't obtain good results. The main reason is that the sampling randomness of M ciphertext groups is greatly destroyed. Two new concepts are proposed for overcoming this problem.

Maximum Reuse Frequency: During the generation of M ciphertext groups, a ciphertext pair is likely to be reused several times. Let's denote the reuse frequency of the i_{th} ciphertext pair as $RF_i, i \in [1, N]$. Maximum Reuse Frequency (MRF) is defined as the maximum value of RF_i:

$$MRF = \max RF_i, i \in [1, N] \tag{12}$$

Sample Similarity Degree: For any two ciphertext groups G_i, G_j, the similarity of these two ciphertext groups is defined as the number of the same ciphertext pairs. As for M ciphertext groups, Sample Similarity Degree (SSD) is defined as the maximum of any two ciphertext groups' similarity:

$$\begin{gathered} SSD = \max \left| G_i \bigcap G_j \right|, i, j \in [1, M] \\ G_i = \{X_{i1}, \cdots, X_{ik}\} \\ G_j = \{X_{j1}, \cdots, X_{jk}\} \\ i1, \cdots, ik, j1, \cdots, jk \in [1, N] \end{gathered} \tag{13}$$

MRF can ensure that the contribution of each ciphertext pair is similar. SSD can increase the distribution uniformity of M ciphertext groups as much as possible. Based on the above two concepts, we propose the following ***Data Reuse Strategy*** that can reduce data complexity and maintain sampling randomness:

1. Set two upper thresholds for MRF and SSD.
2. Randomly select k ciphertext pairs from N ciphertext pairs to form a ciphertext group.
3. Repeat step 2 for M times to obtain M ciphertext groups.
4. Compute MRF and SSD. If two values are both smaller than the threshold we set, return the M ciphertext groups. Or start from step 2 again.

6 Key Recovery Attacks on Speck32/64

Our neural distinguishers can obtain much higher distinguishing accuracy than baseline distinguishers. When applied to the same key recovery attack, the performance of different neural distinguishers can be compared in two aspects.

When we decrypt enough ciphertexts with a key and feed partially decrypted ciphertexts into a neural distinguisher, the value of the output signal is related to the key. When the output signal of the correct key is very different from the output signals of the wrong keys, it indicates the neural distinguisher can well distinguish the correct key from wrong keys. For deep learning based distinguishers, we are the first to find a widespread output signal clustering phenomenon. Based on this phenomenon, we can directly assess the performance of neural distinguishers by observing the output signal distribution.

We can also compare the performance of different neural distinguishers indirectly through a complete key recovery attack. Such an evaluation method will be greatly affected by the key recovery attack strategy. But the final comparison result can be used as a reference.

In this section, we will introduce the output signal clustering phenomenon firstly. The direct comparison between baseline distinguishers and our neural distinguishers is also given. Then the same key recovery attack [12] on 11-round Speck32/64 is performed again with our neural distinguishers. At last, a new key recovery attack scheme is proposed based on two important rules derived from the output signal clustering phenomenon. It can further reduce the time complexity.

6.1 Output signal clustering phenomenon

There is a common property for most symmetric ciphers.

Property 1. Let a ciphertext be decrypted one round with two different round keys, $C^r_{T-1} = Dec(C_T, SK_1)$, $C^w_{T-1} = Dec(C_T, SK_2)$. *If* SK_1 *and* SK_2 *are different at a few bits (e.g. just 1 bit or 2 bits), the Hamming distance between* C^r_{T-1} *and* C^w_{T-1} *will be very small.*

At the same time, there is also a common property for a neural network.

Property 2. Given a trained neural network $f(\cdot)$ *for solving a binary classification problem, if two input samples* X_1, X_2 *are very close to each other in the input space, two outputs* $f(X_1)$ *and* $f(X_2)$ *obtained from the neural network may satisfy* $f(X_1) \approx f(X_2)$ *with a high probability.*

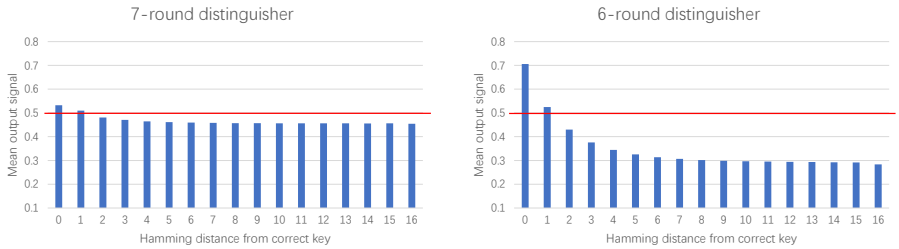

Fig. 3. Mean output signals of baseline distinguishers for 6/7-round Speck32/64.

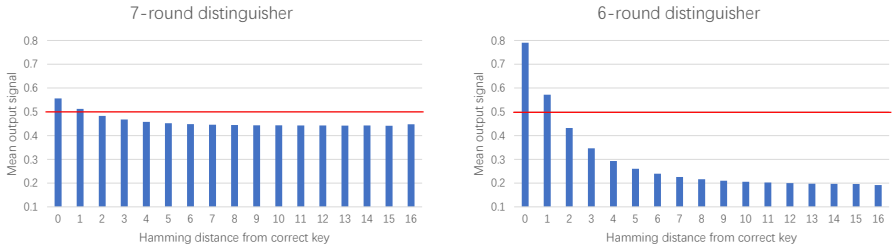

Fig. 4. Mean output signals of our neural distinguishers for 6/7-round Speck32/64.

Although the real distance metric for a neural network is unknown, the Hamming distance is a good approximation. Thus there exists an output signal clustering phenomenon for neural distinguishers. When two keys have the same Hamming distance from the correct key, corresponding output signals should be similar.

Given a key difference δ and a cipher reduced to $R+1$ rounds, we randomly generate N 5-tuple $(P_{i,1}, P_{i,2}, C_{i,1}, C_{i,2}, SK_{i,R+1}), P_{i,1} \oplus P_{i,2} = \alpha, i \in [1, N]$. $C_{i,1}$, $C_{i,2}$ are encrypted from $P_{i,1}, P_{i,2}$ with the same key, and $SK_{i,R+1}$ is the last round key. Then $C_{i,1}$ and $C_{i,2}$ are decrypted one round with $\delta \oplus SK_{i,R+1}$. At last, these decrypted ciphertext pairs are reshaped into $\frac{N}{k}$ ciphertext groups and fed into an R-round neural distinguisher. Denote the neural distinguisher's output for $(C_{i,1}, C_{i,2})$ as $Z_{\delta,i}$. Then we can obtain the mean value of $Z_{\delta,i}, i \in [1, N]$ and denote it as Z_δ. It can be regarded as the expected output signal when there is a difference δ between the key guess and the true key.

After we obtain all the $Z_\delta, \delta \in [0, 2^{16}-1]$, we further calculate the Hamming weight HD_δ of each δ. For Speck32/64, HD_δ ranges from 0 to 16. At last, the following mean value is formulated:

$$Z_{HD_\delta} = \sum_{HD_\delta = i} \frac{Z_\delta}{C_{16}^i}, i \in [0, 16] \tag{14}$$

where Z_{HD_δ} stands for the expectation output signal when the Hamming distance between the key guess and true key is HD_δ. In this paper we denote Z_{HD_δ} as the mean output signal.

Let $N = 2000$, Figure 3 shows the mean output signals of neural distinguishers [12] reduced to 6/7 rounds. Figure 4 shows the mean output signals of our neural distinguishers reduced to 6/7 rounds. The group size of our neural distinguishers is $k = 2$. Compared with the baseline model, our neural distinguishers can increase the mean output signal of the correct key. As for wrong keys, when the Hamming distance from the correct key is larger than 2, our neural distinguishers can decrease the mean output signal. Then the mean output signal difference can be further increased by our neural distinguishers. Such a good property proves that the superiority of our neural distinguishers. It can directly show the positive impact of derived features on key recovery attacks.

Besides, there are two useful properties that can help us develop a new key recovery attack scheme.

Property 3. The right key has the highest mean output signal. Thus the right key can obtain the highest key rank score as long as the number of input samples is large enough in the key recovery attack.

Property 4. The mean output signal decreases as the Hamming distance from the right key increases. A high key rank score will only occur when the Hamming distance between the key guess and true key is very small. Taking the true key as the center, the keys which can obtain a high score is involved in a small cluster around the center.

6.2 Key Recovery Attack Using Gohr's Attack Scheme

In this section, we will further assess our neural distinguishers through the same key recovery attack [12] on 11-round Speck32/64.

Gohr's attack scheme. The 7-round distinguisher is extended to a 9-round distinguisher by prepending a 2-round differential transition $(0x211, 0xa04) \rightarrow (0x0040, 0)$ with a probability of about $\frac{1}{64}$. The 9-round distinguisher is then extended by another round at no additional cost by asking for encryptions of ciphertext pairs (P_1, P_2) that encrypt to the desired input difference α after one round of Speck encryption. This is easy since no key addition happens in Speck before the first nonlinear operation.

This 10-round distinguisher is applied to a key recovery attack on 11-round Speck32/64. Six neutral bits $\{20, 21, 22, 14, 15, 23\}$ are used to creat from a plaintext pair a plaintext structure consisting of 64 plaintext pairs. Such a plaintext

structure is expected to pass the prepended 2-round differential transition. For each plaintext structure, decrypt the resulting ciphertexts under all final subkeys and rank each partially decrypted ciphertext structure using the 7-round distinguisher. If the resulting score is beyond a threshold c_{11}, decrypt another round and grade the resulting partially-decrypted ciphertexts using the 6-round neural distinguisher. A key guess is returned if the resulting score for the partially decrypted ciphertext structure then exceeds another threshold c_{10}.

In the attack scheme above, the key rank score is formulated as

$$S_{SK} = \sum_{i=1}^{\frac{N}{k}} \log_2 \left(\frac{Z_i}{1 - Z_i} \right) \tag{15}$$

where S_{SK} stands for the key rank score of SK, and Z_i is the i_{th} input sample's output signal given by the neural distingusiher. As for the baseline model, the group size is $k = 1$.

By transforming this attack scheme into a multi-armed bandit problem, an accelerated version that can reduce the time complexity of the 11-round attack to 38 bits is given in [12]. This accelerated version is based on the standard Upper Confidence Bound (UCB) algorithm [2]. A key recovery attack is considered as a success only when the returned final subkey is correct and the second-to-last subkey is incorrect in at most 2 bits. If the maximum number of plaintext structures is 100, the key recovery attack using the baseline model can obtain a success rate of 52%. The average data complexity is $2^{13.5}$.

Key recovery attack using our neural distinguisher. When the maximum number of randomly generated plaintext structure is given, the success rate of the key recovery attack is strongly related to the performance of the neural distinguisher.

Our neural distinguisher can be applied to Gohr's attack scheme directly. In order to evaluate our neural distinguisher fairly, the data complexity is unchanged. The maximum number of plaintext structures is 100. For our neural distinguisher, the group size is $k = 2$. A data reuse strategy with $MRF = 2$ and $SSD = 1$ is applied. Two thresholds are $c_{11} = 18$, $c_{10} = 150$.

With the help of the accelerated attack scheme proposed by Gohr, the key recovery attack using our neural distinguisher is performed 1000 times. Based on these given settings, our attack can succeed in 544 out of 1000 trials The average data complexity of our attack is $2^{13.2}$.

As we have proved in the previous section, our neural distinguishers can increase the mean output signal of the correct key and reduce the mean output signal of the wrong keys. When the data complexity and time complexity are limited, the output signal difference will play a vital role in the key recovery attack. This is why the same key recovery attack using our neural distinguishers can obtain a little higher success rate with the same data complexity. If the output signal difference is very high, it's possible to perform this key recovery attack with a 7-round neural distinguisher. Thus it's very important to develop more powerful neural distinguishers.

6.3 Key Recovery Attack Using New Attack Scheme

In this section, a new key recovery attack scheme is proposed based on the output signal clustering phenomenon. The key recovery attack on 11-round Speck32/64 is performed again based on this new scheme. Compared with Gohr's attack scheme, it can further reduce the time complexity. Besides, experiments in this section can also show the superiority of our neural distinguishers.

New key recovery attack scheme. The most important property of Gohr's attack scheme is that the correct ciphertext structure which passes the prepended differential can be found after the 2-round attack. Based on the correct ciphertext structure, the unknown round keys can be recovered at a negligible cost.

According to *Property 3*, the correct round key can obtain the highest key rank score as long as the correct ciphertext structure contains enough ciphertext pairs. Thus the correct ciphertext structure can be found with a 1-Round attack. Based on this conclusion, we propose a new key recovery attack scheme as follows:

1. Set a series of thresholds $c_1, \cdots, c_T$ for a cipher reduced to T rounds.
2. Generate enough plaintext structures, and obtain the ciphertext structures through $T+1$ rounds of encryption.
3. For each ciphertext structure, we decrypt it under all possible subkeys of the last round and obtain the key rank scores.
4. If any final round key's key rank score is higher than c_T, we think the related ciphertext structure is a correct ciphertext structure. Then we save the partially decrypted ciphertext structure and try to recover other round keys as follows:
   (a) Decrypt the partially decrypted ciphertext structure under all round keys of the i_{th} round, $i \in [1, T-1]$.
   (b) If the key rank score of a key is higher than c_i, repeat the process and attack the $(i-1)_{th}$ round.
5. If a ciphertext structure can pass the attack for each round, return recovered keys as the final key guesses.

It's worth noticing that all thresholds $c_1, \cdots, c_T$ should be large enough. This can ensure only the correct ciphertext structure can pass the attack for the T_{th} round. Besides, only a few keys will survive in each round. This is based on the second finding in the output signal clustering phenomenon. In fact, all the surviving keys belong to the cluster centered on the right key.

Like Gohr's attack scheme, the most important point of the new attack scheme is also finding the correct ciphertext structure. Based on this new key recovery attack scheme, we have performed tests on 11-round Speck32/64. The attack target is to find the correct ciphertext structure with only one round attack. Besides, the key which has the highest key rank score is returned as the key guess of the T_{th} round.

Seven neutral bits $\{20, 21, 22, 14, 15, 23, 7\}$ are adopted for generating plaintext structures. c_{11} is 15 for the 7-round distinguisher [12]. As for our 7-round neural distinguisher, the group size is 2, $c_{11} = 35$. Parameters of the data reuse strategy are $MRF = 2$ and $SSD = 1$. 100 experiments are then performed with

two neural distinguishers above respectively. The maximum number of plaintext structures is set to 100. There are about 49 trials in which a correct plaintext structure will arise. Both our neural distinguisher and the baseline model can find the right plaintext structure in the 49 experiments.

Besides, the Hamming distance between the final key guess and the true round key is lower than 2. By regarding the number of keys which have equal or higher key rank score than the true key as the key rank of the true key, we observe that the mean key rank is both lower than 4 no matter which neural distinguisher is used. When we attack the same cipher reduced to shorter rounds, the mean rank of the true key will be higher. Thus we can also recover unknown round keys at a negligible cost once the correct ciphertext structure is found.

Accelerate the new key recovery attack scheme. It's time-consuming to find the correct ciphertext structure by decrypting each ciphertext structure with all final round keys. In fact, It can be accelerated in two aspects.

First, the search for the true key in each round can be broken down into two simpler processes as follows:

1. Find the correct cluster which contains the true round key. The size of the cluster is defined by the Hamming distance. According to Figure 3, Figure 4, the size is set to 2. The Hamming distance from each element in the cluster to the center does not exceed 2.
2. Search the true key in the cluster.

Since the output signal decreases as the Hamming distance from the right key increases, we can set two thresholds $c_{i,1}, c_{i,2}, c_{i,1} < c_{i,2}$. Most keys in the cluster centered in the true round key should have a key rank score higher than $c_{i,1}$. But only the true key can obtain a key rank score higher than $c_{i,2}$ with a high probability. Thus we can randomly generate some seed keys and obtain corresponding key rank scores. If a seed key's key rank score is higher than $c_{i,1}$, we will test keys in the cluster centered on this seed key. If any key can obtain a higher key rank score, take it as the new center and continue the search until the center is unchanged. If the final center key's key rank score is higher than $c_{i,2}$, return it as the final key guess of the i_{th} round.

Second, we can focus on the most promising ciphertext structure instead of spending the same amount of computation on every ciphertext structure. This can be achieved by the standard Upper Confidence Bound (UCB) algorithm [2]. Set a maximum iteration number. Then we actively choose a ciphertext structure for the test at each iteration. This decision is made based on a concept named the priority score. The priority score of each ciphertext structure is

$$S_i = w^i_{\max} + \alpha \times \sqrt{\frac{\log_2(j)}{n_i}} \tag{16}$$

where $w^i_{\max}$ is the highest score obtained from the current neural distinguisher so far for the i_{th} ciphertext structure, j is the number of current iteration, n_i is the number of iterations which the i_{th} ciphertext structure has been selected, and α is the balance factor. Before each iteration, all priority scores are updated and the ciphertext structure that has the highest score is selected.

Key recovery attack using the new accelerated attack scheme. Key recovery attack on 11-round Speck32/64 is performed with this accelerated attack scheme again. 7 probabilistic neutral bits $\{20, 21, 22, 14, 15, 23, 7\}$ are used to generate 100 plaintext structures randomly. The maximum iteration number is 500. At each iteration, we will test 150 possible keys. Key recovery is considered a success only when the hamming distance between the returned subkey and the correct key is no more than 2. Except for the total success rate, two indicators will also be observed. The first one is the number of experiments that the correct ciphertext structure really exists. The second one is the success rate of these experiments above.

Key recovery attack is firstly performed 1000 times with the distinguisher in [12]. Except for settings above, $c_{11,1} = 5$, $c_{11,2} = 15$. 572 experiments out of 1000 are successful attacks. The Hamming distance between the returned key and the true key is no more than 2. The corresponding key rank score also exceeds the threshold c_2. There are 551 experiments in which the correct ciphertext structure exists. Among these experiments above, only 10 experiments are failed. In the remaining 31 successful experiments, the number of ciphertext pairs that pass the prepended differential transition in the returned ciphertext structure is also greater than 100. The mean data complexity of our attack is $2^{14.12}$.

Based on the data reuse strategy with $MRF = 2$ and $SSD = 1$, key recovery attack is performed for 1000 times with our neural distinguisher which the group size is 2. Except for settings above, $c_{11,1} = 25$, $c_{11,2} = 35$. 595 experiments out of 1000 are successful attacks. There are 564 experiments in which the correct ciphertext structure exists. Among these experiments above, only 15 experiments are failed. In the remaining 46 successful experiments, the number of ciphertext pairs that pass the prepended differential transition in the returned ciphertext structure is also greater than 64. The mean data complexity of our attack is $2^{14.11}$.

According to these results, we can obtain the following conclusions. First, this new key recovery attack scheme is practical and reasonable. Second, the key recovery attack can also be improved by using our 7-round neural distinguisher. Although the difference between the mean output signal of the correct key and the output signal of the wrong key is not increased significantly, it still has a non-negligible effect on the final success rate.

Time complexity. The time consumption of Gohr's key recovery scheme and the new key recovery scheme are both related to the key search strategy. The total time complexity is proportional to the number of keys that need to be searched. When a powerful graphics card is available, the time required for the attack depends almost entirely on the number of keys to be searched.

We have estimated the time consumption of two key recovery schemes under the same experiment environment. A single GTX 1080 Ti graphics card is used. When the key recovery attack on 11-round Speck32/64 is performed using Gohr's attack scheme, a key guess is returned in about 50 seconds. When the same attack is performed using our new attack scheme, a key guess is returned in about 12 seconds. Besides, the 7-round neural distinguisher adopted here is also the

baseline model. The time complexity of Gohr's key recovery attack is equivalent to 2^{38} Speck32/64 encryption. Thus the time complexity of the key recovery attack using our attack scheme is about 2^{36} Speck32/64 encryption.

7 Conclusion

As long as a distribution is not a uniform distribution, there will be many derived features generated from the non-uniformity. These derived features can be captured from multiple samples. Based on such a finding, we propose a new neural distinguisher model which takes k ciphertext pairs as input.

This new neural distinguisher model can successfully capture derived features, which can always promote the distinguishing accuracy under various settings of k. Besides, it can be used to improve the key recovery attack. At the same time, it is as generic as Ghor's distinguisher model. All the works in this paper have shown the superiority of our new neural distinguisher model.

Based on our new neural distinguisher model and some happened interesting phenomenon, there are several important research directions that are worthy of exploring. First, applying this new neural distinguisher to key recovery attacks on other ciphers still needs to perform. Second, if we can add some suitable prior knowledge to the neural distinguisher, its performance is expected to be further enhanced. Now there are no extra requirements for k ciphertext pairs composing a ciphertext group. Third, derived features extracted from the non-uniformity of the distribution generated by a certain cipher may be applicable to other ciphers constructed based on the same nonlinear components. In the **FNT** and **FPT**, there are similar results for symmetric ciphers that are built based on the same nonlinear components. Thus, it's likely to reduce the cost of building neural distinguishers for different ciphers. Noval cryptanalysis ways are also likely to be developed for symmetric ciphers based on the same nonlinear component.

Acknowledgement

The authors are grateful to the anonymous reviewers for their insightful comments that can help improve the quality of the paper. This work is supported by the National Key Research and Development Program of China (2018YFB0803405, 2017YFA0303903).

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential cryptanalysis of round-reduced simon and speck. fast software encryption pp. 525–545 (2014)
2. Auer, P.: Using upper confidence bounds for online learning. foundations of computer science pp. 270–279 (2000)
3. Beaulieu, R., Shors, D., Smith, J., Treatmanclark, S., Weeks, B., Wingers, L.: The simon and speck lightweight block ciphers. design automation conference p. 175 (2015)

4. Bengio, Y., Ducharme, R., Vincent, P.: A neural probabilistic language model. Neural Information Processing Systems (NeurIPS) pp. 932–938 (2000)
5. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. Lecture Notes in Computer Science, vol. 537, pp. 2–21. Springer (1990). https://doi.org/10.1007/3-540-38424-3_1, https://doi.org/10.1007/3-540-38424-3_1
6. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer (1993). https://doi.org/10.1007/978-1-4613-9314-6, https://doi.org/10.1007/978-1-4613-9314-6
7. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: Present: An ultra-lightweight block cipher. cryptographic hardware and embedded systems pp. 450–466 (2007)
8. Chen, Y., Yu, L., Ota, K., Dong, M.: Robust activity recognition for aging society. IEEE Journal of Biomedical and Health Informatics **22**(6), 1754–1764 (2018)
9. Chen, Y., Yu, L., Ota, K., Dong, M.: Hierarchical posture representation for robust action recognition. IEEE Transactions on Computational Social Systems **6**(5), 1115–1125 (2019)
10. Choi, E., Xiao, C., Stewart, W.F., Sun, J.: Mime: Multilevel medical embedding of electronic health records for predictive healthcare. Neural Information Processing Systems (NeurIPS) pp. 4547–4557 (2018)
11. Francois, C.e.a.: Keras (2015), https://keras.io
12. Gohr, A.: Improving attacks on round-reduced speck32/64 using deep learning. international cryptology conference pp. 150–179 (2019)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. computer vision and pattern recognition pp. 770–778 (2016)
14. Howard, R.: Data encryption standard. Information Age archive **9**(4), 204–210 (1987)
15. Huang, S., Wang, X., Xu, G., Wang, M., Zhao, J.: Conditional cube attack on reduced-round keccak sponge function. theory and application of cryptographic techniques pp. 259–288 (2017)
16. Itai, D.: Improved differential cryptanalysis of round-reduced speck. International Workshop on Selected Areas in Cryptography pp. 147–164 (2014)
17. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Feifei, L.: Large-scale video classification with convolutional neural networks. computer vision and pattern recognition pp. 1725–1732 (2014)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2015), http://arxiv.org/abs/1412.6980
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems (NeurIPS) pp. 1097–1105 (2012)
20. Lai, X., Massey, J.L., Murphy, S.: Markov ciphers and differential cryptanalysis. Lecture Notes in Computer Science, vol. 547, pp. 17–38. Springer (1991). https://doi.org/10.1007/3-540-46416-6_2, https://doi.org/10.1007/3-540-46416-6_2
21. Lee, J., Heo, M., Kim, K., Kim, C.: Single-image depth estimation based on fourier domain analysis. computer vision and pattern recognition pp. 330–339 (2018)
22. Matsui, M.: Linear cryptanalysis method for DES cipher. Lecture Notes in Computer Science, vol. 765, pp. 386–397. Springer (1993). https://doi.org/10.1007/3-540-48285-7_33, https://doi.org/10.1007/3-540-48285-7_33
23. Matthies, L., Szeliski, R., Kanade, T.: Incremental estimation of dense depth maps from image sequences. computer vision and pattern recognition pp. 366–374 (1988)

24. Meiqin, W.: Differential cryptanalysis of present. IACR eprint (2009)
25. Mikolov, T., Karafiat, M., Burget, L., Cernocký, J., Khudanpur, S.: Recurrent neural network based language model. conference of the international speech communication association pp. 1045–1048 (2010)
26. Moriai, S., Sugita, M., Aoki, K., Kanda, M.: Security of E2 against truncated differential cryptanalysis. Lecture Notes in Computer Science, vol. 1758, pp. 106–117. Springer (1999). https://doi.org/10.1007/3-540-46513-8_8, https://doi.org/10.1007/3-540-46513-8_8
27. Mouha, N., Mennink, B., Van Herrewege, A., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An efficient mac algorithm for 32-bit microcontrollers. selected areas in cryptography pp. 306–323 (2014)
28. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local svm approach. international conference on pattern recognition **3**, 32–36 (2004)
29. Shirai, T., Shibutani, K.: Improving immunity of feistel ciphers against differential cryptanalysis by using multiple MDS matrices. Lecture Notes in Computer Science, vol. 3017, pp. 260–278. Springer (2004). https://doi.org/10.1007/978-3-540-25937-4_17, https://doi.org/10.1007/978-3-540-25937-4_17
30. Tosi, F., Aleotti, F., Poggi, M., Mattoccia, S.: Learning monocular depth estimation infusing traditional stereo knowledge. computer vision and pattern recognition pp. 9799–9809 (2019)
31. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Mining actionlet ensemble for action recognition with depth cameras. computer vision and pattern recognition pp. 1290–1297 (2012)