Improved Neural Distinguishers with Multi-Round and Multi-Splicing Construction

JiaShuo Liu, JiongJiong Ren, ShaoZhen Chen and ManMan Li

Information Engineering University, ZhengZhou, P.R.China, jiongjiong_fun@163.com

Abstract. In CRYPTO 2019, Gohr successfully applied deep learning to differential cryptanalysis against the NSA block cipher Speck32/64, achieving higher accuracy than traditional differential distinguishers. Until now, the improvement of neural differential distinguishers is a mainstream research direction in neural aided cryptanalysis. But the current development of training data formats for neural distinguishers forms barriers: (1) The source of data features is limited to linear combinations of ciphertexts, which does not provide more learnable features to the training samples for improving the neural distinguishers. (2) Lacking breakthroughs in constructing data format for network training from deep learning perspective. In this paper, considering both the domain knowledge about deep learning and information of differential cryptanalysis, we use the output features of the penultimate round to propose a two-dimensional and non-realistic input data generation method of neural differential distinguishers. Then, we validate that the proposed new input data format has excellent features through experiments and theoretical analysis. Moreover, combining the idea of multiple ciphertext pairs, we generate two specific models for data input construction: MRMSP(Multiple Rounds Multiple Splicing Pairs) and MRMSD(Multiple Rounds Multiple Splicing Differences), and then build new neural distinguishers against Speck and Simon family, which effectively improve the performance compared with the previous works. To the best of our knowledge, our neural distinguishers achieve the longest rounds and the higher accuracy for NSA block ciphers Speck and Simon.

Keywords: Deep Learning · Block Cipher · Neural Distinguisher · Speck · Simon

1 Introduction

Differential cryptanalysis [1] is one of the most powerful analysis techniques in cryptography. This statistical cryptanalysis exploits how a specific input difference propagates through the cipher into the output difference. The most important step of differential cryptanalysis is to find differential characteristics with high probability. This technique has been widely applied to block ciphers and hash functions, and many new constructions of these primitives are specifically designed to withstand this attack. In order to evaluate the security of block ciphers against differential cryptanalysis, Lai *et al.* [2] introduced Markov ciphers in 1991. Following the Markov cipher assumption, the probability of a differential characteristic can be computed by multiplying the probability of differential propagation of each round. Then, the Markov cipher assumption is used in differential attacks on block ciphers practically.

Deep learning has made remarkable improvement in many fields: machine translation [3, 4], imageNet classification [5] and so on. It is worth mentioning that some researchers have investigated the viability of applying deep learning to cryptanalysis. In CRYPTO 2019, Gohr [6] first presented differential-neural cryptanalysis combined deep learning with differential analysis. He trained neural distinguishers of Speck32/64 based on the deep residual neural networks (ResNet). The labeled data used as training data is composed of ciphertext pairs: half of the training data comes from ciphertext pairs encrypted by plaintext pairs with a fixed input difference, half from random values. Gohr obtained a high accuracy for 6-round and 7-round neural distinguishers of Speck32/64 and achieved 11-round and 12-round key recovery attacks based on the neural distinguishers. The advantage of the high accuracy of neural distinguishers is reflected in distinguish attack and the key recovery phase. Improving the accuracy of neural distinguishers is helpful to distinguish block ciphers from pseudo-random permutations. Meanwhile, the complexity of key recovery decreases as the accuracy of neural distinguishers increases. And if the accuracy rate is improved, the key recovery attack can be implemented on longer rounds.

To improve the accuracy of neural distinguishers, the researchers have explored two mainstream directions. One is adopting different neural networks. Bao *et al.* [7] used Dense Network and Squeeze-and-Excitation Network with deep architectures to train neural distinguishers, and obtained effective (7-11)-rounds neural distinguishers for Simon32/64. Zhang *et al.* [8] adopted the idea of the inception block of GoogLeNet to construct the new neural network architecture to train neural distinguishers for (5-8)-rounds Speck32/64 and (7-12)-rounds Simon32/64 and achieved significant accuracy raise.

The other popular research direction is changing the input data format of neural distinguishers. Chen *et al.* [9] proposed multiple groups of ciphertext pairs instead of single ciphertext pair [6] as the training sample of the neural network and effectively improved the accuracy of the (5-7)-rounds neural distinguishers of Speck32/64. Hou *et al.* [10] built multiple groups of output differences pairs instead of multiple groups of ciphertext pairs [9] to further improve the accuracy of neural distinguishers. Lu *et al.* [11] proposed groups in combination with ciphertext pairs and output differences pairs as the training sample and significantly improved the accuracy of the neural distinguishers of the Simon and Simeck family. We view the existing input data formats of neural distinguishers as the same type since the sources of features provided for the neural network directly come from ciphertext pairs.

To achieve a breakthrough in input data format and improve neural distinguishers, we draw inspiration from cryptanalysis and deep learning.

In cryptanalysis, if an iterated cipher is a Markov cipher and its round subkeys are independent and uniformly random, the sequence of differences at each round output forms a homogenous Markov chain. The characteristics of the i-round difference feature are determined by the statistical properties of the (i-1)-round difference, which enlightens us to concentrate on the penultimate round differential information. Besides, in EUROCRYPT 2021, Benamira *et al.* [12] gave a detailed explanation of Gohr's neural distinguishers. Their explanation showed that Gohr's neural distinguishers made their decisions on the difference of ciphertext pair and the internal state difference in penultimate and ante-penultimate rounds. These conclusions inspired us to investigate the feasibility of providing penultimate round information to train the neural distinguishers without knowing the last round subkey.

In deep learning, excellent results rely on large networks, which usually require large amounts of data to be properly trained. So data augmentation technology has been hot research in recent years. In 2019, Torres *et al.* [13] proposed a new method of data augmentation in autonomous driving. The neural network requires a database of road images containing traffic signs, and Torres *et al.* used a clever combined approach for substitution. They used the combination of traffic sign templates and arbitrary natural background images as training databases. The new models of constructing databases have solved three problems: expensive annotation, real images from the target domain, and balanced data sets. After training, their deep detector showed excellent results and indicated that detection models can still achieve good performance trained out of the context of the problem. The results are quite surprising because this is the opposite of common sense for deep learning.

Inspired by the above works, we have done some pioneering works to train better neural distinguishers by modifying the input data format. Our novelty input data generation methods are applied to the NSA ciphers Speck and Simon [14], which archive outstanding training results.

1.1 Our contributions

In this paper, our major contributions are listed as follows:

- First, we propose a new data generation model. The existing data format improvements stop in two directions: increasing the amount of ciphertext pairs in a single sample and changing the combination method with ciphertext pairs. Our model extends the data format from r-round to (r-1)-round, which is equivalent to improving the data format of training from one-dimensional to two-dimensional. Specifically, based on the round subkeys of the Markov cipher being assumed to be independent we first use randomly generated subkeys to decrypt one round of the ciphertext pair. And then we splice the results and ciphertext pairs to develop a new input data format.

- Next, we validate that the proposed new input data format has excellent features through experiments and theoretical analysis. We design three experiments step-by-step to present the validity and innovativeness of the new data generation model and obtain positive results. (1) Validity: the training dataset generated by using the new model can obtain effective neural distinguishers with high accuracy. (2) Discrepancy: the new data format provided new learnable features for neural networks compared to the single ciphertext pair(Gohr's input data format). (3) Scalability: stitching two different data formats can further improve the accuracy of neural distinguishers. Besides, as supplementary, we show with an experiment that expanding the number of rounds in the data format may not improve the accuracy but increases the amount of data for a single sample.

- Finally, considering the neural distinguishers in the above experiments do not fully capture the features contained in the spliced data format, we provide more ciphertext pairs in a single sample and generate a new data format MRMSP(Multiple Rounds Multiple Splicing Pairs). Then we convert the ciphertext pairs to output differences and generate a new data format MRMSD(Multiple Rounds Multiple Splicing Differences). We apply the new two formats to build neural distinguishers for Speck and Simon families. To the best of our knowledge, we achieve the longest rounds and the highest accuracy of neural distinguishers for Speck32, Speck48, and Speck64. Specifically, for Speck48, we obtain an effective 8-round distinguisher for the first time. For Simon48, we obtain the highest accuracy of the (10-11)-rounds neural distinguishers and first obtain an effective 12-round distinguisher. For Simon64, we achieve a breakthrough in accuracy on the existing number of rounds. Table 1 shows the new results of the neural distinguishers for Speck and Simon.

1.2 Outlines

This paper is organized as follows. In Section 2, we introduce Markov cipher and give a brief description of Speck and Simon as well as review the existing neural distinguishers model. Section 3 presents a new method of input data generation and verifies the properties of the proposed new data format through experiments and theoretical analysis. We adopt the ideas of multiple cipher pairs and multiple output differences in a single sample to improve the new input data format and then apply them to Speck and Simon in Section 4. Finally, our work is summarized in Section 5.

Table 1: Summary of the neural distinguishers for Speck and Simon family

Ciphers	Data Format	Round	Input Difference	Accuracy	Source
Speck32/64	MCP	7	(0x40, 0x0)	66.94%	[9]
	MCP^2	7	(0x40, 0x0)	89.63%	[8]
	MOD	7	(0x40, 0x0)	88.19%	[10]
	MRMSP	7	(0x40, 0x0)	89.16%	Sect. 4.2
	MRMSD	7	(0x40, 0x0)	94.11%	Sect. 4.2
	MCP^2	8	(0x2800, 0x10)	58.53%	[8]
	MOD	8	(0x2800, 0x10)	56.49%	[10]
	MRMSP	8	(0x2800, 0x10)	57.74%	Sect. 4.2
	MRMSD	8	(0x2800, 0x10)	65.02%	Sect. 4.2
Speck48/96	MOD	7	(0x20082,0x120200)	63.43%	[10]
	MRMSP	7	(0x20082, 0x120200)	57.17%	Sect. 4.2
	MRMSD	7	(0x20082, 0x120200)	71.38%	Sect. 4.2
	MRMSD	8	(0x20082, 0x120200)	54.62%	Sect. 4.2
Speck64/128	MOD	8	(0x1202, 0x2000002)	63.20%	[10]
	MRMSD	8	(0x1202, 0x2000002)	71.81%	Sect. 4.2
Simon32/64	MOD	9	(0x0, 0x80)	82.27%	[10]
	MOD-CP	9	(0x0, 0x80)	99.34%	[11]
	MRMSP	9	(0x0, 0x80)	96.30%	Sect. 4.2
	MRMSD	9	(0x0, 0x80)	99.08%	Sect. 4.2
	MOD	10	(0x0, 0x80)	61.09%	[10]
	MOD-CP	10	(0x0, 0x80)	86.08%	[10]
	MRMSP	10	(0x0, 0x80)	78.72%	Sect. 4.2
	MRMSD	10	(0x0, 0x80)	83.02%	Sect. 4.2
	MOD-CP	11	(0x20, 0x88)	59.55%	[11]
	MRMSP	11	(0x0, 0x80)	56.16%	Sect. 4.2
	MRMSD	11	(0x0, 0x80)	60.81%	Sect. 4.2
Simon48/96	MOD	10	(0x0, 0x100000)	81.40%	[10]
	MRMSP	10	(0x1000, 0x4400)	82.13%	Sect. 4.2
	MRMSD	10	(0x1000, 0x4400)	99.55%	Sect. 4.2
	MOD	11	(0x1000, 0x4400)	61.43%	[10]
	MRMSP	11	(0x1000, 0x4400)	66.19%	Sect. 4.2
	MRMSD	11	(0x1000, 0x4400)	78.35%	Sect. 4.2
	MRMSD	12	(0x1000, 0x4400)	61.59%	Sect. 4.2
Simon64/128	MOD	11	(0x0, 0x10)	73.79%	[10]
	MOD-CP	11	(0x0, 0x4)	99.28%	[11]
	MRMSP	11	(0x0, 0x10)	95.01%	Sect. 4.2
	MRMSD	11	(0x0, 0x10)	99.95%	Sect. 4.2
	MOD	12	(0x0, 0x10)	69.57%	[10]
	MOD-CP	12	(0x0, 0x4)	83.78%	[11]
	MRMSP	12	(0x0, 0x10)	75.06%	Sect. 4.2
	MRMSD	12	(0x0, 0x10)	93.86%	Sect. 4.2
	MOD-CP	13	(0x0, 0x40)	60.32%	[11]
	MRMSD	13	(0x0, 0x10)	70.10%	Sect. 4.2

[1] We choose the highest accuracy of NDs in these papers.

[2] MCP: Multiple Ciphertext Pairs. MOD: Multiple Output Differences. MCP^2: Adding the correct part of decrypting one round into MCP. MOD-CP: Multiple Output Differences and Ciphertext Pairs. MRMSP: Multiple round Multiple Splicing Pairs. MRMSD: Multiple round Multiple Splicing Differences.

Table 2: The notations

Notation	Description
Simon $2n/nm$	Simon acting on $2n$-bit plaintext blocks and using $nm(k)$-bit key
Speck $2n/nm$	Speck acting on $2n$-bit plaintext blocks and using $nm(k)$-bit key
$\oplus$	Bitwise XOR
&	Bitwise AND
$S^{\alpha}(x)$	Circular left shift of x by α bits
K	Master key
rk_i	i-round subkey
Δ_{in}	Input difference
(P, P')	Plaintext pair
(C, C')	Ciphertext pair
(P_l, P_r, P'_l, P'_r)	$P = P_l \parallel P_r$ and $P' = P'_l \parallel P'_r$
(C_l, C_r, C'_l, C'_r)	$C = C_l \parallel C_r$ and $C' = C'_l \parallel C'_r$
$(\Delta C_l, \Delta C_r)$	$\Delta C_l = C_l \oplus C'_l$ and $\Delta C_r = C_r \oplus C'_r$

2 Preliminaries

2.1 Notations

Table 2 presents the major notations.

2.2 Markov Cipher

Markov chain is defined in [15] which is a sequence of random variables $v_0, v_1, ..., v_r$, the current state of variable v only depends on the previously adjacent state.

Definition 1 (Markov Chain [15]). Given a sequence of discrete random variables $v_0, v_1, ..., v_r$ is a Markov chain, if for $0 < i < r$,

$$Pr(v_{i+1} = \beta_{i+1} \mid v_i = \beta_i, v_{i-1} = \beta_{i-1}, \ldots, v_0 = \beta_0) = Pr(v_{i+1} = \beta_{i+1} \mid v_i = \beta_i).$$

Given a group operation $\otimes$, we define the corresponding input differences as $\Delta Y_0, \Delta Y_1, \ldots, \Delta Y_r$, where $\Delta Y_i = Y_i \otimes Y'_i$. Then, Lai *et al.* [2] introduce the following definition of Markov cipher as given in Definition 2.

Definition 2 (Markov Cipher [2]). An iterated cipher with round function $Y = f(X, K)$ is a Markov cipher if there is a group operation $\otimes$ for defining differences such that, for all choices of $\alpha(\alpha \neq 0)$ and $\beta(\beta \neq 0)$, $Pr(\Delta Y = \beta \mid \Delta X = \alpha, X = \gamma)$ is independent of γ when the subkey K is uniformly random, or equivalently, if $\Pr(\Delta Y = \beta \mid \Delta X = \alpha, X = \gamma) = Pr(\Delta Y(1) = \beta_1 \mid \Delta X = \alpha)$ for all choices of γ when the sub-key K is uniformly random.

The concept of Markov cipher is introduced by Lai *et al.* in [2] due to its significance in differential cryptanalysis. He also proved that if an r-round iterated cipher is a Markov cipher, and the r-round subkeys are independent and uniformly random, then the sequence of differences $\Delta = \Delta Y(0), \Delta Y(1), ..., \Delta Y(r)$ is a homogenous Markov chain.

2.3 Brief Description of Speck and Simon

Speck and Simon are two iterated block ciphers proposed by the National Security Agency (NSA) [14]. They adopt ARX construction that applies a combination of rotation, XOR, and either addition (Speck) or the logical AND (Simon) iteratively over some rounds.

Both cipher families are defined for state sizes $2n$ and key sizes k: 32/64, 48/72, 48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, and 128/256.

For Speck $2n/nm$, the round function $F : F_2^k \times F_2^{2k} \to F_2^{2k}$ takes as input a k-bit subkey rk_i and a cipher state consisting of two w-bit words (L_i, R_i) and produces from this the next round state (L_{i+1}, R_{i+1}) as follows:

$$L_{i+1} = \left(\left(S^{-\alpha} L_i\right) \& R_i\right) \oplus rk_i, R_{i+1} = L_{i+1} \oplus \left(S^{\beta} R_i\right) \tag{1}$$

where α, β are constants specific to each member of the Speck cipher family ($\alpha = 7$, $\beta = 2$ for Speck32/64 and $\alpha = 8$, $\beta = 3$ for the other variants).

For Simon $2n/mn$, the round function $F : F_2^k \times F_2^{2k} \to F_2^{2k}$ as follows:

$$L_{i+1} = f\left(L_i\right) \oplus R_i \oplus rk_i, R_{i+1} = L_i \tag{2}$$

where $f(x) = \left(\left(S^1 x\right) \& \left(S^8 x\right)\right) \oplus \left(S^2 x\right)$.

2.4 Overview of Existing Neural Distinguishers Model

Gohr [6] trained a deep neural network for classifying accurately random from real ciphertext pairs. It is the first known machine learning model that successfully performed cryptanalysis tasks on modern ciphers. Chen *et al.* [9] and Hou *et al.* [10] improved Gohr's neural distinguishers, respectively, by changing the input data format of the neural distinguishers. We take Speck32/64 as an example to introduce three input data formats in detail.

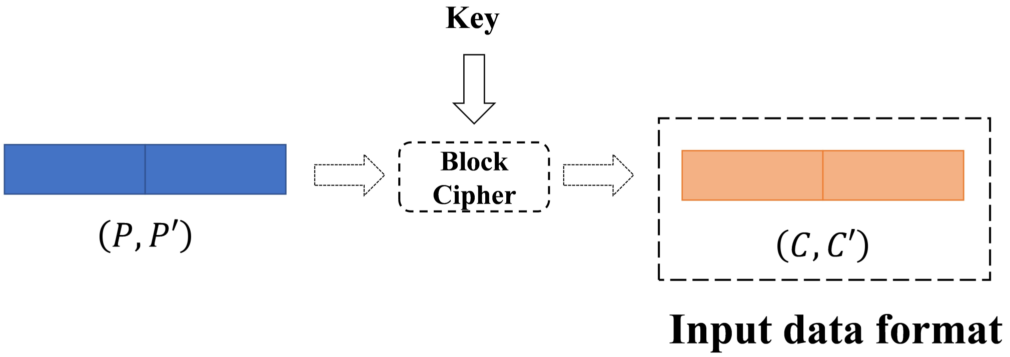

Figure 1: Single Ciphertext Pair

Figure 1 shows Gohr's input data generation process. The single plaintext pair (P, P') is encrypted by a random master key to obtain single ciphertext pair (C, C') as a training sample. During the training of the neural network, each sample is given a label taking the value 0 or 1. The value 1 means that data pairs are generated from encrypting (P, P') with input difference Δ_{in}, and the value 0 means that the data pair is generated from random pair.

Figure 2 presents Chen's data generation, the m plaintext pairs $\{(P_1, P_1'), ..., (P_m, P_m')\}$ are encrypted by a random master key to obtain m ciphertext pairs $(C_1, C_1'), ..., (C_m, C_m')$ as a training sample. Same as Gohr's method, each training sample labeled by a value 0 or 1, where 0 means $(C_1, C_1'), ..., (C_m, C_m')$ is generated randomly, and 1 means $(C_1, C_1'), ..., (C_m, C_m')$ generation from $(P_1, P_1'), ..., (P_m, P_m')$ with a particular input difference Δ_{in}.

Comparing with multiple ciphertext pairs, Hou *et al.* converted the m ciphertext pairs $(C_1, C_1'), ..., (C_m, C_m')$ to output differences. So Hou's input data format is

$$\{(\Delta C_{1,l}, \Delta C_{1,r}), ..., (\Delta C_{m,l}, \Delta C_{m,r})\},$$

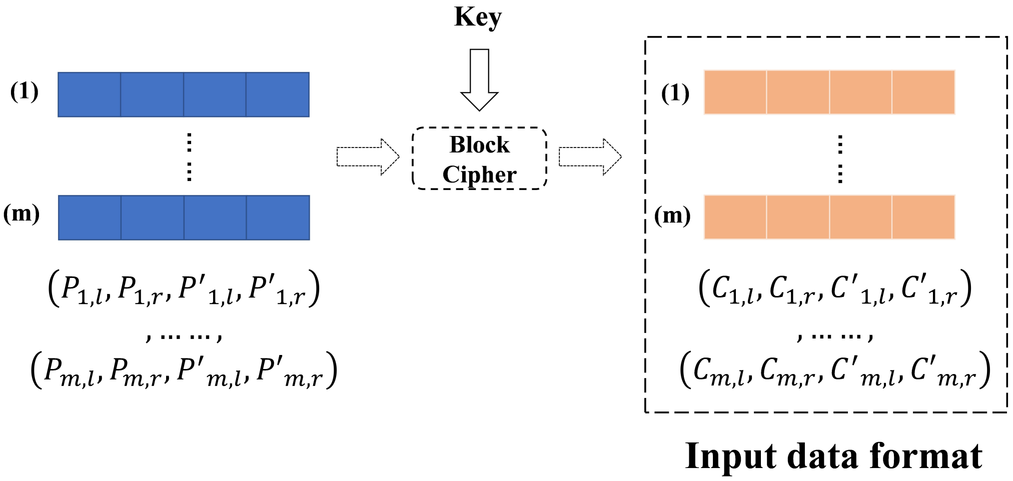
Key
(1)
(m)
Block Cipher
(1)
(m)
$(P_{1,l}, P_{1,r}, P'_{1,l}, P'_{1,r})$
, … …,
$(P_{m,l}, P_{m,r}, P'_{m,l}, P'_{m,r})$
$(C_{1,l}, C_{1,r}, C'_{1,l}, C'_{1,r})$
, … …,
$(C_{m,l}, C_{m,r}, C'_{m,l}, C'_{m,r})$
Input data format

Figure 2: Multiple Ciphertext Pairs

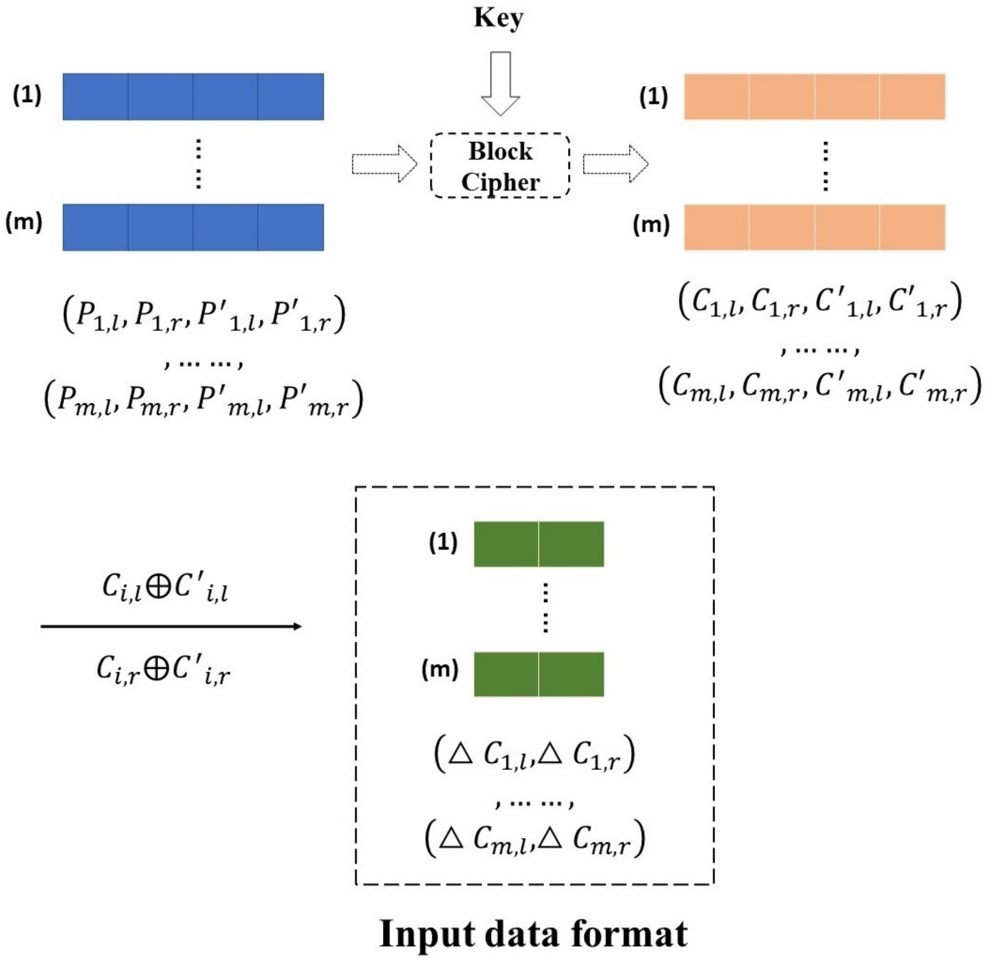
Key
(1)
(m)
Block Cipher
(1)
(m)
$(P_{1,l}, P_{1,r}, P'_{1,l}, P'_{1,r})$
, … …,
$(P_{m,l}, P_{m,r}, P'_{m,l}, P'_{m,r})$
$(C_{1,l}, C_{1,r}, C'_{1,l}, C'_{1,r})$
, … …,
$(C_{m,l}, C_{m,r}, C'_{m,l}, C'_{m,r})$
$C_{i,l} \oplus C'_{i,l}$
$C_{i,r} \oplus C'_{i,r}$
(1)
(m)
$(\triangle C_{1,l}, \triangle C_{1,r})$
, … …,
$(\triangle C_{m,l}, \triangle C_{m,r})$
Input data format

Figure 3: Multiple Output Differences

together with a label taking the value 0 or 1. The detailed input data generation process is shown in Figure 3.

We define the format of Gohr's, Chen's, and Hou's input data as SCP(Single Ciphertext Pair), MCP(Multiple Ciphertext Pairs), and MOD(Multiple Output differences), and define the corresponding neural distinguishers as ND_{SCP}, ND_{MCP} and ND_{MOD}.

To assure the neural accuracy of distinguishers, a large number of samples are used for network training. If a neural distinguisher can obtain a stable distinguishing accuracy higher than 0.5 on a validation dataset, it can effectively distinguish ciphertext and a random value.

The neural distinguishers model can be described as:

$$\begin{gathered}\operatorname{Pr}\left(Y=1 \mid X_0, \cdots, X_{m-1}\right) \\ =F\left(f\left(X_0\right), \cdots, f\left(X_{m-1}\right), \varphi\left(f\left(X_0\right), \cdots, f\left(X_{m-1}\right)\right)\right)\end{gathered} \tag{3}$$

when $m=1, X_0=(C_0, C_0')$ for ND_{SCP}, $m \in \{2,4,8,16\}, X_i=(C_i, C_i')$ for ND_{MCP} and $m \in \{32,48,64\}, X_i=(\Delta C_i, \Delta C_i')$ for ND_{MOD}. If $Pr(Y=1 \mid X_0, \cdots, X_{m-1})>0.5$, the label of $(X_0, \cdots, X_{m-1})$ predicted by a neural distinguisher is 1, otherwise the prediction result is 0.

3 New Input Data Generation Model

3.1 Motivation

In the beginning, Gohr [6] used the ciphertext pairs (C, C') as the input of the network for training distinguishers. Then, Chen *et al.* [9] proposed multiple groups of ciphertext pairs $(C_1, C_1'), .., (C_m, C_m')$ as the training sample for providing more features to improve the accuracy of neural distinguishers. Recently, Hou *et al.* [10] built multiple groups of output differences pairs $\{(\Delta C_{1,l}, \Delta C_{1,r}), ..., (\Delta C_{m,l}, \Delta C_{m,r})\}$ instead of multiple groups of ciphertext pairs and further improved accuracy of neural distinguishers on several versions of Speck and Simon. By summarizing the existing neural distinguishers model, we find that the current approaches are suffering from the following disadvantages. (1) The source of data features is limited to linear combinations of ciphertexts and provides limited learnable features to the training samples for increasing the accuracy, which can be improved by constructing data fully utilizing the traditional cryptanalysis technique, like differential features and inner state information as Markov cipher. (2) Lacking breakthroughs in data format construction from a deep learning perspective. Deep learning is becoming more and more mature in various fields, and there are abundant ways to construct sample formats for network training. We can select some reasonable ideas and apply them to neural distinguishers.

To overcome the above drawbacks, we propose a new data format to train neural distinguishers, considering both the domain knowledge about deep learning and information on differential cryptanalysis.

First, in deep learning, solving challenging tasks with deep neural networks usually requires an annotated database with real samples belonging to the context of the problem. The human effort and other costs of gathering such data have motivated research on alternative ways to train the models. In autonomous driving, Torres *et al.* [13] proposed a more flexible and effortless construction method of the training dataset by superposing the templates on natural images. The generation of the training database consists of three steps. Primarily, templates of the traffic signs of interest are acquired. Then, background images that do not belong to the domain of interest are collected (i.e., random natural images). Lastly, the training samples (i.e., images with annotated traffic signs) are generated. The novelty training data generation method showed that detectors can be trained without problem domain data for the background. This is quite surprising because it is the opposite of common sense for deep learning. Therefore, we will explore ways to apply the idea of non-realistic training samples to the neural distinguishers.

Second, in cryptanalysis, Benamira *et al.* [12] indicates ND_{SCP} can identify some information for the penultimate rounds. The conclusion inspires us to investigate the feasibility of providing penultimate round information to construct training templates of the neural distinguishers. However, the subkeys for each round are unknown, which makes it impossible to obtain the real output of the penultimate round from the ciphertexts. We combine the concept of Markov cipher and assume the round subkeys are independent and uniformly random, thus we can use random subkeys to decrypt one round of the deterministic ciphertext pairs for one round to generate a non-realistic output for the penultimate round. Based on the result, we design the new input data generation mode and verify the validity and correctness through a series of experiments.

3.2 A New Input Data Generation Model

Considering a cipher E and a plaintext difference Δ_{in}, the new generation model of the positive sample consists three-step.

Step 1. Ciphertext pair acquisition

The first step is to generate cipher pair (P, P') with Δ_{in}. For the plaintext pair (P, P'), it is encrypted using a randomly generated encryption key to obtain the ciphertext pair (C, C').

Step 2. “Decryption" process

The second step is using random subkey to decrypt one round of (C, C') and defining the result as $(C^{(1)}, C'^{(1)})$.

Step 3. Training samples generation

The last step is splicing the ciphertext pair (C, C') and the corresponding $(C^{(1)}, C'^{(1)})$ to generate the training sample $(C, C', C^{(1)}, C'^{(1)})$.

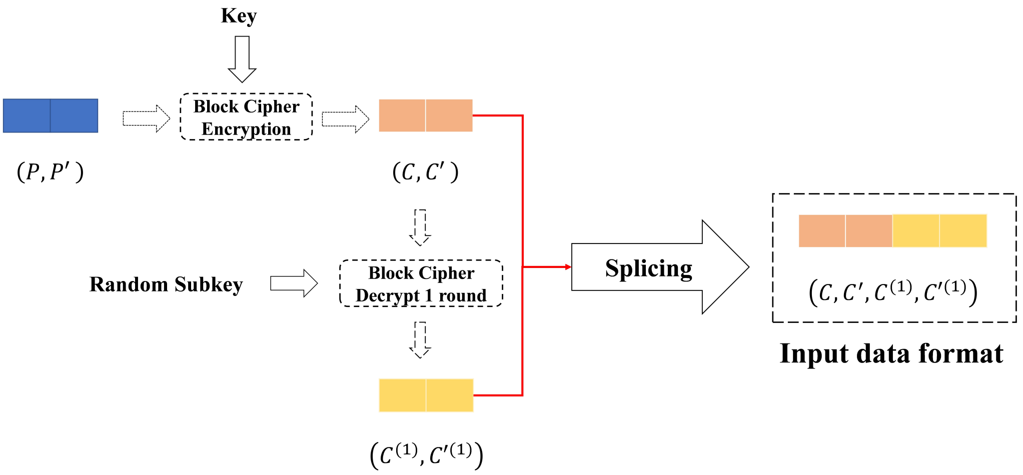

Figure 4: New data generation process

The proposed method (illustrated in Figure 4) mainly comprises the generation of input data which requires a random last round subkey to decrypt the ciphertext pair. Gohr used (C, C') as a single training sample, and we splice $(C^{(1)}, C'^{(1)})$ directly after (C, C') to form a novel sample format $(C, C', C^{(1)}, C'^{(1)})$. Compared to Gohr's format, our training data generation process has three advantages:

1. The more information contained in a single training sample, the more features neural distinguishers can learn. The new input data format provides two different forms

Table 3: Parameters of the Network Architecture for training ND

Hyperparameters	Value	Hyperparameters	Value
Train size	10^7	Conv size(k_s)	3
Validation size	10^6	Regularization parm	10^{-4}
Batch size	10000	Optimizer	Adam
Epochs	50	Loss function	MSE

Table 4: Results of Experiment 1: Accuracy of ND_{SDP} and ND_{SCP}

	Round	Accuracy	
		ND_{SDP}	ND_{SCP}
Speck32/64	5	0.9033	0.9290
	6	0.7326	0.7880

of features for neural distinguishers which contributes to improving the accuracy of neural distinguishers.

2. A largescale database can be generated without much cost, since there are a lot of different possible combinations between the random subkeys and the ciphertext pairs, we can construct large amounts of data to be properly trained.

3. The new input data format opens up a new direction of data format generation for neural differential distinguishers and provides more room for improvement in the future development of neural differential cryptanalysis.

3.3 Analyzing the Input Data Generation Model

In this section, we experimentally verify the properties of the new input data format. For comparison with Gohr's results, the network used for training in the experiments is the same as Gohr's except for the input, which consists of four components: input representation, initial convolution, convolutional block, and prediction head. And the neural network training parameters are shown in Table 3. Meanwhile, we limit and focus the discussions and results on the neural distinguishers for 5-round and 6-round Speck32/64.

We refer to $(C^{(1)}, C'^{(1)})$ as SDP(Single Decryption Pair), and refer to the new proposed data format $(C, C', C^{(1)}, C'^{(1)})$ as SOP(Splicing Output Pair). The new data format generation model is effective when the accuracy of the neural distinguishers trained with the new data format is higher than 0.5. Meanwhile, the higher the accuracy rate is, the more effective the new data format is. So, a natural question is:

Are the distinguishers trained with the new data format valid?

We do the following experiments to verify the validity.

Experiment 1: Validity verification

1. Generate 10^7 plaintext pairs such that $\frac{1}{2}$ of the pairs satisfy with initial difference $\Delta_{in} = 0x0040/0000$.

2. Encrypt the plaintext pairs with random keys for 5(6)-round Speck32/64 to generate 10^7 ciphertext pairs as the training samples.

3. Randomly generate 10^7 "subkeys" to decrypt one round of ciphertext pairs.

4. Train the neural network with 10^7 "decrypt one round" results and record validation database accuracy.

The results of Experiment 1 are shown in Table 4 including the accuracy of ND_{SDP} and comparison with Gohrs's ND_{SCP} for Speck32/64 in [6]. For 5-round Speck32/64, the accuracy of ND_{SDP} is 90.33%. For 6-round Speck32/64, the accuracy of ND_{SDP} is 73.26%. Observation results, our neural distinguishers are slightly less effective than ND_{SCP}. But the accuracy of (5-6)-rounds ND_{SDP} both exceeded 50%, indicating that

Table 5: Results of Experiment 2: Summary of evaluation accuracy

	Round	Evaluation accuracy	
		Experiment 2-1	Experiment 2-2
Speck32/64	5	0.4954	0.5048
	6	0.5028	0.4995

our new input data format can provide effective features for the neural distinguishers. So the validity of the new data generation model is established.

Since the trained distinguishers using two different data formats obtain similar accuracy, we start to look at the differences between them. Neural distinguishers learn the features from the training dataset. The foundation of neural distinguishers trained by using different data formats is different. Thus, we decide to use each of the two data formats as the validation set for the corresponding neural distinguishers of the other, and then explore the differences between them. Based on this, we conduct another experiment to answer the following question:

Are the features provided by the two data formats the same?

Experiment 2: Discrepancy verification

Experiment 2-1:

1. Generate 10^7 plaintext pairs such that $\frac{1}{2}$ of the pairs satisfy with initial difference $\Delta_{in} = 0x0040/0000$.

2. Encrypt the plaintext pairs with random keys for 5(6)-round Speck32/64 to generate 10^7 ciphertext pairs.

3. Randomly generate 10^7 subkeys to decrypt one round of ciphertext pairs to obtain 10^7 SDP.

4. Use 10^7 SDP as the evaluation dataset for ND_{SCP} to obtain the evaluation accuracy.

Experiment 2-2:

1. Generate 10^7 plaintext pairs such that $\frac{1}{2}$ of the pairs satisfy with initial difference $\Delta_{in} = 0x0040/0000$.

2. Encrypt the plaintext pairs with random keys for 5(6)-round Speck32/64 to generate 10^7 ciphertext pairs.

3. Use 10^7 SCP as the evaluation dataset for ND_{SDP} to obtain the evaluation accuracy.

In Table 5, we show the evaluation accuracies derived using each other's data formats for (5-6)-rounds of Speck32/64 which are both close to 50%. Indicating that the learned features by ND_{SCP} and ND_{SDP} are different. In other words, two data formats provide different features for neural networks. Thus, we verify that there are discrepant between the two data formats.

At the same time, we find that in the process of training both neural networks, the accuracy of the training sets is going to exceed the accuracy of the validation set, producing an overfitting phenomenon. So we believe that the neural network has adequately captured the features contained in both data formats. Then, we can improve the accuracy of the neural distinguisher by adding more learnable features to the existing data formats. Considering the methods of data augmentation in the field of image recognition, we venture the following conjecture:

Splicing the two formats together can provide more learnable features, which can further improve the accuracy of neural distinguishers.

We denote the splicing pair$(C, C', C^{(1)}, C'^{(1)})$ as SOP(Splicing Output Pair). In order to verify the conjecture, we perform the following experiment:

Experiment 3: Scalability verification

1. Generate 10^7 plaintext pairs such that $\frac{1}{2}$ of the pairs satisfy with initial difference $0x0040/0000$.

Table 6: Results of Experiment 3: Accuracy of ND_{SOP} and compare to ND_{SDP}

	Round	Accuracy	
		ND_{SDP}	ND_{SOP}
Speck32/64	5	0.9033	0.9293
	6	0.7326	0.7881

Table 7: Results of Experiment 4: Accuracy of ND_{3r-SOP} and compare to ND_{SOP}

	Round	Accuracy	
		ND_{3r-SOP}	ND_{SOP}
Speck32/64	5	0.9104	0.9127
	6	0.7538	0.7619

2. Encrypt the plaintext pairs with random keys for 5(6)-round Speck32/64 to generate 10^7 ciphertext pairs.

3. Randomly generate 10^7 subkeys to decrypt one round of ciphertext pairs to generate 10^7 SDP.

4. Splice the SCP and SDP together to generate 10^7 new training samples and donate them as SOP(Splicing Output Pair).

5. Train the neural network with 10^7 SOP and the parameters in Table 3 and record validation dataset accuracy.

The results of Experiment 3 are shown in Table 6 including the accuracy rates of two neural distinguishers. The accuracy of ND_{SOP} is better than the accuracy of ND_{SDP} for (5-6)-rounds of Speck32/64, which indicates that our extensions based on SDP are successful.

In addition, we consider possible directions for further expansion of the data format. In [12], Benamira *et al.* gave a detailed explanation of Gohr's neural distinguishers which showed neural distinguishers make their decisions on the difference of ciphertext pair and the internal state difference in penultimate and ante-penultimate rounds. So we decide to continue to extend our data format with the conjecture verified by Experiment 3. We consider the feasibility of providing ante-penultimate round information for neural networks, thus extending the data format from a two-round structure to a three-round structure. We splice the data from two rounds using random key decryption after ciphertext pairs, allowing more data information to be contained in a single training sample. Combining the above discussion, we pose the following key question and designed an experiment to explore the answers.

Can the accuracy of the distinguisher be improved by expanding the number of rounds in the data format? We perform the following experiment as supplementary:

Experiment 4: Continuability verification

1. Generate 10^7 plaintext pairs such that $\frac{1}{2}$ of the pairs satisfy with initial difference $0x0040/0000$.

2. Encrypt the plaintext pairs with random keys for 5(6)-round Speck32/64 to generate 10^7 ciphertext pairs.

3. Randomly generate 10^7 two-round subkeys to decrypt two rounds of ciphertext pairs to generate 10^7 results.

4. Splicing the SCP and results together to generate 10^7 new training samples and donate this format as 3r-SOP(3-round Splicing Output Pair).

5. Train the neural network with 10^7 3r-SOP and the parameters in Table 3 and record validation dataset accuracy.

Table 7 shows the accuracy comparison of the above experiment with (5-6)-rounds Speck32/64. The accuracy of ND_{3r-SOP} is approximated by the accuracy of ND_{SOP}.

Therefore, the 3-round structure does not improve the accuracy but increases the amount of data for a single sample. According to the above analysis, we find that there is little evidence showing the longer of randomly decrypted rounds in the input data format, the higher the accuracy of the obtained neural distinguishers.

In our data format generation process, only the ciphertext pairs obtained by master key encryption are real, while the rest of the data decrypted using random subkeys are non-real. For Speck, the characteristics of the i-round difference feature are completely determined by the statistical properties of the (i-1)-round difference as the Markov ciphers. So the results of the two rounds of decryption by random keys have little connection with the real ciphertext pairs. This is a possible explanation of the experimental results in Table 7.

Through the four experiments described above, we verify the properties of the proposed new data format generation model. First, we validate the effectiveness of the proposed data generation model in Experiment 1 by observing the accuracy of the neural distinguishers trained using the new data generation model. Second, we perform a preliminary exploration of the features contained in our data format and conclude that it contains learnable features that are different from those contained in Gohr's data format in Experiment 2. Third, we borrow the idea of data augmentation in deep learning to propose an improvement on the new data format by extending the data format into two rounds and using it to train the neural distinguishers in Experiment 3, which achieves better results. Finally, as supplementary, we explore the feasibility of further improving the data format by extending it from two rounds to three rounds in Experiment 4. However, the 3-round structure does not improve the accuracy but increases the amount of data for a single sample.

There is still a problem that remains to be solved. The accuracy of ND_{SOP} is not as good as the accuracy of ND_{SCP}. It may be the reason that the input data format SOP contains more features for the network, but the limitation of the amount of data in a single sample causes ND_{SOP} not fully utilize the features in the sample. To address the problem, we propose two improved neural distinguishers and apply them to improve the neural distinguishers of Speck and Simon.

4 Improved Neural Distinguishers of Speck and Simon

For 5-round and 6-round Speck, the accuracies of ND_{SOP} are higher than ND_{SDP}, but still lower than ND_{SCP}. It could be the limitation of data volume that the distinguishers do not fully utilize the features in the sample. We take inspiration from the format MCP(Multiple Ciphertext Pairs) [9] and MOD(Multiple Output Difference) [10] to improve our new format SOP and present new distinguisher models, which can apply to obtain the high accuracy of neural distinguishers for Speck and Simon.

4.1 New Neural Distinguishers mdoels

- *Improved input data format by multiple ciphertext pairs*

Extending a single ciphertext pair into multiple ciphertext pairs:

As shown in Figure 5, the m plaintext pairs $(P_1, P_1', ..., P_m, P_m')$ are encrypted by random master keys to generate m ciphertext pairs $(C_1, C_1', ..., C_m, C_m')$. The m ciphertext pairs $(C_1, C_1', ..., C_m, C_m')$ are decrypted one round by random “subkeys" to generate m pairs:

$$(C_1^{(1)}, C_1'^{(1)}, ..., C_m^{(1)}, C_m'^{(1)}).$$

Finally, we splice two type pairs to generate a new input data format:

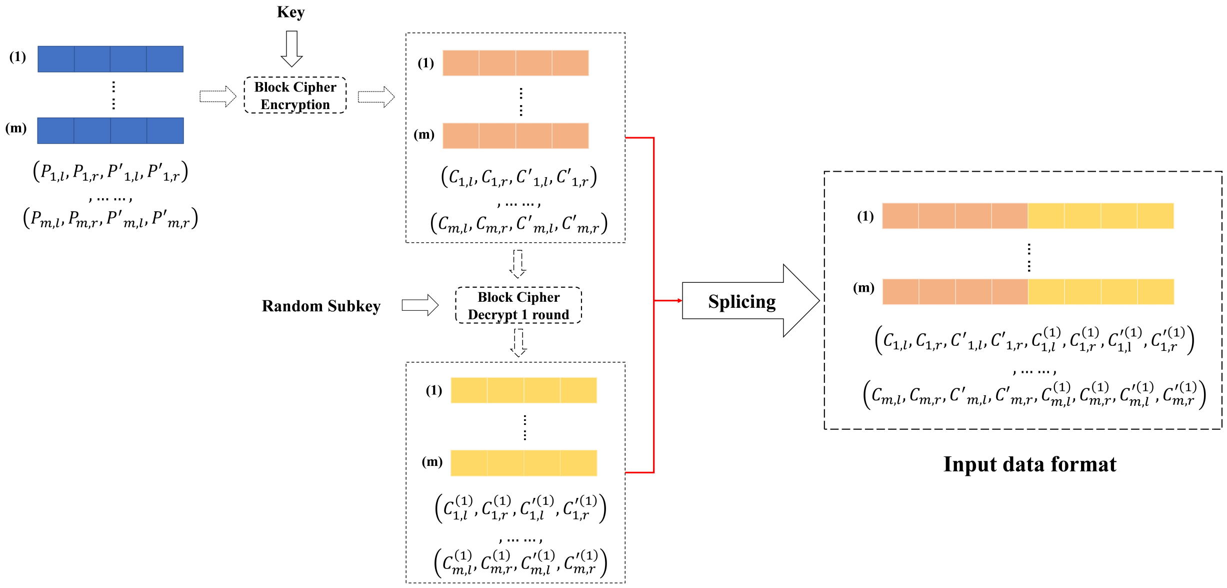
Key
(1)
(m)
Block Cipher Encryption
$(P_{1,l}, P_{1,r}, P'_{1,l}, P'_{1,r})$
, ... ,
$(P_{m,l}, P_{m,r}, P'_{m,l}, P'_{m,r})$
$(C_{1,l}, C_{1,r}, C'_{1,l}, C'_{1,r})$
$(C_{m,l}, C_{m,r}, C'_{m,l}, C'_{m,r})$
Random Subkey
Block Cipher Decrypt 1 round
$(C^{(1)}_{1,l}, C^{(1)}_{1,r}, C'^{(1)}_{1,l}, C'^{(1)}_{1,r})$
$(C^{(1)}_{m,l}, C^{(1)}_{m,r}, C'^{(1)}_{m,l}, C'^{(1)}_{m,r})$
Splicing
$(C_{1,l}, C_{1,r}, C'_{1,l}, C'_{1,r}, C^{(1)}_{1,l}, C^{(1)}_{1,r}, C'^{(1)}_{1,l}, C'^{(1)}_{1,r})$
$(C_{m,l}, C_{m,r}, C'_{m,l}, C'_{m,r}, C^{(1)}_{m,l}, C^{(1)}_{m,r}, C'^{(1)}_{m,l}, C'^{(1)}_{m,r})$
Input data format

Figure 5: Multiple Rounds Multiple Splicing Pairs

$$(C_1, C'_1, C^{(1)}_1, C'^{(1)}_1, ..., C_m, C'_m, C^{(1)}_m, C'^{(1)}_m),$$

we define the new data format as MRMSP(Multiple Rounds Multiple Splicing Pairs).

- *Improved input data format by multiple output differences*

Converting the output pairs in MRMSP to output differences:

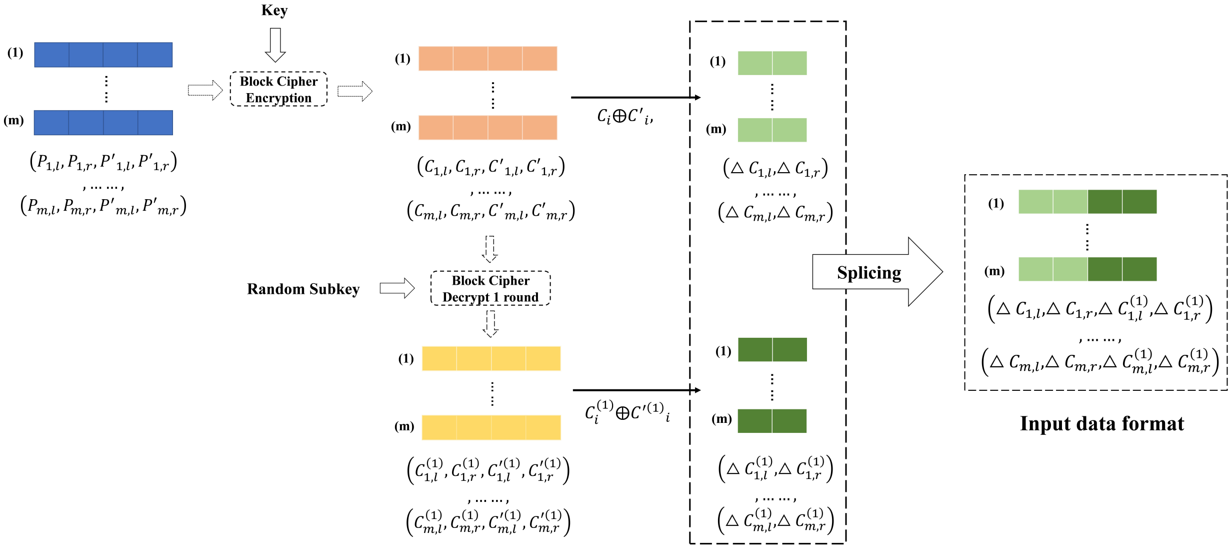
Key
(1)
(m)
Block Cipher Encryption
$(P_{1,l}, P_{1,r}, P'_{1,l}, P'_{1,r})$
, ... ,
$(P_{m,l}, P_{m,r}, P'_{m,l}, P'_{m,r})$
$(C_{1,l}, C_{1,r}, C'_{1,l}, C'_{1,r})$
$(C_{m,l}, C_{m,r}, C'_{m,l}, C'_{m,r})$
$C_i \oplus C'_i$
$(\triangle C_{1,l}, \triangle C_{1,r})$
$(\triangle C_{m,l}, \triangle C_{m,r})$
Random Subkey
Block Cipher Decrypt 1 round
$(C^{(1)}_{1,l}, C^{(1)}_{1,r}, C'^{(1)}_{1,l}, C'^{(1)}_{1,r})$
$(C^{(1)}_{m,l}, C^{(1)}_{m,r}, C'^{(1)}_{m,l}, C'^{(1)}_{m,r})$
$C^{(1)}_i \oplus C'^{(1)}_i$
$(\triangle C^{(1)}_{1,l}, \triangle C^{(1)}_{1,r})$
$(\triangle C^{(1)}_{m,l}, \triangle C^{(1)}_{m,r})$
Splicing
$(\triangle C_{1,l}, \triangle C_{1,r}, \triangle C^{(1)}_{1,l}, \triangle C^{(1)}_{1,r})$
$(\triangle C_{m,l}, \triangle C_{m,r}, \triangle C^{(1)}_{m,l}, \triangle C^{(1)}_{m,r})$
Input data format

Figure 6: Multiple Rounds Multiple Splicing Differences

As shown in Figure 6, the m plaintext pairs $(P_1, P'_1, .., P_m, P'_m)$ are encrypted by a random master key to generate m ciphertext pairs:

$$(C_1, C'_1, ..., C_m, C'_m).$$

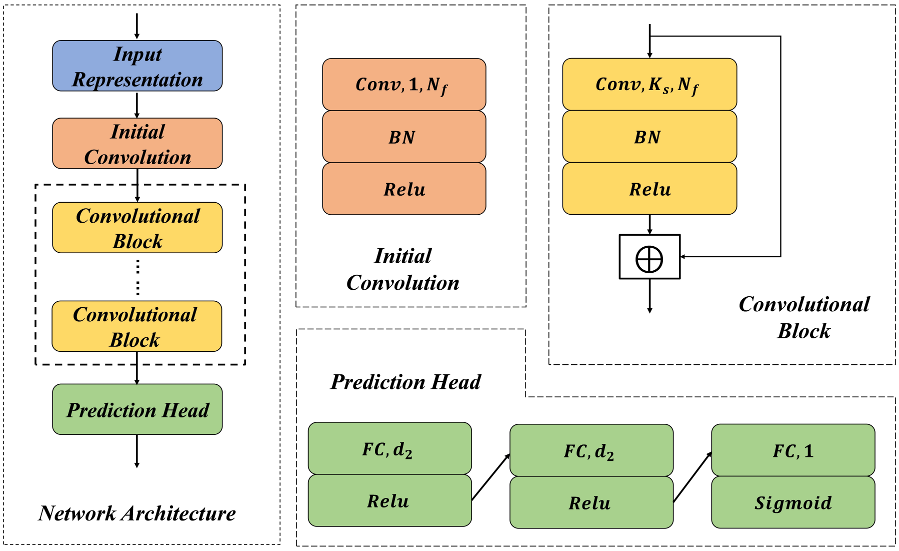

Figure 7: Multiple Rounds Multiple Splicing Differences

The result of decrypting one round of $(C_1, C'_1, ..., C_m, C'_m)$ with randomly generated subkeys are recorded as

$$(C_1^{(1)}, C_1'^{(1)}, ..., C_m^{(1)}, C_m'^{(1)}).$$

The $(C_1, C'_1, ..., C_m, C'_m)$ and $(C_1^{(1)}, C_1'^{(1)}, ..., C_m^{(1)}, C_m'^{(1)})$ are converted to output differences. Finally, we splice two output differences to generate a new input data format:

$$(\Delta C_1, \Delta C_1^{(1)}, ..., \Delta C_m, \Delta C_m^{(1)}).$$

We define the new input data format as MRMSD(Multiple Rounds Multiple Splicing Differences).

Similar to Gohr's training dataset [6], each sample will be attached a label Y according to the following equation:

$$Y = \begin{cases} 1, if\ P_i \oplus P'_i = \Delta_{in}, i \in [1, m] \\ 0, if\ P_i \oplus P'_i \neq \Delta_{in}, i \in [1, m] \end{cases} \tag{4}$$

If the label is 1, the data is denoted as a positive sample. Otherwise, it is denoted as a negative sample. Because the neural distinguishers learn the features of the target cipher instead of the features of the plaintext or key. So we request the plaintext pairs in a sample are randomly generated and the encryption keys are also randomly generated and different. All the requirements are designed to ensure that the m splicing output pairs(MRMSP and MRMSD) do not have any identical properties except for the same plaintext difference constraint. Due to the increased data amount in a single sample, for the same number of training sets, MRMSP and MRMSD require $2m$ and m times more data than Gohr's data generate the model(SCP), respectively.

To illustrate the effect of data format, we train the new neural distinguishers and focus on the accuracy rate. We use the network similar to Gohr's work [6]. The network consists of four parts: an input layer for processing training datasets, an initial convolutional layer, a residual tower consisting of multiple two-layer convolutional neural networks, and a prediction head consisting of fully connected layers. Because only the channel dimension is changed, we refer to Figure 7 for the description of network architecture.

We define the new neural distinguishers using MRMSP data format as ND_{MRMSP} and neural distinguishers trained with MRMSD data format as ND_{MRMSD}.

Table 8: Summary of the neural distinguishers for Speck family

Ciphers	Data Format	Round	Input Difference	Accuracy	Source
Speck32/64	MCP	7	$(0x40, 0x0)$	66.94%	[9]
	MOD	7	$(0x40, 0x0)$	88.19%	[10]
	MCP^2	7	$(0x40, 0x0)$	89.63%	[8]
	MRMSP	7	$(0x40, 0x0)$	89.16%	Sect. 4.2
	MRMSD	7	$(0x40, 0x0)$	94.11%	Sect. 4.2
	MOD	8	$(0x2800, 0x10)$	56.49%	[10]
	MCP^2	8	$(0x2800, 0x10)$	58.53%	[8]
	MRMSP	8	$(0x2800, 0x10)$	57.74%	Sect. 4.2
	MRMSD	8	$(0x2800, 0x10)$	65.02%	Sect. 4.2
Speck48/96	MOD	7	$(0x20082, 0x120200)$	63.43%	[10]
	MRMSP	7	$(0x20082, 0x120200)$	57.17%	Sect. 4.2
	MRMSD	7	$(0x20082, 0x120200)$	71.38%	Sect. 4.2
	MRMSD	8	$(0x20082, 0x120200)$	54.62%	Sect. 4.2
Speck64/128	MOD	8	$(0x1202, 0x2000002)$	63.20%	[10]
	MRMSD	8	$(0x1202, 0x2000002)$	71.81%	Sect. 4.2

[1] We choose the highest accuracy NDs in these papers.

[2] MCP: Multiple Ciphertext Pairs. MOD: Multiple Output Differences. MCP^2: Adding the correct part of the decrypting a round into MCP. MRMSP: Multiple round Multiple Splicing Pairs. MRMSD: Multiple round Multiple Splicing Differences.

4.2 Applications to the NSA Block Ciphers

The training parameters are shown in Table 3 and setting N_f = state size as channels number of convolution kernel in convolutional blocks. For training data format, we set group size m = (word size)/2.

- **Application to Speck**

The results are presented in Table 8, the SCP refers to the input data format of Gohr's input data format: single ciphertext pair. The MOD refers to the input data format of Hou's multiple output differences. The MCP refers to the input data format of Chen's multiple ciphertext pairs. The MRMSP and MRMSD refer to our new input data format as shown in Section 4.1.

For Speck32/64, Chen *et al.* [9] trained effective (5-7)-rounds neural distinguishers against Speck32/64 and Hou *et al.* [10] improved the effective round to 8-round. Then, Zhang *et al.* [8] further improved accuracy in (5-8)-rounds for Speck32/64. For Speck48/96 and Speck64/128, Hou *et al.* [10] gave a 7-round neural distinguisher with the accuracy of 63.43% and an 8-round neural distinguisher with the accuracy of 63.20%, respectively.

Based on the new input data format MRMSP and MRMSD, we comprehensively improve the accuracy of neural distinguishers for Speck32/64, Speck48/96, and Speck64/128. We build neural distinguishers against Speck32/64 cover to (7-8)-rounds with the accuracy of 94.11% and 65.02%, respectively. For Speck48/96, our new neural distinguishes not only greatly improve the accuracy of the 7-round neural distinguisher to 71.38%, but first achieve an effective 8-round neural distinguisher with the accuracy of 54.62%. For Speck64/128, our neural distinguisher obtained by training with the MRMSD input data format is valid, with the accuracy of 71.81%.

- **Applications to Simon**

Table 9: Summary of the neural distinguishers for Simon family

Ciphers	Data Format	Round	Input Difference	Accuracy	Source
Simon32/64	MOD	9	$(0x0, 0x80)$	82.27%	[10]
	MOD-CP	9	$(0x0, 0x80)$	99.34%	[11]
	MRMSP	9	$(0x0, 0x80)$	96.30%	Sect. 4.2
	MRMSD	9	$(0x0, 0x80)$	99.08%	Sect. 4.2
	MOD	10	$(0x0, 0x80)$	61.09%	[10]
	MOD-CP	10	$(0x0, 0x80)$	86.08%	[10]
	MRMSP	10	$(0x0, 0x80)$	78.72%	Sect. 4.2
	MRMSD	10	$(0x0, 0x80)$	83.02%	Sect. 4.2
	MOD-CP	11	$(0x20, 0x88)$	59.55%	[11]
	MRMSP	11	$(0x0, 0x80)$	56.16%	Sect. 4.2
	MRMSD	11	$(0x0, 0x80)$	60.81%	Sect. 4.2
Simon48/96	MOD	10	$(0x0, 0x100000)$	81.40%	[10]
	MRMSP	10	$(0x1000, 0x4400)$	82.13%	Sect. 4.2
	MRMSD	10	$(0x1000, 0x4400)$	99.55%	Sect. 4.2
	MOD	11	$(0x1000, 0x4400)$	61.43%	[10]
	MRMSP	11	$(0x1000, 0x4400)$	66.19%	Sect. 4.2
	MRMSD	11	$(0x1000, 0x4400)$	78.35%	Sect. 4.2
	MRMSD	12	$(0x1000, 0x4400)$	61.59%	Sect. 4.2
Simon64/128	MOD	11	$(0x0, 0x10)$	73.79%	[10]
	MOD-CP	11	$(0x0, 0x4)$	99.28%	[11]
	MRMSP	11	$(0x0, 0x10)$	95.01%	Sect. 4.2
	MRMSD	11	$(0x0, 0x10)$	99.95%	Sect. 4.2
	MOD	12	$(0x0, 0x10)$	69.57%	[10]
	MOD-CP	12	$(0x0, 0x4)$	83.78%	[11]
	MRMSP	12	$(0x0, 0x10)$	75.06%	Sect. 4.2
	MRMSD	12	$(0x0, 0x10)$	93.86%	Sect. 4.2
	MOD-CP	13	$(0x0, 0x40)$	60.32%	[11]
	MRMSD	13	$(0x0, 0x10)$	70.10%	Sect. 4.2

[1] We choose the highest accuracy NDs in these papers.

[2] MOD: Multiple Output Differences. MOD-CP: Multiple Output Differences and Ciphertext Pairs. MRMSP: Multiple round Multiple Splicing Pairs. MRMSD: Multiple round Multiple Splicing Differences.

The new training data format can also be applied to Simon effectively. Compared with previous works, our new distinguishers improve the number of rounds and accuracy. For Simon 48/96, we obtain (10-12)-rounds neural distinguishers with the accuracy of 99.55%, 78.35%, and 61.59%, respectively. For Simon64/128, our new neural distinguishers achieve an overall improvement, which covers (11-13)-rounds with the accuracy of 99.95%, 93.86%, and 70.10%, respectively. The results of Simon64/128 demonstrate the advantages of our data format: MRMSD has the advantage in a larger state size version of Simon. The full results are displayed in Table 9.

5 Conclusion

In this paper, we propose a new method of constructing training data for neural differential distinguishers from the perspective of traditional cryptanalysis, combined with the methods of constructing training data formats in the fields of deep learning. The new data format utilizes the output features of the penultimate round and performs a two-dimensional and non-realistic input data generation method. We verify the properties of the proposed new data format generation model through experiments and theoretical analysis step-by-step. Besides, by simultaneously considering multiple ciphertext pairs and multiple output differences, we propose two improved input data formats: MRMSP(Multiple Rounds Multiple Splicing Pairs) and MRMSD(Multiple Rounds Multiple Splicing Differences). We apply them to train the new neural distinguishers for NSA ciphers, Speck and Simon. As far as we know, our neural distinguishers achieve the longest rounds and the higher accuracy for Speck and Simon.

DATA AVAILABILITY

The data underlying this article are available in the article. Supplementary code and data for this paper is available at *https://github.com/CangXiXi/Improved-Neural-Distinguishers-with-Multi-Round-and-Multi-Splicing-Construct.*

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China [grant number 62206312].

References

[1] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. *Journal of Cryptology.*, 4(1):3–72, 1991.

[2] Xuejia Lai, James L Massey, and Sean Murphy. Markov ciphers and differential cryptanalysis. In *Proceedings of Advances in Cryptology - EUROCRYPT 1991, Brighton, UK, 8-11 April*, pages 17–38. Springer, Berlin, Heidelberg, 1991.

[3] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the EMNLP 2014, Doha, Qatar, 2529 October*, pages 1724–1734. Association for Computational Linguistics, Stroudsburg, 2014.

[4] Jing He, Ming Zhou, and Long Jiang. Generating chinese classical poems with statistical machine translation models. In *Proceedings of Twenty-Sixth AAAI Conference on Artificial Intelligence, Toronto, Canada, 2226 July*, pages 1650–1656. AAAI Press, Menlo Park, California, 2012.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, - CVPR 2016, Las Vegas, USA, 27-30 June*, pages 770–778. IEEE Computer Society, 2016.

[6] Aron Gohr. Improving attacks on round-reduced speck32/64 using deep learning. In *Proceedings of Advances in Cryptology - CRYPTO 2019, Santa Barbara, USA, 18-22 August*, pages 150–179. Springer-verlag, Cham, 2019.

[7] Zhenzhen Bao, Jian Guo, Meicheng Liu, Li Ma, and Yi Tu. Enhancing differential-neural cryptanalysis. *IACR Cryptol. ePrint Arch.*, 2021(719):1–65, 2021.

[8] Liu Zhang, Zilong Wang, and Boyang Wang. Improving differential-neural cryptanalysis with inception blocks. *IACR Cryptol. ePrint Arch.*, 2022(719):1–33, 2022.

[9] Yi Chen and Hongbo Yu. A new neural distinguisher considering features derived from multiple ciphertext pairs. *The Computer Journal*, 2022:1–15, 2022.

[10] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Improve neural distinguishers of SIMON and SPECK. *Security and Communication Networks*, 2021(9288229):1–11, 2021.

[11] Jinyu Lu, Guoqiang Liu, Yunwen Liu, Bing Sun, Chao Li, and Li Liu. Improved neural distinguishers with (related-key) differentials: Applications in SIMON and SIMECK. *ArXiv e-prints, 2201.03767*, 2022:1–20, 2022.

[12] Adrien Benamira, David Gérault, Thomas Peyrin, and Quan Quan Tan. A deeper look at machine learning-based cryptanalysis. In *Proceedings of Advances in Cryptology - EUROCRYPT 2021, Zagreb, Croatia, 17-21 October*, pages 805–835. Springer-verlag, Cham, 2021.

[13] Lucas Tabelini Torres, Thiago M Paixão, Rodrigo F Berriel, Alberto F De Souza, Claudine Badue, Nicu Sebe, and Thiago Oliveira-Santos. Effortless deep training for traffic sign detection using templates and arbitrary natural images. In *Proceedings of 2019 international joint conference on neural networks - IJCNN 2019, Budapest, Hungary, 14–19 July*, pages 1–7. IEEE, Springer-verlag, Cham, 2019.

[14] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK families of lightweight block ciphers. In *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference - DAC 2015, San Francisco, USA, 8-12 June*, pages 1–6. Association for Computing Machinery, New York, 2015.

[15] Julian Keilson. *Markov Chain Models–Rarity and Exponentiality.* Springer, New York, 1974.