A New Neural Distinguisher Considering Features Derived from Multiple Ciphertext Pairs

Yi Chen, Yantian Shen, Hongbo Yu, Sitong Yuan

Department of Computer Science and Technology, Tsinghua University, Haidian District, Beijing, 100084, P. R. China
Email: {chenyi19, shenyt18, yst20}@mails.tsinghua.edu.cn, yuhongbo@mail.tsinghua.edu.cn

Neural aided cryptanalysis is a challenging topic, in which the neural distinguisher ($\mathcal{ND}$) is a core module. In this paper, we propose a new $\mathcal{ND}$ considering multiple ciphertext pairs simultaneously. Besides, multiple ciphertext pairs are constructed from different keys. The motivation is that the distinguishing accuracy can be improved by exploiting features derived from multiple ciphertext pairs. To verify this motivation, we have applied this new $\mathcal{ND}$ to five different ciphers. Experiments show that taking multiple ciphertext pairs as input indeed brings accuracy improvement. Then, we prove that our new $\mathcal{ND}$ applies to two different neural aided key recovery attacks. Moreover, the accuracy improvement is helpful for reducing the data complexity of the neural aided statistic attack. The code is available at `https://github.com/AI-Lab-Y/ND_mc`.

Keywords: Neural Distinguisher; Cryptanalysis; Multiple Ciphertext Pairs; Data Reuse;

1. INTRODUCTION

In CRYPTO'19, Gohr improved attacks on round reduced Speck32/64 using deep learning [1], which created a precedent for neural aided cryptanalysis. The neural distinguisher ($\mathcal{ND}$) proposed by Gohr plays a core role in [1]. Its target is to distinguish real ciphertext pairs (C_0, C_1) corresponding to plaintext pairs with a specific difference from random ciphertext pairs. $\mathcal{ND}$ takes a ciphertext pair (C_0, C_1) as input, and gives the classification result.

The performance of $\mathcal{ND}$ is important for neural aided cryptanalysis. For Gohr's key recovery attack [1], the most important step is identifying the right plaintext structure that passes the differential placed before $\mathcal{ND}$. To attack 11-round Speck32/64, Gohr adopted a 6-round $\mathcal{ND}$ and 7-round $\mathcal{ND}$ for identifying the right plaintext structure. The identification result is given by the 6-round $\mathcal{ND}$ instead of 7-round $\mathcal{ND}$. Compared with the 7-round $\mathcal{ND}$, the 6-round $\mathcal{ND}$ achieves higher distinguishing accuracy. This implies that a stronger ND is more helpful for Gohr's attacks. Recently, Chen et al proposed a generic neural aided statistical attack (NASA) for cryptanalysis [2]. The data complexity of NASA is strongly related to the distinguishing accuracy of $\mathcal{ND}$.

To improve the performance of $\mathcal{ND}$, researchers have explored $\mathcal{ND}$ from different directions. The most popular direction is adopting different neural networks. In [3], Jain et al proposed a multi-layer perceptron network (MLP) to build $\mathcal{ND}s$ against PRESENT reduced to 3, 4 rounds. In [4], Yadav et al also built an MLP-based 3-round $\mathcal{ND}$ against Speck32/64. In [5], Bellini et al compared MLP-based and Convolutional Neural Network-based distinguishers with classic distinguishers. In [6], Pareek et al proposed fully-connected network-based distinguisher against the key scheduling algorithm of PRESENT. Another popular direction is changing the input of $\mathcal{ND}$. In [7], Baksi et al used the ciphertext difference $C_0 \oplus C_1$ as the input. In [2], Chen et al suggested that the ND can be built by flexibly taking some bits of a ciphertext pair as input. In [8], Hou et al investigate the influence of input difference pattern on the accuracy of $\mathcal{ND}s$ against round reduced Simon32/64.

These above $\mathcal{ND}s$ can be viewed as the same type since only features hidden in a single ciphertext pair are exploited. Thus, another natural way is taking more ciphertexts as the input. In [9], Benamira et al initially tested this idea as follows. First, a group of B ciphertexts is constructed from the same key. Second, take a group of B ciphertexts as the input of $\mathcal{ND}$. Finally, based on a large B, the accuracy of $\mathcal{ND}s$ against 5-round and 6-round Speck32/64 is increased to 100%, which is a huge improvement.

Previous findings especially the work in [9] inspired us to think about the deeper motivation of taking more ciphertexts as input. We believe that the deeper motivation stands for a generic method for improving $\mathcal{ND}$. The $\mathcal{ND}$ processing a group of B ciphertexts has two important characteristics: (1) the input contains more ciphertexts, (2) all the ciphertexts in a group share the same key. Since Ankele and Kölbl [10], as well as Gohr [1], reported significant key-dependency in the output distribution in round reduced Speck, we wonder whether the same key is a core factor that brings significant improvement.

1.1. Our Contributions

In this paper, our work contains five contributions:

- By introducing a clear deep motivation, we propose a new $\mathcal{ND}$ considering multiple ciphertext pairs simultaneously. The motivation is as follows. When ciphertext pairs corresponding to plaintext pairs with a specific difference obey a non-uniform distribution, there are some derived features from multiple ciphertext pairs. Once neural networks capture these features, $\mathcal{ND}$ would obtain performance improvement.
- We prove that the same key is not the core factor that brings significant improvement in [9]. We made the conclusion by testing the accuracy of $\mathcal{ND}s$ against round reduced Speck32/64 under two different scenarios: one is that ciphertext pairs in a group share the same key, one is that ciphertext pairs in a group adopt different keys. In the first scenario, the key for generating a ciphertext group each time is randomly selected. Experiments show that the same key has small or no influence on $\mathcal{ND}s$.
- We design a verification framework for further directly checking that derived features from multiple ciphertext pairs are learned. This framework is composed of two tests: false-negative test (FNT), false-positive test (FPT).
- We build two types of $\mathcal{ND}s$ for five round reduced ciphers: Speck32/64, Chaskey, PRESENT, DES, SHA3-256. The first one is the $\mathcal{ND}$ proposed by Gohr, and the other one is our new $\mathcal{ND}$. These experiments further prove the advantage of taking multiple ciphertext pairs as input and support the presented deep motivation.
- We prove that the $\mathcal{ND}$ taking multiple ciphertext pairs as input applies to key recovery, which is not discussed in previous research. At the time of writing, there are only two key recovery attacks [1, 2] based on the $\mathcal{ND}$ proposed by Gohr. We show how to apply new $\mathcal{ND}$ to these two attacks. Due to the performance improvement, the data complexity of the attack [2] can be reduced by using the new $\mathcal{ND}$.

1.2. Outlines

This paper is organized as follows:

- Section 2 presents preliminaries, including some important notations and five related ciphers.
- In section 3, the $\mathcal{ND}$ proposed by Gohr and two key recovery attacks are briefly reviewed.
- Section 4 presents the new $\mathcal{ND}$ including the motivation, model, the neural network for implementing the new $\mathcal{ND}$, and the training pipeline.
- Section 5 presents the verification framework.
- In section 6, we build $\mathcal{ND}s$ for five ciphers and perform an analysis.
- In section 7, we show how to perform key recovery attacks using the new $\mathcal{ND}$. A data reuse strategy is also proposed in this section.

2. PRELIMINARIES

2.1. Notations

P, C	Plaintext, Ciphertext
α	Plaintext difference
N, M	The number of plaintext or ciphertext pairs
$\mathcal{ND}_{k=?}$	$\mathcal{ND}$ with k ciphertext pairs as input
Z	The output of an $\mathcal{ND}$
r	The number of reduced rounds

2.2. Five Ciphers

We choose five different ciphers for supporting our work.

- Speck32/64 [11] is a lightweight block cipher whose block size is 32 bits. Its non-linear component is the modular addition.
- Chaskey [12] is a Message Authentication Code (MAC) algorithm whose intermediate state size is 128 bits. Its non-linear component is the modular addition.
- Present64/80 [13] is a block cipher whose block size is 64 bits. Its non-linear component is a 4×4 Sbox.
- DES [14] is a block cipher whose block size is 64 bits. Its non-linear component is given by eight different 6×4 Sboxes.
- SHA3-256 [15] is a hash function whose intermediate state size is 1600 bits. Its non-linear component can be seen as the application of a 5-bit Sbox applied in parallel 320 times.

We refer readers to [11, 12, 13, 14, 15] for more details of these ciphers.

2.3. Computing Resources

In this paper, the available computing resources are: an Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz, a graphics card (NVIDIA GeForce GTX 1060 6GB).

3. RELATED WORK

3.1. Gohr's Neural Distinguisher

In [1], Gohr built $\mathcal{ND}s$ against round reduced Speck32/64. The $\mathcal{ND}$ proposed by Gohr is a generic distinguisher since it only requires a plaintext difference constraint.

Consider a cipher E and a plaintext difference α. Gohr's $\mathcal{ND}$ aims at distinguishing two classes of ciphertext pairs

$$Y(C_0, C_1) = \left\{ \begin{array}{l} 1, if\ \ P_0 \oplus P_1 = \alpha \\ 0, if\ \ P_0 \oplus P_1 \neq \alpha \end{array} \right. \tag{1}$$

where (C_0, C_1) is the ciphertext pair corresponding to the plaintext pair (P_0, P_1), and Y is the label of (C_0, C_1).

We denote ciphertext pairs corresponding to plaintext pairs with the target difference α as **positive samples**, and denote ciphertext pairs corresponding to plaintext pairs with a random difference as **negative samples**.

If a neural network achieves a distinguishing accuracy higher than 0.5 over randomly selected ciphertext pairs, the neural network is a valid $\mathcal{ND}$.

In [1], Gohr chose a residual network [16] with one output neuron. Thus, the output Z of Gohr's $\mathcal{ND}$ is also used as the following posterior probability

$$\begin{gathered} Pr\left(Y = 1 \left| (C_0, C_1) \right.\right) = F_1\left(f\left(C_0, C_1\right)\right) \\ 0 \leqslant Pr\left(Y = 1 \left| (C_0, C_1) \right.\right) \leqslant 1 \end{gathered} \tag{2}$$

where $f(C_0, C_1)$ stands for features learned by the ND from (C_0, C_1), $F_1(\cdot)$ is the posterior probability estimation function learned by the $\mathcal{ND}$. If $Pr(Y = 1|(C_0, C_1)) > 0.5$, the label of (C_0, C_1) predicted by the $\mathcal{ND}$ is 1.

3.2. Gohr's Key Recovery Attack

Given an $\mathcal{ND}$, we denote the output of $\mathcal{ND}$ as Z. Positive samples are expected to obtain a higher posterior probability than negative samples, which is the core idea of Gohr's key recovery attack [1].

Consider an $(r+1)$-round cipher E and an r-round $\mathcal{ND}$ built over a plaintext difference α. Gohr's attack recovers the subkey of the $(r+1)-th$ round as follows:

1. Generate m positive samples with α randomly.
2. For each possible subkey guess kg:
   (a) Decrypt m positive samples with kg.
   (b) Feed partially decrypted samples into the $\mathcal{ND}$ and collect the outputs $Z_i, i \in [1, m]$.
   (c) Compute the rank score V_{kg} of kg as:

$$V_{kg} = \sum_{i=1}^{m} \log_2 \left(\frac{Z_i}{1 - Z_i} \right) \tag{3}$$

   (d) If V_{kg} exceeds a threshold c_1, save kg as a subkey candidate.
3. Return kg with the highest key rank score as the final subkey guess.

The value of c_1 and m is set experimentally.

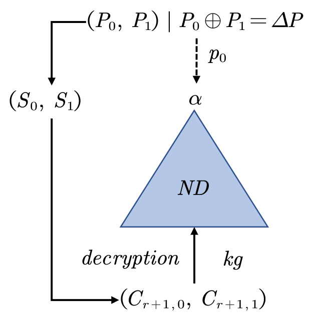
$(P_0,\ P_1)\ |\ P_0 \oplus P_1 = \Delta P$
p_0
$(S_0,\ S_1)$
α
ND
decryption
kg
$(C_{r+1,0},\ C_{r+1,1})$

FIGURE 1. The key recovery process. The prepended differential $\Delta P \rightarrow \alpha$ is satisfied with a probability p_0. The intermediate state pair is (S_0, S_1).

A differential $\Delta P \rightarrow \alpha$ can be placed before the $\mathcal{ND}$ to extend the rounds covered by the attack (see Fig.1). With the help of neutral bits [17], ciphertext structures consisting of m positive samples or negative samples can be generated. Then a high rank score occurs only when the structure consisting of positive samples is decrypted by the true subkey. More details can refer to [1].

3.3. Neural Aided Statistical Attack

The neural aided statistical attack proposed by Chen et al [2] is performed as follows:

1. Randomly generate N plaintext pairs with a difference ΔP.
2. Collect the ciphertext pairs.
3. For each possible subkey guess kg:
   (a) Decrypt N ciphertext pairs with kg.
   (b) Feed partially decrypted ciphertext pairs into the ND and collect the outputs $Z_i, i \in [1, N]$.
   (c) Count the following statistic T:

$$T = \sum_{i=1}^{N} \phi\left(Z_i\right), \quad \phi\left(Z_i\right) = \left\{ \begin{array}{l} 1, if\ \ Z_i > c_2 \\ 0, if\ \ Z_i \leqslant c_2 \end{array} \right. \tag{4}$$

   (d) If T exceeds a decision threshold t, save kg as a subkey candidate.
4. Return all the surviving subkey candidates.

Chen et al proposed a theoretical framework to estimate N and t. The value of c_2 is set in advance, which doesn't influence the estimation of N, t.

According to Fig.1, Chen et al summarized three types of probabilities:

$$Pr(Z > c_2 | S_0 \oplus S_1 = \alpha, kg = sk) = p_1 \tag{5}$$

$$Pr(Z > c_2 | S_0 \oplus S_1 = \alpha, kg \neq sk) = p_2 \tag{6}$$

$$Pr(Z > c_2 | S_0 \oplus S_1 \neq \alpha) = p_3 \tag{7}$$

where sk is the true subkey. These three probabilities p_1, p_2, p_3 are related to the $\mathcal{ND}$.

NASA returns all the possible subkey candidates. Besides, NASA allows us to set two ratios β_0, β_1 in advance. The ratio β_0 is the expected probability that the true subkey sk survives the attack. The ratio β_1 is the expected probability that wrong subkey guesses survive the attack.

Based on $p_0, p_1, p_2, p_3, \beta_0, \beta_1$, the required N is:

$$\sqrt{N} = \frac{z_{1-\beta_0} \times v_0 + z_{1-\beta_1} \times v_1}{(p_1 - p_2) \times p_0} \tag{8}$$

where

$$v_0 = \sqrt{p_0 \times p_1(1-p_1) + (1-p_0)p_3(1-p_3)}$$

$$v_1 = \sqrt{p_0 \times p_2(1-p_2) + (1-p_0)p_3(1-p_3)},$$

and $z_{1-\beta_0}$, $z_{1-\beta_1}$ are the quantiles of the standard normal distribution.

The decision threshold t is:

$$t = \mu_0 - z_{1-\beta_0} \times \sigma_0 \tag{9}$$

where

$$\mu_0 = N \times (p_0 p_1 + (1-p_0)\, p_3)$$

$$\sigma_0 = \sqrt{N \times p_0 \times p_1(1-p_1) + N(1-p_0)p_3(1-p_3)}$$

If $c_2 = 0.5$, the distinguishing accuracy of the ND is $(p_1 + 1 - p_3) \times 0.5$. Thus the data complexity of NASA is strongly related to the $\mathcal{ND}$. We refer readers to [2] for more details of NASA.

4. NEW NEURAL DISTINGUISHER

4.1. Motivations

The motivations of our new neural distinguisher contain two aspects.

First, in the machine learning community, providing more features is a common method to improve the accuracy of neural networks. For example, depth map estimation [18] and action recognition [19] are both tackled by feeding various features (eg. stereo knowledge [20], depth maps [21]) into neural networks simultaneously.

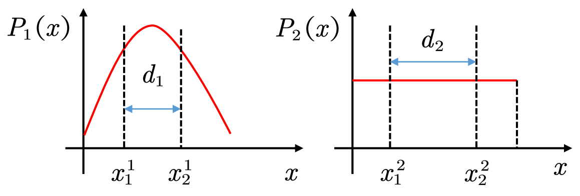

FIGURE 2. $P_1(x)$: a Gaussian distribution. $P_2(x)$: a uniform distribution.

Second, there are some useful features among multiple samples drawn from the same non-uniform distribution. Fig.2 shows a simple example. If we randomly draw two samples $(x_1^1, x_2^1)/(x_1^2, x_2^2)$ from a Gaussian distribution or a uniform distribution, the average distance of two samples is d_1/d_2. Then it is expected that $d_1 < d_2$, which is useful for distinguishing the two distributions.

Based on the two common phenomena, we obtain the idea of building a new neural distinguisher by considering multiple ciphertext pairs.

4.2. New Distinguisher Model

Our new $\mathcal{ND}$ needs to distinguish two types of ciphertext groups $(C_{1,1}, C_{1,2}, \cdots, C_{k,1}, C_{k,2})$:

$$Y = \begin{cases} 1, if\ P_{j,1} \oplus P_{j,2} = \alpha, j \in [1,k] \\ 0, if\ P_{j,1} \oplus P_{j,2} \neq \alpha, j \in [1,k] \end{cases} \tag{10}$$

where Y is the label of ciphertext groups, and $(C_{j,1}, C_{j,2})$ is the ciphertext pair corresponding to the plaintext pair $(P_{j,1}, P_{j,2}), j \in [1,k]$.

According to the introduced motivation, the requirement is that ciphertext pairs in a group are randomly sampled from the same distribution. To minimize influencing factors, we ask that a ciphertext group is constructed from k random keys if the cipher needs a key. This ensures that k ciphertext pairs do not have any same properties except for the same plaintext difference constraint.

Our new $\mathcal{ND}$ can be described as

$$\begin{gathered} Pr\left(Y=1\left|X_1, \cdots, X_k\right.\right) = \\ F_2\left(f\left(X_1\right), \cdots, f\left(X_k\right), \varphi\left(f\left(X_1\right), \cdots, f\left(X_k\right)\right)\right) \\ X_i = \left(C_{i,1}, C_{i,2}\right), i \in [1,k] \end{gathered} \tag{11}$$

where $f\left(X_i\right)$ represents the basic features extracted from the 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.

The motivation also puts forward some design guidelines for the neural network to be used. Since we hope more features $\varphi(f(X_1), \cdots, f(X_k))$ are extracted from the distribution of basic features $f(X_i), i \in [1,k]$, $\mathcal{ND}$ should learn basic features from each ciphertext pair firstly. From the perspective of neural networks, this requirement can be satisfied by placing one-dimensional convolutional layers before two-dimensional convolutional layers.

4.3. Residual Network

4.3.1. Network Architecture

The network architecture adopted by Gohr [1] is also applied in this article. According to the requirement of the motivation, except for the first one-dimensional convolutional layer, the remaining one-dimensional convolutional layers are replaced by two-dimensional convolutional layers.

Figure 3 shows the neural network architecture. The input consisting of k ciphertext pairs is arranged in a

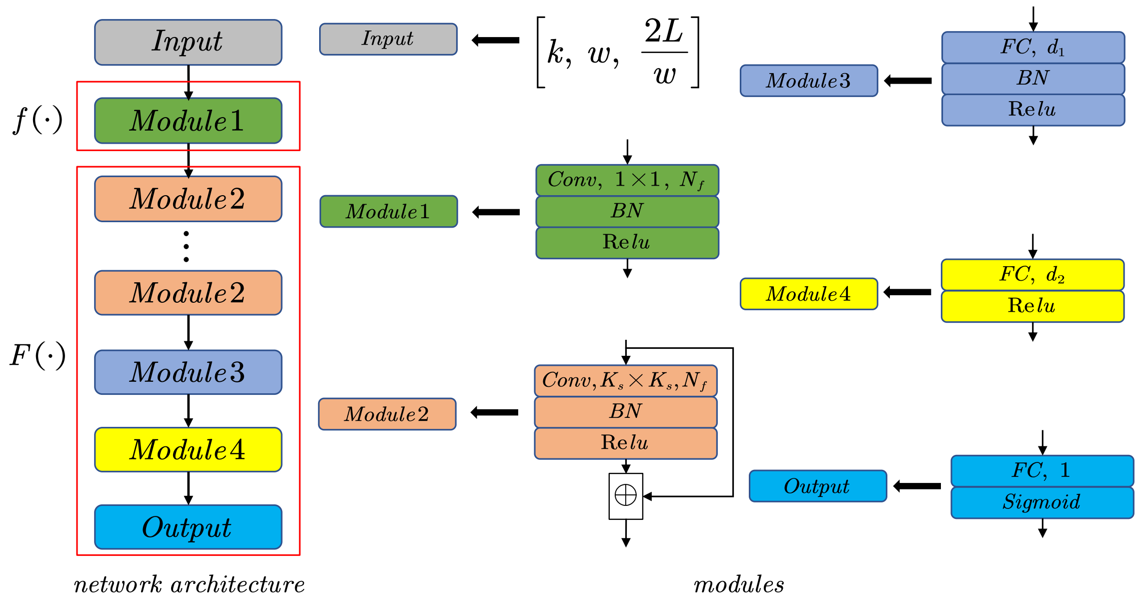

FIGURE 3. The network architecture of our new ND. *Conv* stands for a convolution layer with N_f filters. The size of each filter is $K_s \times K_s$. Module 2 also adopts the skip connection [16]. *FC* is a fully connected layer that has d_1 or d_2 neurons. *BN* is batch normalization. *Relu* and *Sigmoid* are two different activation functions. The output of *Sigmoid* ranges from 0 to 1.

$k \times w \times \frac{2L}{w}$ 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.

The network architecture contains two core modules. The first one (Module 1) is a bit slice layer that contains convolution kernels with a size of 1×1. This layer can learn basic features from each input ciphertext pair that is arranged in a $1 \times w \times \frac{2L}{w}$ array. The second one (Module 2) is a residual block that is built over a two-dimensional convolutional layer. The two-dimensional filters with a size of $K_s \times K_s$ can learn derived features from k ciphertext pairs. In this article, we use one residual block for building our new $\mathcal{ND}s$.

4.3.2. Training Pipeline

New $\mathcal{ND}s$ are obtained by following three processes:

1. **Data Generation:** Consider a plaintext difference ΔP and a cipher E. Randomly generate k plaintext pairs with ΔP. If E needs a key, randomly generate k keys. Collect the k ciphertext pairs with E and k keys. Regard these k ciphertext pairs as a ciphertext group with a size of k, and the label is $Y = 1$. We denote a ciphertext group with $Y = 1$ as a positive sample. If the plaintext differences of k plaintext pairs are random, the label of the resulting ciphertext group is $Y = 0$. And we denote it 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. We need to generate a training set and a testing set.
2. **Training:** Train the neural network (Figure 3) on the training dataset.
3. **Testing:** Test the distinguishing accuracy of the trained neural network on the testing dataset. If the test accuracy exceeds 0.5, return the neural network as a valid $\mathcal{ND}$. Or choose a different α and start from the data generation process again.

In the training phase, the neural network is trained for E_s epochs with a batch size of B_s. The cyclic learning rate scheme in [1] is adopted. Optimization is performed against the following loss function:

$$loss = \sum_{i=1}^{\frac{N}{k}} \left(Z_{i,p} - Y_i\right)^2 + \lambda \times \|W\| \tag{12}$$

where $Z_{i,p}$ is the output of the $\mathcal{ND}$, Y_i is the true label, W is the parameters of the neural network, and λ is the penalty factor. The Adam algorithm [22] with default parameters in Keras [23] is applied to the optimization.

5. THE VERIFICATION FRAMEWORK

Although the distinguishing accuracy of new $\mathcal{ND}_k$ is the best evidence for supporting the motivation of taking k ciphertext pairs as input, we propose an auxiliary verification framework to further show that new $\mathcal{ND}_k$ captures features derived from multiple ciphertext pairs. This framework is composed of two tests: False Negative Test (FNT), False Positive Test (FPT).

The idea of FPT and FNT is as follows. When features $f(X_i), i \in [1, k]$ hidden in a single ciphertext pair do not lead to the right classification, only derived features $\varphi\left(f\left(X_1\right), \cdots, f\left(X_k\right)\right)$ can provide useful clues for classification.

It is hard to directly select k ciphertext pairs that satisfy the above requirement based on the $\mathcal{ND}_k$ itself. Thus, An $\mathcal{ND}$ that takes a single ciphertext pair as input is used to select k wrongly classified ciphertext pairs. This is an approximate but reasonable method that is based on the following reasons

- When we build $\mathcal{ND}_k$, all the ciphertext pairs are constructed from different keys. This ensures that only two types of features are available: one is features hidden in a single ciphertext pair, the other one is features derived from multiple ciphertext pairs.
- When the $\mathcal{ND}$ that takes one ciphertext pair as input has high accuracy, it means that features hidden in the ciphertext pair provide strong clues leading to wrong classifications. If new $\mathcal{ND}_k$ still correctly classifies such k ciphertext pairs with a high probability, we can believe that this is due to features derived from multiple ciphertext pairs.

5.1. False Negative Test (FNT)

If k ciphertext pairs with label 1 are all wrongly classified by the $\mathcal{ND}$ that takes a single ciphertext pair as input

$$\begin{aligned} p\left(Y=1\left|X_1\right.\right) &= F_1\left(f\left(X_1\right)\right) < 0.5 \\ &\vdots \\ p\left(Y=1\left|X_k\right.\right) &= F_1\left(f\left(X_k\right)\right) < 0.5, \end{aligned} \tag{13}$$

such ciphertext pairs are false negative samples. These k samples are combined into a ciphertext group and fed into $\mathcal{ND}_k$.

Generate a large number of such ciphertext groups and feed them to $\mathcal{ND}_k$. What we care about is the following pass ratio

$$F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) \geqslant 0.5 \tag{14}$$

Now, the classification is determined by $\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)$. The final pass ratio under such a setting can show whether derived features have been learned and their effects. If $\mathcal{ND}_k$ 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 pair.

5.2. False Positive Test (FPT)

Similarly, if k ciphertext pairs with label 0 are wrongly classified

$$\begin{aligned} p\left(Y=1\left|X_1\right.\right) &= F_1\left(f\left(X_1\right)\right) \geqslant 0.5 \\ &\vdots \\ p\left(Y=1\left|X_k\right.\right) &= F_1\left(f\left(X_k\right)\right) \geqslant 0.5, \end{aligned} \tag{15}$$

such ciphertext pairs are false positive samples. These k samples are combined into a ciphertext group and fed into $\mathcal{ND}_k$. Now what we care about is the following pass ratio

$$F_2\left(f\left(X_1\right),\cdots,f\left(X_k\right),\varphi\left(f\left(X_1\right),\cdots,f\left(X_k\right)\right)\right) < 0.5 \tag{16}$$

TABLE 1. Parameters for constructing our new $\mathcal{ND}$

N_f	d_1	d_2	K_s	B_s
32	64	64	3	1000
λ	L_r	E_s	N	M
10^{-5}	$0.002 \to 0.0001$	10	10^7	10^6

TABLE 2. Distinguishing accuracy of $\mathcal{ND}_s$ against Speck32/64.

r	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
5	0.926	0.9739	0.9914	0.9991	0.9999
6	0.784	0.8667	0.9358	0.9528	0.9786
7	0.607	0.6396	0.6847	0.7009	0.6493

6. APPLICATIONS TO FIVE CIPHERS

We apply our new $\mathcal{ND}$ as well as Gohr's $\mathcal{ND}$ to five ciphers introduced in section 2.2. The training pipeline of Gohr's $\mathcal{ND}$ is presented in [1]. Table 1 summarizes the parameters that are related to the residual network and training pipeline that are introduced in section 4.3.

Since Gohr provided $\mathcal{ND}s$ against round reduced Speck32/64 in CRYPTO'19, we perform in-depth analysis by taking the application to Speck32/64 as an example. Applications to the remaining four ciphers are listed as supporting materials. For convenience, we denote Gohr's $\mathcal{ND}$ as $\mathcal{ND}_{k=1}$.

6.1. Experiments on Speck32/64

6.1.1. Neural Distinguishers

The plaintext difference is $\alpha = (0x0040, 0)$ introduced in [24]. We built $\mathcal{ND}_k, k \in \{2,4,8,16\}$ against Speck32/64 reduced to 5, 6, and 7 rounds respectively.

Table 2 lists the accuracy of $\mathcal{ND}s$. Compared with $\mathcal{ND}_{k=1}$, all the $\mathcal{ND}_k, k > 1$ achieve accuracy improvement. Besides, we find that the overfitting phenomenon [25] always appears in the training process of $\mathcal{ND}_{k=16}$ against 7-round Speck32/64. If this problem could be solved, it is possible to further improve the accuracy.

In the above setting, our distinguishers take k ciphertext pairs as input while Gohr's distinguishers take one ciphertext pair as input. To prove the positive influence of features derived from multiple ciphertext pairs, we compare the distinguishing accuracy under a fair setting.

The concrete process is as follows:

1. Generate n ciphertext pairs with the same sample

TABLE 3. Distinguishing accuracy of $\mathcal{ND}_s$ against Speck32/64 under the fair setting.

r / n	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}, m=\frac{n}{k}$	$\mathcal{ND}_{k=4}, m=\frac{n}{k}$
6 / 8	0.9573	0.9767	0.9823
7 / 8	0.7333	0.7421	0.7506
7 / 16	0.7859	0.8020	0.8090
7 / 32	0.8352	0.8682	0.8751
7 / 64	0.8757	0.9282	0.9387

label. It is worth noticing that n random keys are used.

2. For Gohr's distinguishers $\mathcal{ND}_{k=1}$, feed n ciphertext pairs into $\mathcal{ND}_{k=1}$, and use the median value of n outputs to give the prediction label of the n ciphertext pairs.
3. For our new distinguishers $\mathcal{ND}_{k>1}$, collect m ciphertext groups by uniformly sampling from the n ciphertext pairs, feed m ciphertext groups into $\mathcal{ND}_{k>1}$, and use the median value of n outputs to give the prediction label.
4. Repeat the above steps 10^6 times and count the distinguishing accuracy.

Such a setting ensures that our distinguishers do not use more prior knowledge. Taking $\mathcal{ND}_{k=2}, \mathcal{ND}_{k=4}$ as examples, we have performed several experiments under the setting.

Table 3 summarizes our experiment results. Under the fair setting, our distinguishers achieve higher accuracy. This proves that some features derived from multiple ciphertext pairs have been captured by our distinguishers, and these features bring accuracy improvement.

Besides, we find that the distinguishing accuracy can be further improved, if we increase m by adopting the data reuse strategy that will be introduced in Section 7.1.

6.1.2. The Impact of the Same Key Setting

As introduced in section 1, Benamira et al also tested the idea of taking multiple ciphertext pairs as input [9]. The difference with our $\mathcal{ND}s$ is that ciphertext pairs belonging to a group are constructed from the same key in [9].

To prove that the same key setting is not the core factor that brings huge accuracy improvement in [9], we build $\mathcal{ND}s$ by adopting the same key setting as follows

- we randomly generate a key for each ciphertext group.
- k ciphertext pairs belonging to a group are constructed from the same key.

Then we test the distinguishing accuracy of these $\mathcal{ND}s$ over two kinds of testing sets

- **testing set 1:** k ciphertext pairs of a group are constructed from k different keys.
- **testing set 2:** k ciphertext pairs of a group are constructed from the same key.

TABLE 4. Distinguishing accuracy of $\mathcal{ND}s$ over two kinds of testing sets. These $\mathcal{ND}s$ are built under the same key setting.

r	**testing set 1**			
	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
5	0.9744	0.9906	0.9989	0.9999
6	0.8663	0.9317	0.9561	0.9762
r	**testing set 2**			
	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
5	0.9745	0.9903	0.9990	0.9999
6	0.8662	0.9309	0.9557	0.9770

Table 4 summarizes the accuracy of $\mathcal{ND}s$ over two kinds of testing sets. Based on the comparision with results as shown in Table 2, we find that the same key setting has small or no influence on the accuracy.

TABLE 5. The comparison of neural network parameters as well as the accuracy of $\mathcal{ND}_k, k \in \{1, 2\}$.

r	$\mathcal{ND}_{k=1}$, 1 residual blocks	
	parameter	accuracy
5	44321	0.926
6	44321	0.784
r	$\mathcal{ND}_{k=1}$, 10 residual blocks	
	parameter	accuracy
5	102497	0.929
6	102497	0.788
r	$\mathcal{ND}_{k=2}$	
	parameter	accuracy
5	89377	0.9738
6	89377	0.8613

6.1.3. The comparison of Neural Network parameters

Since the neural network adopted in this paper is different from the neural network adopted by Gohr in [1], we also focus on the comparison of neural network parameters.

Table 5 summarizes the comparison of neural network parameters as well as the accuracy of some $\mathcal{ND}s$. Gohr reported the best accuracy of 5-round and 6-round $\mathcal{ND}_{k=1}$ by using 10 residual blocks. Besides, Gohr also provided $\mathcal{ND}_{k=1}$ by using 1 residual block. These two kinds of distinguishers almost achieve the same accuracy. Compared with $\mathcal{ND}_{k=1}$ with 10 residual blocks, our new distinguishers $\mathcal{ND}_{k=2}$ achieve significant accuracy improvement but contains fewer parameters. This comparison proves that taking more ciphertext pairs as input is the reason that brings accuracy improvement.

TABLE 6. Pass ratios of **FPT** and **FNT** of $\mathcal{ND}_k$ against Speck32/64.

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

TABLE 7. Distinguishing accuracy of $\mathcal{ND}s$ against Chaskey

r	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
3	0.8608	0.8958	0.9583	0.9887	0.9986
4	0.6161	0.6589	0.6981	0.7603	0.7712

6.1.4. The Results of FPT and FNT

We further perform the **FPT** and **FNT**. Corresponding pass ratios are presented in Table 6. For each $\mathcal{ND}_k$, there is at least one type of pass ratio higher than 0. This further proves that $\mathcal{ND}_k$ captures derived features from k ciphertext pairs.

6.2. Experiments on Chaskey

Based on the plaintext difference $\alpha = (0x8400, 0x0400, 0, 0)$ [12], we build $\mathcal{ND}s$ against Chaskey reduced to 3, 4 rounds. The accuracies are presented in Table 7. Table 8 summarizes the results of the FPT and FNT.

6.3. Experiments on Present64/80

Based on the plaintext difference $\alpha = (0, 0, 0, 0x9)$ provided in [26], we build $\mathcal{ND}s$ against Present64/80 reduced up to 6, 7 rounds respectively. The penalty factor is 10^{-4} and other related parameters are the same as Table 1. The distinguishing accuracies are presented in Table 9. Table 10 summarizes the results of FPT and FNT.

TABLE 8. Pass ratios of FPT and FNT of $\mathcal{ND}s$ against Chaskey.

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

TABLE 9. Distinguishing accuracy of $\mathcal{ND}s$ against Present64/80

r	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
6	0.6584	0.7198	0.7953	0.8308	0.8259
7	0.5486	0.5503	0.5853	0.5786	0.5818

TABLE 10. Pass ratios of FNT and FPT of $\mathcal{ND}s$ against Present64/80.

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

6.4. Experiments on DES

Based on the analysis of DES in [27], the plaintext difference $\alpha = (0x40080000, 0x04000000)$ is adopted. We build $\mathcal{ND}s$ against 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 1. The distinguishing accuracies are presented in Table 11. The pass ratios of the FPT and FNT of $\mathcal{ND}s$ are presented in Table 12.

6.5. Experiments on SHA3-256

SHA3-256 is a hash function. When one message block is fed into reduced SHA3-256, we collect the first 32 bytes of the output process after r-rounds permutation is applied to this message block. Given a message difference $\alpha = 1$, we build $\mathcal{ND}s$ against SHA3-256 reduced up to 3 rounds.

The number of ciphertext pairs is $N = 2 \times 10^6$. The batch size is 500, and the penalty factor is 10^{-5}. The accuracies are presented in Table 13. The pass ratios of the FPT and FNT of $\mathcal{ND}s$ are presented in Table 14.

TABLE 11. Distinguishing accuracy of $\mathcal{ND}s$ against DES.

r	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
5	0.6261	0.7209	0.8382	0.9318	0.9585
6	0.5493	0.5653	0.5568	0.5507	0.5532

TABLE 12. Pass ratios of FNT and FPT of $\mathcal{ND}s$ against DES.

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

TABLE 13. Distinguishing accuracy of $\mathcal{ND}s$ against SHA3-256.

r	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
3	0.7228	0.8149	0.9241	0.971	0.9904

7. KEY RECOVERY ATTACKS

In this section, we propose a data reuse strategy for reducing data complexity. Then we prove that our $\mathcal{ND}$ can be applied to the two key recovery attacks introduced in section 3. Since the data complexity of NASA is directly related to the performance of $\mathcal{ND}s$, NASA is first performed to highlight the extra superiority of our $\mathcal{ND}s$.

7.1. Data Reuse Strategy for Reducing Data Complexity

There is a potential problem when we directly apply our new $\mathcal{ND}$ to key recovery attacks.

Assuming Gohr's distinguisher and our new $\mathcal{ND}_k$ 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 $\mathcal{ND}_k$ is k times as much as the data complexity of Gohr's distinguisher.

Given M ciphertext pairs $X_i = (C_{i,0}, C_{i,1}), i \in [1, M]$, there are a total of $\binom{M}{k}$ options for composing a ciphertext group, which is much larger than $\frac{M}{k}$. Thus we can randomly select M ciphertext groups from $\binom{M}{k}$ options. Such a strategy can help reduce 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 do not obtain good results. The main reason is that the sampling randomness of M ciphertext groups is 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. We denote the reuse frequency of the i_{th} ciphertext pair as $RF_i, i \in [1, M]$. *Maximum Reuse Frequency* (MRF) is defined as the maximum value of RF_i:

$$MRF = \max RF_i, i \in [1, M] \tag{17}$$

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:

$$\begin{gathered} SSD = \max |G_i \bigcap G_j|, i, j \in [1, M] \\ G_i = \{X_{i1}, \cdots, X_{ik}\} \\ G_j = \{X_{j1}, \cdots, X_{jk}\} \\ i1, \cdots, ik, j1, \cdots, jk \in [1, M] \end{gathered} \tag{18}$$

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*** (see Algorithm 1) that can reduce data complexity and maintain sampling randomness.

Algorithm 1 Data Reuse Strategy

```
Require: $MRF$; $SSD$; $k$; $M$.
Ensure: $M$ ciphertext groups with a size of $k$.
1: Randomly select $k$ ciphertext pairs from $M$ ciphertext pairs to form a ciphertext group.
2: Repeat step 2 for $M$ times to obtain $M$ ciphertext groups.
3: Compute $MRF$ and $SSD$. If two values are both smaller than the threshold we set, return the $M$ ciphertext groups. Or start from step 1 again.
```

7.2. Application to NASA

When we replace Gohr's distinguisher with our new $\mathcal{ND}$, the process of NASA does not change. The only difference is the data collection.

7.2.1. Data Collection

Consider the attack process as shown in Fig.1. Assuming that our new $\mathcal{ND}$ is built with α. Now, we need to generate ciphertext groups.

Generate k plaintext pairs $(P_0^i, P_1^i), i \in [1, k]$ with the difference ΔP. Collect corresponding ciphertexts $(C_0^i, C_1^i), i \in [1, k]$. The intermediate states are $(S_0^i, S_1^i), i \in [1, k]$.

TABLE 14. Pass ratios of FNT and FPT of $\mathcal{ND}s$ against SHA3-256.

r	False Negative Test			
	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
3	0.2249	0.2347	0.3336	0.2711
r	False Positive Test			
	$\mathcal{ND}_{k=2}$	$\mathcal{ND}_{k=4}$	$\mathcal{ND}_{k=8}$	$\mathcal{ND}_{k=16}$
3	0.1045	0.0961	0.0171	0.0088

According to the introduction in section 4.2, these k ciphertext pairs should satisfy

$$S_0^i \oplus S_1^i = \alpha, \quad or \quad S_0^i \oplus S_1^i \neq \alpha, i \in [1,k]$$

simultaneously. We use neutral bits [17] to generate such k ciphertext pairs.

Here we briefly review the definition of neutral bits. Let E denote the encryption function. We focus on the following conforming pairs

$$P_0 \oplus P_1 = \Delta P, \quad E(P_0) \oplus E(P_1) = \alpha.$$

If the condition $E(P_0 \oplus e^j) \oplus E(P_1 \oplus e^j) = \alpha$ always holds where $e^j = 1 << j$, the j-th bit is a neutral bit.

Thus, we can generate $2^m \geqslant k$ ciphertext pairs using m neutral bits. The probability that these k ciphertext pairs satisfy the difference transition $\Delta P \to \alpha$ simultaneously is still p_0. Then N ciphertext groups with a size of k can be generated as

1. Randomly generate N plaintext pairs with ΔP.
2. Generate N plaintext structures using m neutral bits.
3. Randomly pick k plaintext pairs from a structure and collect the ciphertext pairs.

The total data complexity is $N \times k$.

It is worth noticing that the data reuse strategy is still applicable here. More precisely, the data collection is performed as

1. Randomly generate $\frac{N}{M}$ plaintext pairs with ΔP.
2. Generate $\frac{N}{M}$ plaintext structures using m neutral bits.
3. Randomly pick M plaintext pairs from a structure, and generate M ciphertext groups using the data reuse strategy (Algorithm 1).

The total data complexity is N now.

7.2.2. Experiments on Speck32/64

To prove that our $\mathcal{ND}$ applies to NASA, we perform experiments on Speck32/64.

Our new $\mathcal{ND}$ achieves higher accuracy than the $\mathcal{ND}$ proposed by Gohr. Since the data complexity of NASA is related to the accuracy of $\mathcal{ND}$, it is possible to reduce the data complexity of NASA by adopting our new $\mathcal{ND}$.

Experiment settings. We adopt a 2-round differential $\Delta P = 0x211/0xa04 \xrightarrow{p_0 = 2^{-6}} \alpha = 0x40/0x0$ as the prepended differential. Let $\beta_0 = 0.005, \beta_1 = 2^{-16}, c_2 = 0.5$. The meaning of these parameters is defined in section 3.3. Since c_2 is set, the values of p_1, p_3 are experimentally estimated based on $\mathcal{ND}s$.

The estimation of p_2 is complex. Let $p_{2|d}$ denote the estimated value of p_2 where d is the Hamming distance between the correct key tk and wrong keys kg. According to the introduction in [2], when d increases, $p_{2|d}$ will decrease. Moreover, when p_2 increases, the data complexity of NASA also increases. Thus, if we hope the Hamming distance between tk and surviving kg does not exceed d, the value of p_2 is

$$p_2 = \max\{p_{2|i} | i \in [d+1, 16]\}. \qquad (19)$$

In this paper, we choose two different settings: $d = 2$, $d = 1$.

Comparison of data complexity. Table 15 and Table 16 show the comparison of data complexity under two experiment settings respectively.

The second row corresponds to the data complexity when Gohr's $\mathcal{ND}$ is adopted. These results are also used as the baseline. When an r-round $\mathcal{ND}$ with a group size of k is adopted, the corresponding data complexity is displayed in bold if it is smaller than the baseline.

We test 12 new $\mathcal{ND}s$ in total. Table 15 and Table 16 show that the data complexity is reduced in most cases. There is only one case in which the data complexity is not reduced.

Analysis of the data complexity. There are two questions to be explained: (1) why does the accuracy improvement of $\mathcal{ND}s$ bring the reduction of the data complexity? (2) why does the data complexity is not reduced in the only failed case shown in Table 16?

To answer the **first** question, we need to analyze how the data complexity is influenced by p_1, p_3. Based on Equation 8 in section 3.3, we get two following conclusions:

TABLE 15. Data complexity comparisons when $p_0 = 2^{-6}, d = 2, \beta_0 = 0.005, \beta_1 = 2^{-16}, c_2 = 0.5$. The prepended differential is a 3-round differential that is extended from $0x211/0xa04 \xrightarrow{p_0} 0x40/0x0$ without loss of transition probability.

Distinguisher	Data Complexity ($log_2 N$)		
	$r = 5$	$r = 6$	$r = 7$
$\mathcal{ND}_{k=1}$	14.212	16.911	20.509
$\mathcal{ND}_{k=2}$	**13.165**	**15.821**	**19.761**
$\mathcal{ND}_{k=4}$	**12.901**	**14.764**	**18.886**
$\mathcal{ND}_{k=8}$	**12.19**	**14.681**	**18.744**
$\mathcal{ND}_{k=16}$	**13.107**	**14.720**	**20.215**

TABLE 16. Data complexity comparisons when $p_0 = 2^{-6}, \boldsymbol{d = 1}, \beta_0 = 0.005, \beta_1 = 2^{-16}, c_2 = 0.5$. The prepended differential is a 3-round differential that is extended from $0x211/0xa04 \xrightarrow{p_0} 0x40/0x0$ without loss of transition probability.

Distinguisher	Data Complexity ($log_2 N$)		
	$r = 5$	$r = 6$	$r = 7$
$\mathcal{ND}_{k=1}$	14.72	17.456	21.081
$\mathcal{ND}_{k=2}$	**13.979**	**16.484**	**20.349**
$\mathcal{ND}_{k=4}$	**14.187**	**15.606**	**19.49**
$\mathcal{ND}_{k=8}$	**14.085**	**15.750**	**19.38**
$\mathcal{ND}_{k=16}$	15.399	**16.361**	**20.949**

TABLE 17. The value of p_1, p_2, p_3 related to the 5-round $\mathcal{ND}s$ when $c_2 = 0.5, \boldsymbol{d = 1}, r = 5$. $p_0 = 2^{-6}$.

Distinguisher	p_1	p_2	p_3	$log_2 N$
$\mathcal{ND}_{k=1}$	0.8977	0.3335	0.0462	14.72
$\mathcal{ND}_{k=2}$	0.9665	0.4802	0.0185	**13.979**
$\mathcal{ND}_{k=4}$	0.9894	0.6927	0.0069	**14.187**
$\mathcal{ND}_{k=8}$	0.9988	0.8604	0.0007	**14.085**
$\mathcal{ND}_{k=16}$	0.9999	0.9672	1.92×10^{-5}	15.399

TABLE 18. The value of p_1, p_2, p_3 related to the 5-round $\mathcal{ND}s$ when $c_2 = 0.5, d = 1, r = 5$. $\boldsymbol{p_0 = 2^{-12}}$.

Distinguisher	p_1	p_2	p_3	$log_2 N$
$\mathcal{ND}_{k=1}$	0.8977	0.3335	0.0462	26.657
$\mathcal{ND}_{k=2}$	0.9665	0.4802	0.0185	**25.811**
$\mathcal{ND}_{k=4}$	0.9894	0.6927	0.0069	**25.834**
$\mathcal{ND}_{k=8}$	0.9988	0.8604	0.0007	**24.888**
$\mathcal{ND}_{k=16}$	0.9999	0.9672	1.92×10^{-5}	**24.021**

- when $p_1 | p_1 \geqslant 0.5$ increases, the data complexity N decreases.
- when $p_3 | p_3 \leqslant 0.5$ decreases, the data complexity N decreases.

During the training of $\mathcal{ND}s$, the accuracy can be formulated as

$$acc = 0.5 \times (TPR + TNR)$$

where TPR is the true positive rate and TNR is the true negative rate.

If we set $c_2 = 0.5$, the following conclusions hold

$$\begin{array}{c} TPR = p_1, \quad TNR = 1 - p_3 \\ acc = 0.5 \times (p_1 + 1 - p_3). \end{array} \tag{20}$$

Thus, when the accuracy acc of $\mathcal{ND}s$ increases, there are three phenomena: p_1 increases, or p_3 decreases, or the former two phenomena both occur.

No matter which phenomenon occurs, it is helpful for reducing the data complexity. This is why the data complexity is reduced in most cases shown in Table 15 and Table 16.

To answer the **second** question, we need to consider the impact of p_2. For convenience, we summarized the values of p_1, p_2, p_3 related to the 5-round $\mathcal{ND}s$ in Table 17.

The value of p_2 also increases as shown in Table 17. Chen et al presented that the impact of p_1, p_2 on N is $\mathcal{O}((p_1 - p_2)^{-2})$ [2]. Therefore, the increase of p_2 has a negative impact on the data complexity. If p_2 is very close to p_1, the positive impact of the accuracy improvement may be offset. This is why the data complexity is not reduced when the 5-round $\mathcal{ND}_{k=16}$ is adopted. Actually, when p_0 becomes smaller, the reduction of data complexity is more significant. Table 18 shows an example.

Practical experiments. Based on the attack settings shown in Table 15, we perform NASA against 10-round Speck32/64 based on the $\mathcal{ND}_{k=1}$ and $\mathcal{ND}_{k=2}$ ($r = 6$) respectively. The target is to recover sk_{10}. Since $d = 2$, the number of surviving subkey guesses should not exceed $137 \times (1 - \beta_0) + (2^{16} - 137) \times \beta_1 = 137.31$.

Since the data complexity presented in Table 15 is not low, the attack may take too much time. We adopt an optimization method proposed in [2] to accelerate this attack. This method is building a student distinguisher to reduce the key space to be searched. The student distinguisher is built over 14 ciphertext bits $\{30 \sim 23, 14 \sim 7\}$. Then in the first stage, we guess 8 subkey bits $sk_{10}[8 \sim 0]$. In the second stage, we guess the complete sk_{10} based on surviving guesses of $sk_{10}[8 \sim 0]$.

To filter $sk_{10}[8 \sim 0]$, the student distinguisher with $k = 1$ requires $2^{18.888}$ plaintext pairs. In the second stage, we select $N = 2^{16.911}$ plaintext pairs from $2^{18.888}$ plaintext pairs. When we perform NASA with Gohr's 6-round distinguishers 100 times, the results are

1. the true subkey sk_{10} survives in 97 trails.
2. the average numbers of surviving subkey guesses in two stages are $14.98, 15.16$ respectively.
3. in all the 100 trails, the number of surviving subkey guesses is lower than 137.31.

To filter $sk_{10}[8 \sim 0]$, the student distinguisher with $k = 2$ requires $2^{17.785}$ plaintext pairs. In the second stage, we select $N = 2^{15.821}$ plaintext pairs from $2^{17.785}$ plaintext pairs. When we perform NASA with our 6-round $\mathcal{ND}_{k=2}$ 100 times, the results are

1. the true subkey sk_{10} survives in 90 trails.
2. the average numbers of surviving subkey guesses in two stages are $11.82, 25.07$ respectively.
3. In all the 100 trails, the number of surviving subkey guesses is lower than 137.31.

Figure 4 shows the runtime comparison of the 200 experiments. The practical experiments further prove that our new $\mathcal{ND}s$ can be applied to NASA. Besides, with smaller data complexity, the NASA based on our $\mathcal{ND}$ achieves a competitive result.

7.3. Application to Gohr's Attack

Gohr's attack is not directly related to the distinguishing accuracy of $\mathcal{ND}s$. Thus, we mainly verify whether our new $\mathcal{ND}$ applies to Gohr's attack.

In [1], Gohr performed a key recovery attack on 11-round Speck32/64. In this section, we first perform the same attack using our new $\mathcal{ND}_{k=2}$. Then we present a deeper discussion.

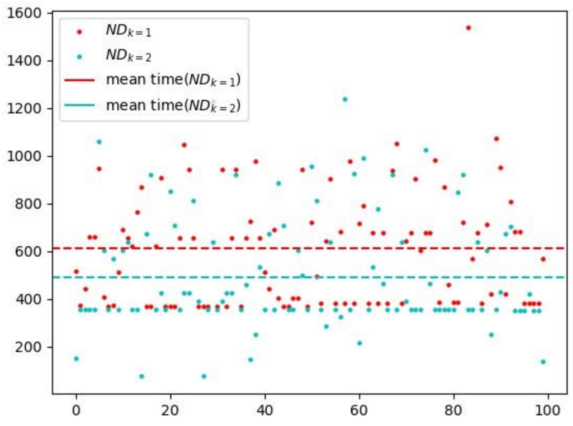

FIGURE 4. The runtime of 100 experiments. When Gohr's 6-round distinguisher $\mathcal{ND}_{k=1}$ is used, the average runtime of NASA is 612 seconds. When our 6-round distinguisher $\mathcal{ND}_{k=2}$ is used, the average runtime of NASA is 492 seconds.

7.3.1. Key Recovery Attack on 11-round Speck32/64

The target of this attack is to recover the last two subkeys (sk_{11}, sk_{10}). This attack returns a pair of subkey guesses (kg_{11}, kg_{10}). If $kg_{11} = sk_{11}$ and kg_{10} is different from sk_{10} at most 2 bits, this attack is viewed as a success [1].

Experiment settings. A 6-round and 7-round $\mathcal{ND}_{k=2}$ are built over $\alpha = (0x40, 0x0)$. A prepended 3-round differential is extended from a 2-round differential $\Delta P = (0x211, 0xa04) \xrightarrow{p_0 = 2^{-6}} \alpha = (0x40, 0x0)$. Six neutral bits $\{14, 15, 20, 21, 22, 23\}$ are used to generate plaintext structures consisting of 64 plaintext pairs. The data reuse strategy is also adopted by letting $MRF = 2$ and $SSD = 1$.

The whole attack is performed as

1. Randomly generate 100 plaintext pairs with a difference ΔP.
2. Generate 100 plaintext structures using 6 neutral bits above, and collect corresponding ciphertext structures.
3. For each ciphertext structure:
   (a) collect possible kg_{11} using the method introduced in section 3.2.
   (b) For each possible kg_{11}:
      i. Decrypt the current ciphertext structure with kg_{11}.
      ii. Collect possible subkey guess pairs (kg_{11}, kg_{10}) using the method introduced in section 3.2.
4. Return surviving (kg_{11}, kg_{10}) with the highest rank score as the final subkey guess.

In section 3.2, we have reviewed how Gohr's attack recovers the subkey sk_{r+1} with an r-round $\mathcal{ND}$. This method needs a rank score threshold. In steps 3a and 3(b)ii, we need a threshold c_3, c_4 respectively. In this paper, let $c_3 = 18$ and $c_4 = 150$.

TABLE 19. Success rates of performing 1000 experiments (Gohr's attack). Repeat 5 times. The first row represents the success rate when Gohr's distinguishers are used. The second row represents the success rate when our new distinguishers with $k = 2$ are used.

	1	2	3	4	5
$\mathcal{ND}_{k=1}$	0.533	0.52	0.501	0.557	0.523
$\mathcal{ND}_{k=2}$	0.536	0.534	0.512	0.552	0.529

Experiment results. Run 1000 experiments each time, and repeat 5 times. These experiments based on Gohr's distinguishers $\mathcal{ND}_{k=1}$ were also performed using the same ciphertexts. Table 19 summarizes the success rates.

7.3.2. Posterior Probability Analysis

We have proved that our $\mathcal{ND}$ applies to Gohr's attack. Moreover, the attack based on our $\mathcal{ND}s$ shows a minor advantage in terms of the success rate. This minor advantage is interesting since the success rate of Gohr's attack is not directly determined by the distinguishing accuracy. To better understand the influence of accuracy improvement on Gohr's attack, we perform a deeper analysis from the perspective of the key rank score.

Consider an $(r+1)$-round cipher E. We first build a r-round $\mathcal{ND}$ based on a difference α. Then we collect numerous ciphertext pairs corresponding to plaintext pairs with a difference α. We decrypt these ciphertext pairs with a subkey guess kg and feed the partially decrypted ciphertext pairs into the $\mathcal{ND}$.

Let tk denote the true subkey of the $(r+1)$-round. Besides, the Hamming distance between tk and kg is d. We focus on the expectation of the following conditional posterior probability

$$Z = Pr\left(Y = 1 \,|X, d\right) = F\left(X\right) \tag{21}$$

where X is the input of the $\mathcal{ND}$, and F is $\mathcal{ND}$. If the $\mathcal{ND}$ is Gohr's distinguisher, X is a decrypted ciphertext pair. If the $\mathcal{ND}$ is our distinguisher $\mathcal{ND}_k$, X is a ciphertext group consisting of k decrypted ciphertext pairs.

Taking $\mathcal{ND}_{k=2}$ against Speck32/64 reduced to 6, 7 rounds as examples, we estimate the expectations of the above conditional posterior probability. As a comparison, we also estimate the expectations based on $\mathcal{ND}_{k=1}$. The final estimation results are shown in Figure 5, Figure 6.

There are two important phenomena. First, compared with Gohr's distinguishers $\mathcal{ND}_{k=1}$, our distinguishers $\mathcal{ND}_{k=2}$ bring higher expectations $Pr(Y = 1|X, d = 0)$. Second, the value of $Pr(Y = 1|X, d = 0) - Pr(Y = 1|X, d = i), i \in [1, 16]$ increases.

The first phenomenon makes that a large key rank score threshold (eg. $c_3 = 18, c_4 = 150$) is applicable.

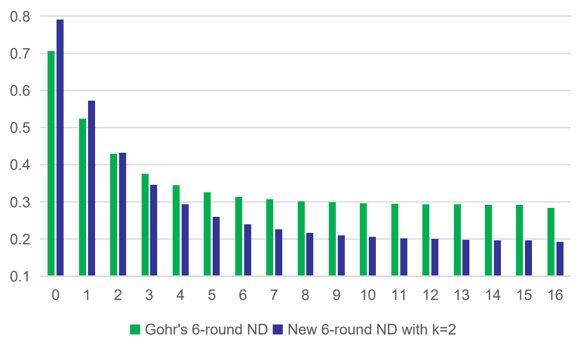

FIGURE 5. The expectations of the conditional posterior probability (Equation 21) of 6-round $\mathcal{ND}s$ against Speck32/64.

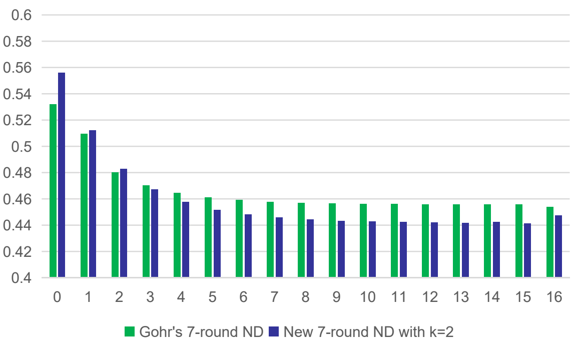

FIGURE 6. The expectations of the conditional posterior probability (Equation 21) of 7-round $\mathcal{ND}s$ against Speck32/64.

The second phenomenon makes the gap between the rank score of the true key and that of wrong keys increase. By setting a high key rank score threshold, wrong keys are less likely to obtain a key rank score higher than the threshold. Thus, a higher success rate is more likely to be obtained by replacing Gohr's NDs with our NDs.

8. OPEN PROBLEMS

Our work in this paper raises some open problems:

- What features derived from multiple ciphertext pairs are learned by our distinguishers?
- The influence of features derived from multiple ciphertext pairs is rather complex. More exactly, except for its positive influence, we find that these features also have a negative influence. For example, when we compare the distinguishing accuracy of $\mathcal{ND}s$ under a fair setting (see Section 6.1.1), if we give the prediction label based on the following metric:

$$v = log(\frac{Z_1}{1-Z_1}) + \cdots + log(\frac{Z_m}{1-Z_m}), \qquad (22)$$

  where $Z_i, i \in [1, m]$ is the output of $\mathcal{ND}s$, our distinguishers have tiny or no advantage in terms of the distinguishing accuracy. Table 20 shows our experiment results based on the above metric. Thus, an important problem is how to make full use of these features and bring more significant positive influence?

TABLE 20. Distinguishing accuracy of $\mathcal{ND}s$ against Speck32/64 under the fair setting. If $v > 0$ (see Formula 22), the prediction label is 1.

r / n	$\mathcal{ND}_{k=1}$	$\mathcal{ND}_{k=2}, m = \frac{n}{2}$	$\mathcal{ND}_{k=4}, m = \frac{n}{2}$
6 / 8	0.987	0.9853	0.9873
6 / 10	0.9947	0.9934	0.9942
7 / 8	0.778	0.7579	0.7636

These problems are out of scope of this paper. We will explore in future research.

9. CONCLUSIONS

In this paper, we focus on the neural distinguisher which is the core module in neural aided cryptanalysis. By considering multiple ciphertext pairs simultaneously, we propose a new neural distinguisher and have performed a deep exploration of it. Compared with the neural distinguisher considering a single ciphertext pair, this new neural distinguisher achieves higher distinguishing accuracy, which is verified by applications to five different ciphers. Moreover, we prove that the accuracy improvement results from features derived from multiple ciphertext pairs.

Our new neural distinguisher also applies to key recovery attacks. We show how to perform two different key recovery attacks based on our new neural distinguishers. The first one is the neural aided statistical attack. Due to the accuracy improvement, the data complexity of neural aided statistical attack is reduced by adopting our new neural distinguisher. A data reuse strategy is proposed to strengthen this advantage. The second one is the key recovery attack proposed by Gohr at CRYPTO'19. Our new neural distinguisher applies to this attack but does not bring a significant positive influence, since this attack is not related to the distinguishing accuracy.

Our new neural distinguisher is full of potential. In the future, as long as neural aided key recovery attacks are related to the performance of neural distinguishers, our new neural distinguisher could be a priority choice. Besides, our neural distinguisher also introduces a novel cryptanalysis direction by considering multiple ciphertext pairs simultaneously.

DATA AVAILABILITY

The data underlying this article are available at `https://github.com/AI-Lab-Y/ND_mc`.

ACKNOWLEDGEMENTS

This work is supported by the National Key Research and Development Program of China (2018YFB0803405, 2017YFA0303903).

REFERENCES

[1] Gohr, A. (2019) Improving attacks on round-reduced speck32/64 using deep learning. In Boldyreva, A. and Micciancio, D. (eds.), *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, Lecture Notes in Computer Science, **11693**, pp. 150–179. Springer.

[2] Chen, Y. and Yu, H. (2020) Neural aided statistical attack for cryptanalysis. *IACR Cryptol. ePrint Arch.*, **2020**, 1620.

[3] Jain, A., Kohli, V., and Mishra, G. (2020) Deep learning based differential distinguisher for lightweight cipher PRESENT. *IACR Cryptol. ePrint Arch.*, **2020**, 846.

[4] Yadav, T. and Kumar, M. (2020) Differential-ml distinguisher: Machine learning based generic extension for differential cryptanalysis. *IACR Cryptol. ePrint Arch.*, **2020**, 913.

[5] Bellini, E. and Rossi, M. (2020) Performance comparison between deep learning-based and conventional cryptographic distinguishers. *IACR Cryptol. ePrint Arch.*, **2020**, 953.

[6] Pareek, M., Mishra, G., and Kohli, V. (2020) Deep learning based analysis of key scheduling algorithm of PRESENT cipher. *IACR Cryptol. ePrint Arch.*, **2020**, 981.

[7] Baksi, A., Breier, J., Chen, Y., and Dong, X. (2021) Machine learning assisted differential distinguishers for lightweight ciphers. *Design, Automation & Test in Europe Conference & Exhibition, DATE 2021, Grenoble, France, February 1-5, 2021*, pp. 176–181. IEEE.

[8] Hou, Z., Ren, J., and Chen, S. (2021) Cryptanalysis of round-reduced SIMON32 based on deep learning. *IACR Cryptol. ePrint Arch.*, **2021**, 362.

[9] Benamira, A., Gérault, D., Peyrin, T., and Tan, Q. Q. (2021) A deeper look at machine learning-based cryptanalysis. In Canteaut, A. and Standaert, F. (eds.), *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, Lecture Notes in Computer Science, **12696**, pp. 805–835. Springer.

[10] Ankele, R. and Kölbl, S. (2018) Mind the gap - A closer look at the security of block ciphers against differential cryptanalysis. In Cid, C. and Jr., M. J. J. (eds.), *Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers*, Lecture Notes in Computer Science, **11349**, pp. 163–190. Springer.

[11] Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., and Wingers, L. (2015) The SIMON and SPECK lightweight block ciphers. *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*, pp. 175:1–175:6. ACM.

[12] Mouha, N., Mennink, B., Herrewege, A. V., Watanabe, D., Preneel, B., and Verbauwhede, I. (2014) Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In Joux, A. and Youssef, A. M. (eds.), *Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers*, Lecture Notes in Computer Science, **8781**, pp. 306–323. Springer.

[13] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., Seurin, Y., and Vikkelsoe, C. (2007) PRESENT: an ultra-lightweight block cipher. In Paillier, P. and Verbauwhede, I. (eds.), *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, Lecture Notes in Computer Science, **4727**, pp. 450–466. Springer.

[14] Coppersmith, D., Holloway, C. L., Matyas, S. M., and Zunic, N. (1997) The data encryption standard. *Inf. Secur. Tech. Rep.*, **2**, 22–24.

[15] Huang, S., Wang, X., Xu, G., Wang, M., and Zhao, J. (2017) Conditional cube attack on reduced-round keccak sponge function. In Coron, J. and Nielsen, J. B. (eds.), *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II*, Lecture Notes in Computer Science, **10211**, pp. 259–288.

[16] He, K., Zhang, X., Ren, S., and Sun, J. (2016) Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society.

[17] Biham, E. and Chen, R. (2004) Near-collisions of SHA-0. In Franklin, M. K. (ed.), *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, Lecture Notes in Computer Science, **3152**, pp. 290–305. Springer.

[18] Lee, J., Heo, M., Kim, K., and Kim, C. (2018) Single-image depth estimation based on fourier domain analysis. *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 330–339. Computer Vision Foundation / IEEE Computer Society.

[19] Schüldt, C., Laptev, I., and Caputo, B. (2004) Recognizing human actions: A local SVM approach. *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004*, pp. 32–36. IEEE Computer Society.

[20] Tosi, F., Aleotti, F., Poggi, M., and Mattoccia, S. (2019) Learning monocular depth estimation infusing traditional stereo knowledge. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 9799–9809. Computer Vision Foundation / IEEE.

[21] Chen, Y., Yu, L., Ota, K., and Dong, M. (2019) Hierarchical posture representation for robust action

recognition. *IEEE Trans. Comput. Soc. Syst.*, **6**, 1115–1125.

[22] Kingma, D. P. and Ba, J. (2015) Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.*

[23] Chollet, F. et al. (2015). Keras. `https://github.com/fchollet/keras`.

[24] Abed, F., List, E., Lucks, S., and Wenzel, J. (2014) Differential cryptanalysis of round-reduced simon and speck. In Cid, C. and Rechberger, C. (eds.), *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, Lecture Notes in Computer Science, **8540**, pp. 525–545. Springer.

[25] Roelofs, R., Shankar, V., Recht, B., Fridovich-Keil, S., Hardt, M., Miller, J., and Schmidt, L. (2019) A meta-analysis of overfitting in machine learning. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 9175–9185.

[26] Wang, M. (2008) Differential cryptanalysis of reduced-round PRESENT. In Vaudenay, S. (ed.), *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings*, Lecture Notes in Computer Science, **5023**, pp. 40–49. Springer.

[27] Biham, E. and Shamir, A. (1991) Differential cryptanalysis of des-like cryptosystems. *J. Cryptol.*, **4**, 3–72.