Effective Network Parameter Reduction Schemes for Neural Distinguisher

JiaShuo Liu, JiongJiong Ren and ShaoZhen Chen

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

Abstract. In CRYPTO 2019, Gohr opens up a new direction for cryptanalysis. He successfully applied deep learning to differential cryptanalysis against the NSA block cipher SPECK32/64, achieving higher accuracy than traditional differential distinguishers. Until now, one of the mainstream research directions is increasing the training sample size and utilizing different neural networks to improve the accuracy of neural distinguishers. This conversion mindset may lead to a huge number of parameters, heavy computing load, and a large number of memory in the distinguishers training process. However, in the practical application of cryptanalysis, the applicability of the attacks method in a resource-constrained environment is very important. Therefore, we focus on the cost optimization and aim to reduce network parameters for differential neural cryptanalysis.

In this paper, we propose two cost-optimized neural distinguisher improvement methods from the aspect of data format and network structure, respectively. Firstly, we obtain a partial output difference neural distinguisher using only 4-bits training data format which is constructed with a new advantage bits search algorithm based on two key improvement conditions. In addition, we perform an interpretability analysis of the new neural distinguishers whose results are mainly reflected in the relationship between the neural distinguishers, truncated differential, and advantage bits. Secondly, we replace the traditional convolution with the depthwise separable convolution to reduce the training cost without affecting the accuracy as much as possible. Overall, the number of training parameters can be reduced by less than 50% by using our new network structure for training neural distinguishers. Finally, we apply the network structure to the partial output difference neural distinguishers. The combinatorial approach have led to a further reduction in the number of parameters (approximately 30% of Gohr's distinguishers for SPECK).

Keywords: Deep Learning · Block Cipher · Neural Distinguisher · Depthwise Separabe Convoution · SPECK

1 Introduction

As deep learning spreads to various fields, large-scale neural networks have become more important in computer tasks. As the most important neural network, the trend of Convolutional Neural Network (CNN) in various fields is to make more complicated and deeper networks to get a higher accuracy such as image classification [1–3], object detection [4–7] and semantic segmentation [8,9]. However, the huge number of parameters, heavy computing load, and large number of memory access lead to huge power consumption, which makes it difficult to apply the model to portable mobile devices with limited hardware resources. Therefore, models with fewer parameters are getting more and more attention [10–12]. MobileNet [13] is one of them which provides a solution for mobile and embedded devices. Instead of using the standard convolution, MobileNet uses a special convolution called depthwise separable convolution. With the depthwise separable

convolution, it needs about one-eighth of the computational cost and has only a little drop in accuracy.

It is worth mentioning that some researchers have investigated the viability of applying deep learning to cryptanalysis. In CRYPTO 2019, Gohr [14] combined deep learning with differential analysis and first presented differential neural cryptanalysis. He trained neural distinguishers of SPECK32/64 based on the deep residual neural networks (ResNet) [2]. The labeled data used as training data is composed of ciphertext pairs: half the training data comes from ciphertext pairs encrypted by plaintext pairs with a fixed input difference, and half comes from random values. Gohr obtained 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.

Subsequently, Gohr's neural distinguishers have been gradually derived into two mainstream directions. One is adopting different neural networks to improve accuracy. Bao et al. [15] used Dense Network and Squeeze-and-Excitation Network to train neural distinguishers, and obtained effective (7-11)-rounds neural distinguishers for SIMON32/64. Zhang et al. [16] adopted the inception block to construct a new neural network architecture to train neural distinguishers for (5-8)-rounds SPECK32/64 and (7-12)-rounds SIMON32/64. The other popular research direction is changing the input data format of neural distinguishers. Chen et al. [17] proposed multiple groups of ciphertext pairs instead of single ciphertext pair [14] as the training sample and effectively improved the accuracy of the (5-7)-rounds neural distinguishers of SPECK32/64. Hou et al. [18] built multiple groups of output differences pairs instead of multiple groups of ciphertext pairs [17] to further improve the accuracy of neural distinguishers.

Similar to the development trajectory of deep learning, developments in neural distinguishers face a common problem. Researchers have focused on how to better improve neural distinguisher performance without exploring the computational costs. This conversion mindset may lead to a huge number of parameters, heavy computing load, and a large number of memory, which is difficult to apply the cryptanalysis method in the resource-constrained environment. Therefore, we decide to focus on the cost optimization of differential neural cryptanalysis. There is not much relevant work available, and the two notable researches are as follows. Nicoleta et al. [19] first evaluated Gohr's neural distinguishers with the Lottery Ticket Hypothesis. The Lottery Ticket Hypothesis states some subnetworks match or even outperform the accuracy of the original network. They conducted pruning based on average activations equal to zero and obtained a smaller or better-performing network. Amirhossein et al. [20] showed experimentally that not all the bits in a block are necessary as features to have an effective neural distinguisher. They also found that different selections of features lead to vastly different accuracy results and that certain bits are consistently better than others for this purpose. On this basis, They proposed a new feature selection method for obtaining a much more compact network for training neural distinguishers.

The existing works are great explorations of the cost optimization of differential neural cryptanalysis, but the development of reducing the time and resources consumed by the attack while ensuring the key recovery accuracy as much as possible still needs to be further explored. Inspired by these works, we propose two cost-optimized neural distinguisher improvement methods from the aspect of data format and network structure, respectively. Our major contributions are listed as follows:

- Firstly, we propose a new training sample generation format to reduce the training cost and significantly improve the training efficiency. Specifically, by observing the round function of SPECK cipher, we propose two key improvement conditions for constructing new advantage bits search algorithm in the partial differential ML-based distinguisher. The symmetry condition provides a better strategy for the advantage bits search algorithm, which ensures the neural distinguisher accuracy. The

difference condition can significantly reduce the number of bits used in the training samples and thus reducing the training parameters. Based on the two improved conditions, we propose a new advantage bits selection algorithm and experimentally verify the validity of the algorithm. We name the new neural distinguishers training with our new advantage bits selection algorithm as partial output difference neural distinguishers. Our new neural distinguishers can further reduce the number of bits included in the training samples while keeping the accuracy. In addition, we perform an interpretability analysis of the new neural distinguishers whose results are mainly reflected in the relationship between the neural distinguishers, truncated differential, and advantage bits. The advantage bits are determined based on the accuracy of multiple neural distinguishers, while the accuracy of the neural distinguishers is influenced by the position and number of bits used. The advantage bits can be propagated from the truncated difference through two rounds of the SPECK round function, while the advantage bits used in the training samples also reflect the important role of truncated differential in the binary classification process.

- Secondly, we propose a new network structure with a smaller training parameter number. Specifically, we replace the traditional convolution in the convolution blocks with a depthwise separable convolution to reduce the training cost without affecting the accuracy as much as possible. The effect of the new network structure is mainly reflected in the neural distinguisher accuracy and the number of training parameters. In particular, we experimentally show that the three-dimensional depthwise separable convolution has achieved excellent results on MCP (Multiple Ciphertext Pairs) data formats [17]. Overall, the number of training parameters can be reduced by 50% by using the new network structure for training neural distinguishers. Based on the new network structure, we explore the number of required depthwise separable convolutional blocks. The experimental results show that the new network structure can only use one depthwise separable convolutional block while maintaining the existing accuracy, which further reduces the number of parameters and training costs. Noteworthy, we apply the improved depthwise separable convolutional network structure to the partial output difference neural distinguishers. The combination of the two improved schemes results in another significant reduction in the number of parameters.

The rest of the paper is organized as follows. Section 2 gives a brief description of SPECK and introduces Gohr's neural distinguisher and depthwise separable convolution. In Section 3, we present the partial output difference neural distinguishers and give a validity interpretation from the cryptanalysis perspective. We adopt the ideas of depthwise separable convolution to improve network structure and further reduce the training costs in Section 4. Finally, our work is summarized in Section 5.

2 Preliminaries

2.1 Notations

Table 1 presents the major notations.

2.2 Brief Description of SPECK

SPECK is a family of lightweight block ciphers proposed by the National Security Agency (NSA) [21]. SPECK adopts ARX construction that applies a composition of the basic functions of modular addition, bitwise rotation, and bitwise addition. Various versions of SPECK were proposed which are defined for block size $2n$ and key sizes k: 32/64, 48/72,

Table 1: The notations

Notation	Description
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 $\Delta C_r = C_r \oplus C'_r$

48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, and 128/256. In this article, we will focus mainly on SPECK with 32 bits block size and 64 bits key size (for simplicity, SPECK32/64 will be referred to as SPECK32 in the rest of the article).

SPECK has a function that iterates for many rounds until the ciphertext is generated. For SPECK32, the round function $F : F_2^k \times F_2^{2k} \to F_2^{2k}$ takes as input a 16-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:

$$L_{i+1} = \left(\left(S^{-7}L_i\right) \& R_i\right) \oplus rk_i, R_{i+1} = L_{i+1} \oplus \left(S^2 R_i\right) \tag{1}$$

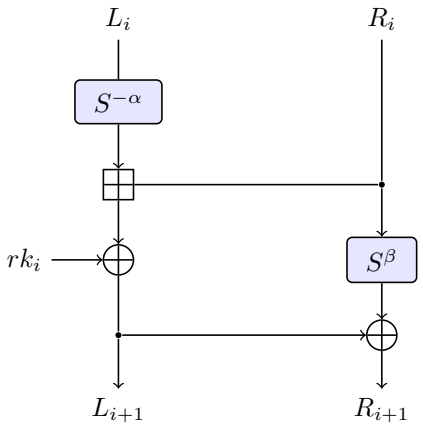

Figure 1: The round function of SPECK

2.3 Differential-based Neural Distinguishers

In the analysis of block cipher security, differential cryptanalysis [22] is one of the most effective attack methods. It is a selective plaintext attack and was firstly introduced by Biham and Shamir to analyze DES block cipher in 1990. Its basic idea is to study the probability of differential propagation of specific plaintext differential values in the encryption process. Specifically, in a random permutation with block size of n, the average

probability of a differential is: $\Pr(\Delta_{in} \to \Delta_{out}) = 2^{-n}$ for all $\Delta_{in}, \Delta_{out}$. If an attacker can find a differential that $\Pr(\Delta_{in} \to \Delta_{out}) > 2^{-n}$, this differential is called differential distinguisher and can be used to attack cipher algorithms.

In [14], Gohr proposed a new differential cryptanalysis method combined with deep learning. He first trained neural distinguishers of SPECK32 based on the deep residual neural networks for distinguishing correct pairs from random pairs. Then he developed a highly selective key search strategy based on a Bayesian optimization method. This deep learning-based differential cryptanalysis can be divided into two parts according to the traditional differential cryptanalysis: generation of neural distinguishers and subkeys recovery based on neural distinguishers. The generation of neural distinguishers is composed of two steps: training data generation and neural network training.

Step1: Training data generation

10^7 plaintext pairs (P, P') are randomly generated in a fit difference $\Delta_{in} = (0x40, 0x0)$. 10^7 labels $Y \in \{0, 1\}$ are randomly generated and allocated to the samples. For the samples with label 1 which is encrypted from plaintext pairs into $r-$rounds to generate ciphertext pairs; For the samples with label 0, re-generate the right half of the plaintext pairs P' randomly, and then encrypt r-rounds to generate ciphertext pairs.

Step2: Neural network training

Set the structure of the residual neural network used for training. Based on the training samples generated above, the neural network is trained to perform a binary classification task: distinguish the ciphertext pairs encrypted with fixed differences from the random ciphertext pairs. The trained neural network is evaluated by generating 10^6 test samples (repeat Step1), and if the accuracy is higher than 50%, the network is a valid neural distinguisher.

After obtaining a valid neural distinguisher, Gohr gave a specific procedure for key recovery attacks (Algorithm 1).

Algorithm 1 [14] Gohr's Key Research

Input: Ciphertext pairs set $C = \{C_0, C_1, ..., C_{n-1}\}$, r-round neural distinguisher ND_r, number of candidates to be generated t, number of iterations l.

Output: Key set L.

$S \leftarrow \{sk_0, sk_1, ..., sk_{t-1}\}$, $sk_i \neq sk_j$ if $i \neq j$
$L \leftarrow \{\}$
for $j \in \{0, 1, ..., l-1\}$ **do**
  $P_{i,sk} \leftarrow Decrypt_1(C_i, sk)$ for all $i \in \{0, 1, ..., n-1\}$ and $sk \in S$
  $v_{i,sk} \leftarrow ND_r(P_{i,sk})$ for all $i \in \{0, 1, ..., n-1\}$ and $sk \in S$
  $w_{i,sk} \leftarrow \log_2\left(\frac{v_{i,sk}}{1-v_{i,sk}}\right)$ for all $i \in \{0, 1, ..., n-1\}$ and $sk \in S$
  $L \leftarrow L||[(sk, \sum_{i=0}^{n-1} w_{i,sk})$ for $sk \in S]$
  $m_{sk} \leftarrow \sum_{i=0}^{n-1} w_{i,sk}/n$ for $sk \in S$
  $\lambda_{sk} \leftarrow \sum_{i=0}^{t-1}\left(\frac{m_{sk_i}-\mu_{sk_i \oplus sk}}{\sigma_{sk_i \oplus sk}}\right)^2$ for $sk \in \{0, 1, ..., 2^{16}-1\}$
  $S \leftarrow argsort_{sk}(\lambda)[0 : t-1]$
end for
return L

In Algorithm 1, the $r-$round neural distinguisher is used for a key recovery attack on $(r+1)$-round SPECK32. Gohr found that since the differential propagation is probabilistic, the encrypted positive samples are not guaranteed to be traveled correctly along the differential path. Therefore, he used k neutral bits [23] to create a ciphertext structure that generates 2^k ciphertext pairs from the same plaintext pair by changing different neutral bits and then encrypting them in (r+1)-round. Gohr gave the generated ciphertext structure to the algorithm and then set the corresponding parameters: the number of

candidate keys and the number of iterations. The algorithm first tries and selects the best ciphertext structure on each structure. Finally, all guessed subkeys are scored and the highest scoring subkey is considered the most likely correct subkey.

2.4 Depthwise Separable Convolution

Depthwise Separable Convolution [13] is a key building block for many efficient neural network architectures [24, 25] and we use them in the present work as well.

Depthwise Separable Convolution was proposed for solving the problem of a large number of training parameters in neural networks. Depthwise separable convolution replaces the standard convolution in the neural network which can reduce the number of parameters required for network training and training time while greatly guaranteeing the performance of the training model. Therefore, the significance of using depthwise separable convolution for building a lightweight block cipher SPECK neural distinguisher is obvious.

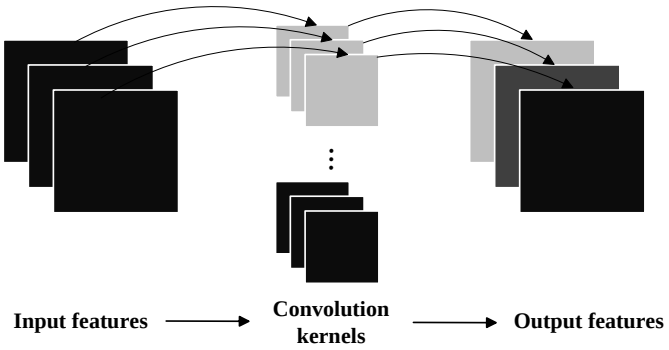

Figure 2: Standard convolution

The standard convolution operation is shown in Figure 2. Give a specific example, if the number of input feature channels is 3, the number of convolution kernel channels is also 3. The specific process of convolution operation is, the convolution of corresponding channel positions to obtain the output feature channels respectively. The number of output feature channels is equal to the number of convolution kernels.

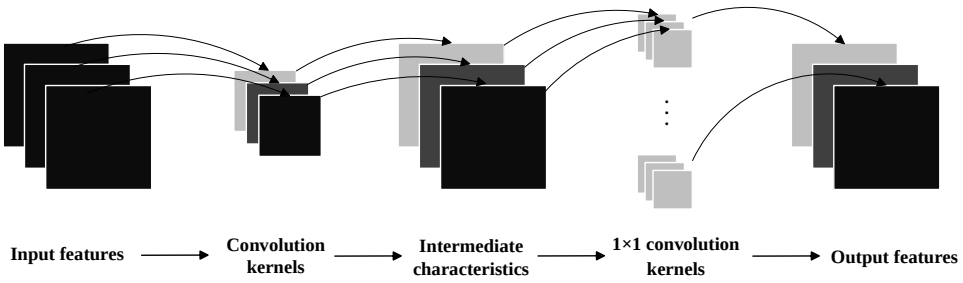

Figure 3: Depthwise separable convolution

The depthwise separable convolution is shown in Figure 3, which mainly decomposes the standard convolution into two steps, depth convolution and 1×1 pointwise convolution. The

specific operation is: each channel of the input features is convolved with the corresponding single-channel convolution kernel to obtain the intermediate features. Then, M 1×1 convolution kernels are used to integrate all the intermediate features to obtain the output features containing M features.

It is important to analyze the number of parameters and operations of the standard convolution and the depthwise separable convolution. Assume that the input feature size is $D_F \times D_F \times M$ and the convolution kernel size is $D_k \times D_k \times M$, whose number is N. It is supposed that one convolution operation is performed for each point in the corresponding input feature, so a total of $D_F \times D_F \times D_k \times D_k \times M$ calculations are required for a single convolution. As a result, the standard convolutional operations and parameter quantities are calculated as follows.

$$F_1 = D_F \times D_F \times D_k \times D_k \times M \times N \tag{2}$$

$$P_1 = M \times D_k \times D_k \times N \tag{3}$$

The number of operations and the number of parameters for the depthwise separable convolution are calculated as follows.

$$F_2 = D_F \times D_F \times D_k \times D_k \times M + M \times 1 \times 1 \times N \tag{4}$$

$$P_2 = M \times D_k \times D_k + M \times 1 \times 1 \times N \tag{5}$$

The ratio of the depthwise separable convolution to the number of standard convolutional parameters is denoted as δ and calculated as follows.

$$\delta = \frac{P_2}{P_1} = \frac{M \times D_k \times D_k + M \times 1 \times 1 \times N}{M \times D_k \times D_k \times N} = \frac{1}{N} + \frac{1}{D_k^2} \tag{6}$$

It can be seen that using depthwise separable convolution instead of standard convolution can significantly reduce the number of parameters required to train the neural network, thus reducing the training cost of the neural network.

3 Partial Output Difference Neural Distinguishers

Currently, the key recovery attack by differential neural cryptanalysis is effective on several lightweight block ciphers [14, 26]. Lightweight block ciphers are usually used in scenarios with resource-constrained hardware, so the cost constraints for implementing key recovery attacks are even greater. For the SPECK family, the training sample is a $4n$ bits ciphertext pair, and the cost to complete the neural distinguisher training will increase significantly as the block size $2n$ keeps increasing. Therefore, reducing the time and resources consumed by the attack while ensuring the key recovery accuracy as much as possible is a key direction of the current differential neural cryptanalysis. Amirhossein et al. [20] proposed a partial ML-distinguisher, but the explorations they made were limited to distinguisher feasibility. They did not make an exploration in terms of integration with the cryptographic techniques, nor did they make a theoretical analysis.

In this section, we propose a new neural distinguisher model based on traditional cryptanalysis techniques and theoretical analysis. Our neural distinguisher model reduces the size of training samples to $\frac{1}{16}$ of ciphertext pairs, but still maintains great accuracy. Meanwhile, we analyze the reason why our distinguisher model can perform well from the theoretical perspective.

3.1 New Neural Distinguisher Model

Our new neural distinguisher model consists of three parts: advantage bits search, training sample generation and neural network training.

Step 1: Advantage bits search

In EUROCRYPT 2021, Benamira et al. [27] proved that Gohr's neural distinguishers made their decisions on the difference of ciphertext pair and the internal state difference in the penultimate and ante-penultimate rounds. In traditional cryptanalysis, the probability is different between different bit positions of the differential characteristics. In some cipher algorithms, the probability fluctuation can be very large. Therefore, the learnable features that can be captured by the neural distinguishers are asymmetric for different bit positions. We further explored the bit positions in the ciphertext pairs used as training samples, with the aim of using the smallest number of bits to obtain an effective distinguisher.

Algorithm 2 Advantage Bits Search

Input: Ciphertext pairs set $C = \{C_0, C_1, ...C_{m-1}\}$, Neural network structure, Block size $2n$, Number of output bits N, Accuracy verification times k.

Output: Advantage bit set S.

$Acc \leftarrow \{\}$
Initialize an all-zero matrix $M_{2n\times k}$
for each $i \in \{0, 1, ..., k-1\}$ **do**
    Randomly select $N/2$ bit positions to form the set $B = \left\{b_0, b_1, ..., b_{\frac{N}{2}-1}\right\}$ for all $b_i \in \{0, 1, ..., n-1\}$
    $B \leftarrow B || \left\{b_0 + n, b_1 + n, .., b_{\frac{N}{2}-1} + n\right\}$
    $C' = \left\{C'_0, C'_1, ...C'_{m-1}\right\} \leftarrow$ Select bit positions in B for all $C_i \in \{C_0, C_1, ...C_{m-1}\}$
    $ND \leftarrow$ TrainNetwork(C')
    $Acc \leftarrow Acc||$AccuracyEvaluation(ND)
    for each $j \in \{0, 1, ..., 2n\}$ **do**
        if $j \in N$ **then**
            $M[j][i] = 1$
        else
            $M[j][i] = 0$
        end if
    end for
end for
$Score \leftarrow \{\}$
for each $i \in \{0, 1, ..., 2n-1\}$ **do**
    $Score \leftarrow Score || \frac{\sum_{j=0}^{k} M[i][j] * Acc[j]}{\sum_{j=0}^{k} M[i][j]}$
end for
$S \leftarrow argsort(Score)[0 : N-1]$
return S

The process of the advantage bits search algorithm is designed as follows. Firstly, setting the number of target bits N and the accuracy verification times k for generating training samples. Taking SPECK32 algorithm as an example (block size $2n = 32$), we set $N = 8$, $k = 100$. Then, we randomly select $N/2$ bits from the first 16 bit positions and record them in the set B. Next, the selected bit position is shifted right by 16 bits and added to B. According to the set B, the ciphertext pairs set are transformed to form the new training set C', and record the verification set accuracy of the new neural distinguisher based C'. We repeat the above steps k times, and we will get k partial neural

distinguishers whose results are saved in the set Acc.

The M matrix record the bit positions selected in each cycle. Then each bit position is evaluated by calculating the average of the neural distinguisher accuracy. Finally, we output the set of bit positions that score the top N. We describe the above search process as Algorithm 2.

Step 2: Training sample generation

New neural distinguisher is to accomplish the binary classification task of distinguishing correct pairs from random values. We denote ciphertext pairs corresponding to plaintext pairs with the fixed difference as positive samples and denote ciphertext pairs corresponding to plaintext pairs with a random difference as negative samples. The positive sample generation method is described in Figure 4.

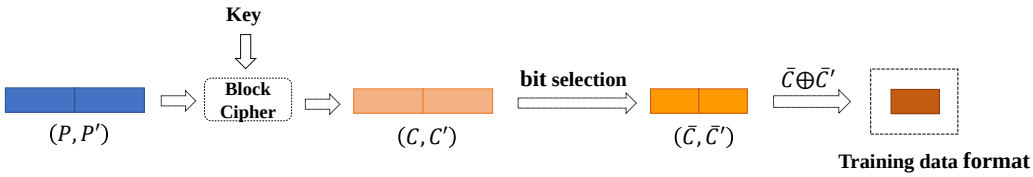

Figure 4: Training data generation

The single plaintext pair (P, P') is encrypted by a random master key to obtain single ciphertext pair (C, C'). The ciphertext pair is then processed based on the advantage bits set obtained in the first step. The processed ciphertext pairs are converted output difference $(\bar{C} \oplus \bar{C}')$ to obtain 8-bits positive training samples.

Step 3: Neural network training

To demonstrate the effect of our neural distinguisher model, we use the same network as Gohr's work [14]. 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. And the neural network training parameters are shown in Table 2.

Table 2: Parameters of the network architecture for training neural distinguishers

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	30	Loss function	MSE

Because we use partial bits of ciphertext pairs and convert them to difference as the training sample format for the neural distinguishers, we call the new distinguishers the partial output difference neural distinguishers.

3.2 Results

Before training the distinguishers, the number and position of the used bits need to be determined. The bit positions are determined by starting from the most efficient. And the number of bits needs to be determined according to the accuracy requirements and cost requirements. For SPECK32, we set the number of bits to block size 32 in Algorithm 2, so that we can obtain the score of all bit positions. For easy observation, we normalize the scores and the results are shown in Table 3.

Table 3: The score of each bit pairs in the advantage bits search

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$i+16$	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Score	0.187	0.277	0.210	0.262	0.364	0.512	0.129	0.000	0.290	0.149	0.144	0.190	1.000	0.760	0.596	0.445

Observing the Table 3, there is significant disequilibrium between bit positions, and most bits have low ratings. Among them, there are 6 pairs of bit positions with high scores: (4,20), (12,28), (13,29), (14,30), (5,21), (15,31). We highlight the bit positions with the highest scores. The darker the color, the more important it is for the accuracy of the neural distinguishers. We will mainly use these 6 pairs of bit positions in the actual neural distinguisher training process.

During the implementation of the bits selection algorithm, the advantage bits set can be divided into two related subsets: randomly select $N/2$ bits from the first 16 bit positions and shift the existing bit positions by 16 bits to the right of the bit positions. The reason is that after observing the SPECK round function, the right half of the round function output can be written as $R_{i+1} = L_{i+1} \oplus \left(S^2 R_i\right)$ which reflects the close relationship between the left and right halves of the output. This statement also is verified by the distribution of the scoring results for each bit position of SPECK32 in [20]. In addition, such a selection approach leads to another cost-saving breakthrough direction. We transform the output pairs into differences during the construction of the training samples. There are two advantages of this approach:

1. Providing the neural network with more directly learnable features;
2. Reducing the number of bits included in the training samples.

To illustrate the advantages of our method more clearly, we train two sets of neural distinguishers simultaneously. One is to complete the method in [20] under the same training conditions, and the second is to use our bits selection algorithm but the training samples are not converted into differences. The results are shown in Table 4 (all results are performed with the same network parameters in Table 2).

Table 4: Accuracy of the partial output difference neural distinguishers for 6-round SPECK32

Reference	Condition	Bit Set	Number	Accuracy
[20]	None	[12,13,14,28]	8	65.2%
	None	[12,13,14,15,28,29]	12	66.8%
	None	[5,12,13,14,15,21,28,29]	16	68.8%
This paper	Symmetry	[12,13,28,29]	8	66.8%
	Symmetry Difference	[12,13,28,29]	4	66.7%
	Symmetry Difference	[12,13,14,28,29,30]	6	67.3%
	Symmetry Difference	[5,12,13,14,21,28,29,30]	8	67.3%
	Symmetry Difference	[5,12,13,14,15,21,28,29,30,31]	10	68.8%

The symmetry condition indicates that the left half of the training sample corresponds to the right half of the position; the difference condition indicates that the training sample is the output difference. For the non-differential condition, each ciphertext in the training sample (ciphertext pair) selects the bit position according to the bit set B, so the number of bits in a training sample is 2×number of elements in the bit set B. For the difference condition, the training sample (output difference) selects the bit position according to the

bit set B, so the number of sample bits is the number of elements in the bit set B.

The results presented in Table 4 show the effectiveness of our proposed two improved conditions for partial bit ML-distinguisher. Compare with the partial bit ML-distinguisher using 8-bit positions in [20], we use the same number of bit training samples in the condition of symmetry, and the accuracy of our distinguisher is improved. In addition, we use the difference condition to train with a smaller training sample size, and we call the new distinguishers partial output difference neural distinguishers. The results in the table 4 show that the new distinguisher accuracy using a 4-bit training sample is comparable to the distinguisher in [20] using a 12-bit training sample, and the new distinguisher accuracy using a 10-bit training sample is comparable to the distinguisher in [20] using a 16-bit training sample.

The proposed bits selection conditions further improve the effectiveness and make it possible to use fewer bits for training the neural distinguishers. At the same time, the benefits of reducing the training sample size are obvious which can reduce the number of network parameters. This advantage means the cost time and data complexity of model training will be significantly reduced. Besides, the partial output difference neural distinguishers can reduce the time and data complexity by reducing the guessed subkey space in the key recovery phase.

3.3 Interpretation from Cryptanalysis Perspective

For the partial output difference neural distinguisher, its effectiveness is validated in Table 4. Yet, our distinguisher model opened some questions. The most important one is the interpretability of the Algorithm 2 (advantage bits search). In Algorithm 2, the advantage bits are chosen based on the accuracy obtained from multiple distinguishers trained using different bit sets. An obvious issue with a neural distinguisher is that its black-box nature is not telling us much about the actual dominance of the analyzed bit positions.

In this section, we want to find out why and how the advantage bits search algorithm works in a cryptanalytic sense. Essentially, we want to answer the following question: What types of bit positions are preferred by the distinguisher? If the neural network is using some currently unknown information to evaluate bit positions, we want to infer the additional statistics it utilizes. If not, we want to find out what is causing the differences between bit positions.

In [27], Benamira et al. proposed a thorough analysis of Gohr's deep neural network distinguishers of SPECK-32/64 [14]. They concluded that neural distinguishers are not only basing their decisions on the ciphertext pairs difference, but also the internal state difference in penultimate and antepenultimate rounds. SPECK32 has the following specific internal state difference:

$$\begin{aligned} &\text{3 rounds: } 10*****00*****00\ 10*****00*****10\\ &\text{4 rounds: } 10*****10*****10\ 10*****10*****00 \end{aligned}$$

They conducted a 5-round neural distinguisher which was used to evaluate the set of positive samples satisfying the above 3-round or 4-round truncated differential. The accuracy is closer to Gohr's neural distinguisher [14]. In other words, they successfully verified that the recognition ability of the neural distinguisher is based on truncated differential and not just by identifying a different set of ciphertext pairs.

Based on the conclusion, we conjecture that the source of the advantage bits used in the partial output difference neural distinguishers is precisely the truncated differential. So we derive the propagation of the 4-round truncated differential for SPECK32.

Figure 5 shows us how the bits evolve along the most probable difference path from a 4-round truncated differential to a 6-round output. As it goes through the modular addition operation, we highlight the bits that have a relatively high probability of being associated

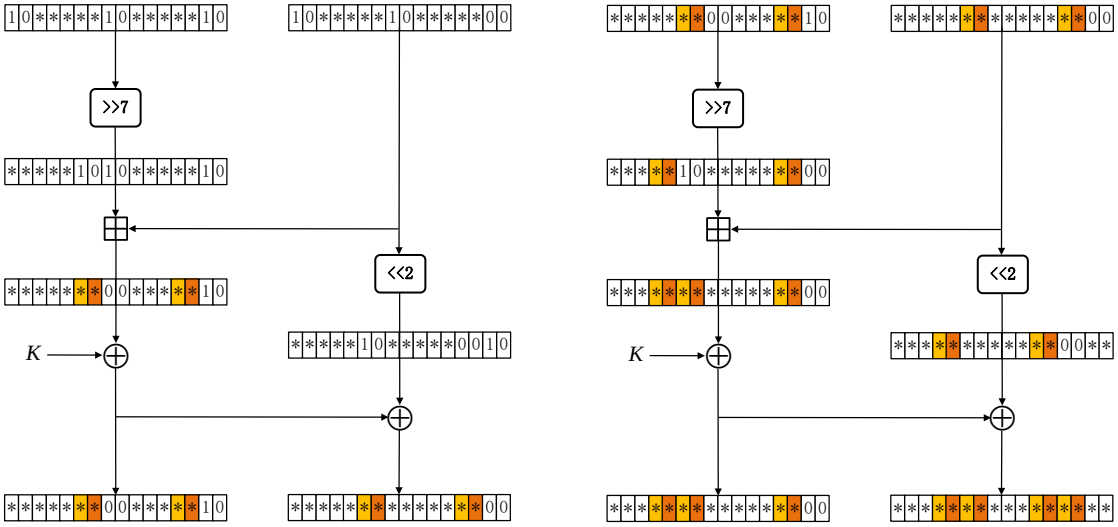

Figure 5: Showing the propagates from the 4-round truncated differential to 6-round output. The darker the color, the higher the probability that it has a carry propagated to.

with the most probable differential. The darker the color, the higher the probability of the difference being toggled. In each round, the fixed difference is changed to a random bit by operating with a random bit, but we still highlight the entry bits. Finally, we obtain a set of possible carrying entry bits $\{3, 4, 5, 6, 12, 13, 19, 20, 21, 22, 26, 27, 28, 29\}$. In addition, we add to the set the bit positions $\{14, 15\}$ that still have a fixed difference with their counterparts in the right half $\{30, 31\}$. Surprisingly, all the elements of the advantage bit set we obtained in Table 3 (6 pairs of 12 bits) are included in the set. This confirms our conjecture about the reason for the advantage bits.

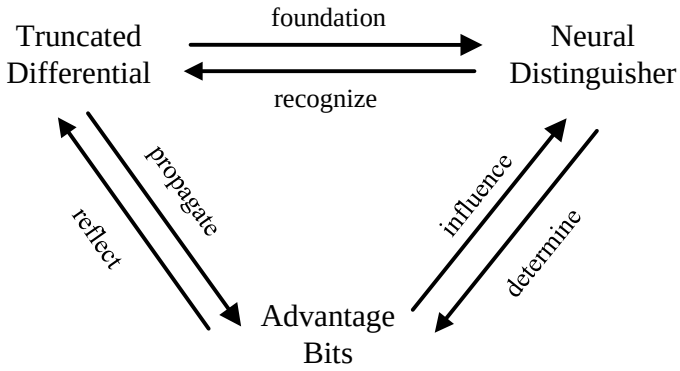

Figure 6: Relationship between the distinguisher, truncated difference and advantage bits

We sorted out the relationship between the distinguisher, truncated differential, and advantage bits which are shown in Figure 6 to help understand the relationship. In summary, the truncated differential provides the learnable features for training neural distinguishers, the distinguisher also recognizes positive samples through truncated differential. For our

new distinguishers, the advantage bits are obtained by training neural distinguishers and then building the partial output difference neural distinguishers using advantage bits. After our theoretical derivation, we find that the advantage bits are effective significantly because they reflect the propagation of the truncated differential.

4 Improving Network Structure with Depthwise Separable Convolution

At present, the neural network improvement of training neural distinguishers mainly focuses on the accuracy, but the effect is not obvious. The neural distinguisher training is the main consumption of the precomputation in the key recovery framework. How to train the network more efficiently is a key research problem. Meanwhile, the reduction of cost in network training can make the application of differential neural cryptanalysis more widely.

In this section, we first describe the core layers that the network structure is built on depthwise separable convolutions to train the neural distinguishers. The starting point of the new network is to reduce the number of training parameters and computation. Then we describe our comparative experimental ideas and results.

4.1 New Network Structure

We provide in Figure 7 a representation of our new network structure, whose main components are four parts: an input layer for processing training datasets, an initial convolutional layer, a residual tower consisting of multiple two-layer depthwise separable convolutional neural networks, and a prediction head consisting of fully connected layers.

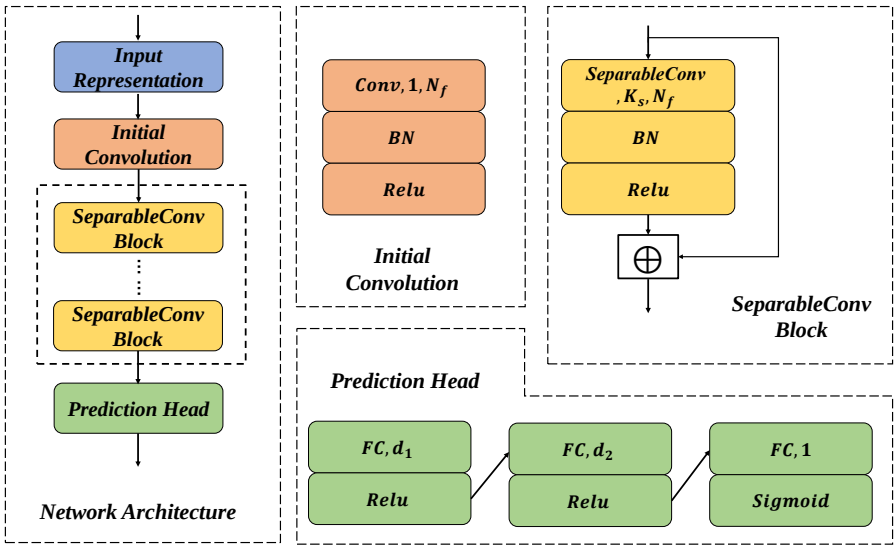

Figure 7: New network structure

Input Representation. We use the SCP (Single Ciphertext Pair) [14] and the MCP (Multiple Ciphertext Pairs) [17] on the input format to demonstrate the excellence of depthwise separable convolution in two-dimensional and three-dimensional training data. Thus, the neural network accept data of the form $\{C_1, C_1', ..., C_m, C_m'\}$ $(m \in 1, 2, 4)$ and

m ciphertext pairs are arranged in a $m \times 4 \times n$ array. $2n$ represents the block size of the target cipher. (For SPECK32, the block size $2n$ is 32.)

Initial Convolution. The input layer is connected to the initial convolutional layer, which is a bit slice layer including 1×1 convolution kernels, Batch normalization, and a ReLU activation function. Batch normalization plays a critical role in addressing the gradient vanishing/explosion problem. Relu function greatly accelerates the convergence of stochastic gradient descent at a smaller cost compared to the sigmoid/tanh functions.

SeparableConv Blocks. The initial convolution is connected to the SeparableConv blocks. Each block is a residual construction and consists of two groups of depthwise separable convolution, Batch normalization, and a ReLU activation function. Effectively depthwise separable convolution reduces computation compared to traditional layers by almost a factor of $kernelsize^2$. However it only filters input channels, it does not combine them to create new features.

Prediction Head. The SeparableConv blocks are connected to the prediction head. First, two layers are densely connected layers with 64 units and followed by a Batch normalization and a ReLU activation function. Then, the final layer consists of a single output unit using the Sigmoid activation function to output a binary classification result.

4.2 Results and Discussion

In order to compare with the existing results, we keep the conditions remain the same except the network, we use the new network to train the neural distinguishers. When using the SCP training format, we use the new network to train the neural distinguisher for 6-round of SPECK32. When using the MCP training format, we use the new network to train the neural distinguisher for 7-round of SPECK32. Meanwhile, to demonstrate the advantages of the new network, we further approximate the network. We try to reduce the number of separable blocks to obtain a network with fewer training parameters but still get a neural distinguisher with great accuracy.

Table 5: Results of the new network structure

Core Net	Pair	Round	Depth	Total Params	Accuracy	Source
Conv	1	6	5	70177	0.7888	[14]
	2	7	5	129409	0.6393	[17]
	4	7	5	162177	0.6861	
SeparableConv	1	6	5	50661	0.7871	Sect. 4.1
	1	6	1	40421	0.7840	
	2	7	5	50369	0.6470	Sect. 4.1
	2	7	1	38593	0.6452	
	4	7	5	83137	0.6939	Sect. 4.1
	4	7	1	71361	0.6598	

Table 5 shows the experimental results of the new network. For the SCP training format, the accuracy of the 6-round SPECK32 neural distinguisher using a new network is similarly comparable to Gohr's results. However, the number of network training parameters we use is 50611, which is a significant reduce over Gohr's network training parameter number. Meanwhile, for the neural distinguishers obtained by further reducing the separable blocks based on the new network structure, the accuracy is comparable to the original one, and the number of network training parameters is only 58% of Gohr's network. For the MCP training format, we selected ciphertext pairs of 2 and 4 for training the 7-round SPECK32 neural distinguisher. When using two ciphertext pairs in a single training sample, the accuracy of the neural distinguisher obtained using the new network is 64.70%. In comparison with Chen's results, our new neural distinguishers not only improve

the accuracy, but the number of training parameters is only 39% of Chen's network. In addition, the neural distinguisher obtained by reducing the number of separable blocks based on the new network to 1 has the same accuracy as the original one, and the number of trained network parameters is further reduced to 30% of Chen's network. When using four ciphertext pairs in a single training sample, the neural distinguisher accuracy obtained using the new network is 69.39%. In contrast to Chen's results, our new neural distinguisher not only improve the accuracy, but also reduce the number of training parameters of the network, which is only 51% of Chen's network. In addition, the neural distinguishers obtained by reducing the number of separable blocks based on the new network to 1 has a little slightly decrease in accuracy, but the number of trained network parameters is further reduced to 44% of Chen's network.

Finally, we apply the improved depthwise separable convolutional network structure to the output difference neural distinguisher presented in Section 3.1. Our main objective is to observe how much our method saves the number of training parameters and how much it affects the neural distinguisher accuracy.

Table 6: Results of a combination of the new network structure and the output difference neural distinguisher

Core Net	Bit Set	Number	Total Params	Accuracy	Source
Conv	[12,13,28,29]	4	41441	66.7%	[20]
	[12,13,14,28,29,30]	6	43489	67.3%	
	All bits	64	70177	78.9%	[14]
SeparableConv	[12,13,28,29]	4	21921	66.7%	Sect. 4.2
	[12,13,14,28,29,30]	6	23969	67.3%	
	All bits	64	50661	78.7%	Sect. 4.1

The results are presented in Table 6. Compared to the neural distinguishers obtained with the traditional convolutional network in Section 3.1, the new neural distinguishers using depthwise separable convolution can reduce the number of training parameters by 50% without reducing the accuracy. The above results show that our two proposed improvement schemes have their own advantages of reducing the parameters for neural distinguishers, and then the combination of the two schemes can further reduce the training cost.

5 Conclusion

In this paper, we focus on two aspects for reducing the cost of training neural distinguishers, including lightweight optimization in terms of data format and network structure.

Firstly, we make the following optimization contributions in terms of data format. In the existing partial ML-distinguisher construction process, the training data format needs to be determined in advance by the bits selection algorithm. By observing the round function of the SPECK cipher, we propose two key improvement conditions with important effects on the advantage bits selection algorithm. The first improvement condition is symmetry condition. The motivation is that the close relationship between the left and right halves of the output. The second improvement condition is difference condition, which represents the conversion of ciphertext pairs into differences. Without reducing the accuracy of the neural distinguishers, the effect of the difference condition significantly reduces the amount of sample data and the number of parameters in the training process. Based on the above two conditions, we present a new complete advantage bits search algorithm 2, and we conduct a series of experiments for validation effectiveness and finally achieve satisfactory results.

Secondly, in order to optimize the network structure with fewer training parameters, we introduce depthwise separable convolution into the existing network. Observing the results obtained from the experiments, we find that the new network structure is greatly optimized in the number of training parameters and training efficiency. In particular, the new network structure can provide better parameter approximate reduction for the MCP (Mutiple Ciphertext Pairs) training data format. Meanwhile, based on our network structure, we further investigate the required number of SeparableConv blocks. According to the experimental results, the new network structure can still maintain great accuracy when using one SeparableConv block. Furthermore, we combine the two proposed improvements for data format and network structure, and the experimental results show that the new combined neural distinguishers can substantially reduce the training cost with a certain loss of accuracy.

In summary, the proposed parameter optimization scheme in terms of data format and network structure has significant effect on the problem of huge number of parameters and heavy computing load, which provides support for deep learning cryptanalysis technology to be applied to larger block size ciphers. Meanwhile, our results have the potential to open the way for efficient neural cryptanalysis in applications for portable mobile devices with limited hardware resources. For future work, we will explore the network structures that have shown significant results in other areas of deep learning that can be migrated to differential neural cryptanalysis and retain their benefits.

ACKNOWLEDGEMENTS

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

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[6] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[10] Jonghoon Jin, Aysegul Dundar, and Eugenio Culurciello. Flattened convolutional neural networks for feedforward acceleration. *arXiv preprint arXiv:1412.5474*, 2014.

[11] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[12] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

[13] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[14] 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.

[15] Zhenzhen Bao, Jian Guo, Meicheng Liu, Li Ma, and Yi Tu. Enhancing differential-neural cryptanalysis. Springer-Verlag, 2022.

[16] Liu Zhang, Zilong Wang, and Boyang Wang. Improving differential-neural cryptanalysis with inception blocks. *Cryptology ePrint Archive*, 2022.

[17] Yi Chen, Yantian Shen, Hongbo Yu, and Sitong Yuan. A new neural distinguisher considering features derived from multiple ciphertext pairs. *Cryptology ePrint Archive*, 2021.

[18] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Improve neural distinguisher for cryptanalysis. *Cryptology ePrint Archive*, 2021.

[19] Nicoleta-Norica BÈČcuiet. Deep neural networks aiding cryptanalysis: A case study of the speck distinguisher. *applied cryptography and network security*, 2022.

[20] Amirhossein Ebrahimi, Francesco Regazzoni, and Paolo Palmieri. Reducing the cost of machine learning differential attacks using bit selection and apartial ml-distinguisher. *IACR Cryptology ePrint Archive*, 2021.

[21] 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, USA, 2015.

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

[23] Eli Biham and Rafi Chen. Near-collisions of sha-0. *Lecture Notes in Computer Science*, 2004.

[24] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[25] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

[26] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Practical attacks of round-reduced simon based on deep learning. 2022.

[27] Adrien Benamira, David Gerault, Thomas Peyrin, and Quan Quan Tan. A deeper look at machine learning-based cryptanalysis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 805–835. Springer, 2021.