SAT-based Method to Improve Neural Distinguisher and Applications to SIMON

Zezhou Hou · Jiongjiong Ren* · Shaozhen Chen

Received: date / Accepted: date

Abstract Cryptanalysis based on deep learning has become a hotspot in the international cryptography community since it was proposed. The key point of differential cryptanalysis based on deep learning is to find a neural differential distinguisher with longer rounds or higher probability. Therefore it is important to research how to improve the accuracy and the rounds of neural differential distinguisher. In this paper, we design SAT-based algorithms to find a good input difference so that the accuracy of the neural distinguisher can be improved as high as possible. As applications, we search and obtain the neural differential distinguishers of 9-round SIMON32/64, 10-round SIMON48/96 and 11-round SIMON64/128. For SIMON48/96, we choose $(0x0, 0x100000)$ as the input difference and train 9-round and 10-round neural distinguishers of SIMON48/96. In addition, with the automatic search based on SAT, we extend the neural 9-round, 10-round distinguishers to 11-round, 12-round distinguishers by prepending the optimal 2-round differential transition $(0x400000, 0x100001) \xrightarrow{2^{-4}} (0x0, 0x100000)$. Based on the 11-round and 12-round neural distinguisher, we complete a 14-round key recovery attack of SIMON48/96. Our attack takes about 1550s to recover the final subkey. Its average time complexity is no more than $2^{22.21}$ 14-round encryption of SIMON48/96, and the data complexity is about $2^{12.8}$. Similar to 14-round key recovery attack, we perform 13-round key recovery attack for SIMON32/64 with input difference $(0x0, 0x80)$ with a success rate of more than 90%. It takes about 23s to complete an attack with the data complexity no more than $2^{12.5}$ and the time complexity no more than $2^{16.4}$. It is worth mentioning that the attacks are practical for 13-round SIMON32/64 and 14-round SIMON48/96.

Keywords Deep Learning · Block Cipher · Neural differential distinguisher · SIMON · SAT/SMT

1 Introduction

As a chosen plaintext attack, differential cryptanalysis is one of the most powerful analysis techniques used in modern block ciphers. It can achieve key recovery attacks utilizing plain-cipher difference pair, which is expressed in input difference and output difference. The first differential cryptanalysis [1] was presented to break Data Encryption Standard (DES) successfully in 1991. Its basic idea is to obtain real key by analyzing the influence of specific plaintext pairs on the ciphertext pairs. Based on this, Biham and Shamir compleated a differential attack on round-reduced DES. Since then, differential cryptanalysis is widely used in block ciphers such as AES [2], PRESENT [3], SIMON [4] *et al.* In traditional differential cryptanalysis, it is the key to find the high-probability differential characteristics. Recently there are some tools for automatic search for the differential characteristics [5,6]. But this still cannot effectively use the differential characteristics.

Zezhou Hou
E-mail: *zezhouhou1998@163.com*

Jiongjiong Ren*
E-mail: *jiongjiong_fun@163.com*

Shaozhen Chen
E-mail: *chenshaozhen@vip.sina.com*

Information Engineering University, Zhengzhou 450001, Henan, China

Deep learning (DL) has played an important role in many fields, but its development is bumpy. In 1943, McCulloch and Pitts [7] proposed the MP neuron model, which was an abstract and simplified model constructed according to the structure and working principle of biological neurons. It opened the simulation of the neural network, but adjusting the weights relied heavily on manual work, very bad for study. In 1958, on the basis of MP neural, Frank Rosenblatt [8] proposed the single-layer feedforward neural network named single-layer perceptron, which can distinguish between triangle, square and other basic shape. In 1986, the second generation of neural network was put forward by Rumelhart [9]. It changed the single fixed feature layer in the first-generation neural network to multiple hidden layers, using Sigmoid as the activation function. At the same time, it used the idea of Back Propagation (BP), which effectively solved the problem that the first generation only can be used in linear classification. When it came to 21st century, shallower neural network cannot complete the corresponding task with the amount of data increasing. In 2006, Jeffery Hinton *et al.* [10] put forward the concept of deep learning for the first time, which played a huge role. With the development of deep learning, there are diverse neural networks used in diverse fields. Convolutional Neural Networks (CNN) [11] are good at image processing. However, Recurrent Neural Networks (RNN) [12] specialize in natural language processing (NLP). At the same time, Generative Adversarial Networks (GAN) [13] are used to image generation.

Related work: Our work is most closely related to combine deep learning and cryptanalytic techniques. At the Crypto2019, Gohr [14] show that deep learning can produce very powerful cryptographic distinguishers and indicated that the neural distinguisher was better than the distinguisher obtained by traditional approach. He trained a neural distinguisher of Speck32 [4] based on the deep residual neural networks (ResNet) [15], which can distinguish the ciphertext pairs from random data roughly five times lower than a distinguisher using the full difference distribution table. At the same time, he developed a highly selective key search policy based on a variant of Bayesian optimization by using neural distinguishers. With this policy, Gohr described a practical key recovery attack on 11-round Speck, and explained that the complexity of the attack based on deep learning was much lower than the traditional attack. In 2020, Botao Hou *et al.* [16] proposed a new linear attack based on deep residual network. And they devised and trained neural networks and achieved efficient key recovery on DES using trained network models. At the same year, Yi Chen *et al.* [17] proposed a new neural distinguisher model considering derived features from multiple ciphertext pairs, which was used to improve the key recovery attack on 11-round Specck32/64.

Our contribution: In this paper, our contributions are as follows:

- **Design an algorithm based on SAT for searching high-accuracy neural differential distinguisher.** In traditional differential cryptanalysis, the input difference directly affects the probability of the differential characteristic, which is similar to neural differential distinguishers. We propose a basic algorithm based on SAT to search neural differential distinguisher. With automatic search based on SAT, we search for the best differential characteristics with probability $P_{\max}$ and choose the input differences to train neural differential distinguishers, where $P_{\max}$ is the optimal probability. Using the basic algorithm, we search and obtain the neural differential distinguishers of 9-round SIMON48/96. Considering that some distinguishers are also effective, which are trained by input difference from some high-probability characteristics, instead of the optimal characteristics, we improve the basic algorithm to search for more high-accuracy neural distinguishers. We expand the search space of the distinguishers by expanding the probability space to $\left[2^{-\frac{n}{4}} \times P_{\max}, P_{\max}\right]$, where n is the block size. Utilizing the improved algorithm, we search and obtain the 10-round neural differential distinguishers of SIMON48/96 with the accuracy about 57% for the first time. In addition, we get neural differential distinguishers of 11-round SIMON64/128 with the accuracy about 60% and 9-round SIMON32/64 with the accuracy about 60% for the first time. The summary of our algorithms together with choosing Abed's [18] input difference is shown in Table 1.

Table 1: Comparison of neural differential distinguisher. The basic algorithm is shown in Section 3.1 and the improved algorithm is shown in Section 3.2. We train neural differential distinguisher by choosing Abed's [18] input difference. The accuracy of neural differential distinguishers is higher with our algorithm.

Block Cipher	Source of neural differential distinguisher	Round	Accuracy
SIMON32/64	[18]	9	59.07%
	Section 3.2	9	59.77%
SIMON48/96	[18]	9	50.22%
	Section 3.1	9	61.60%
	Section 3.2	10	57.89%
SIMON64/128	[18]	10	58.61%
	Section 3.2	11	59.72%

– **Perform practical key recovery attacks on round-reduced SIMON.** With our search algorithm, we choose $(0x0, 0x100000)$ as the input difference and train 9-round and 10-round neural distinguisher of SIMON48/96. Inspired by Gohr's attack, we describe the wrong key response profile for our 9-round and 10-round neural distinguishers. In addition, we search for the best 2-round characteristic $(0x400000, 0x100001) \xrightarrow{2^{-4}} (0x0, 0x100000)$ by SAT so that we can append two additional rounds to the beginning of the 9-round and 10-round neural distinguishers to 11-round and 12-round neural distinguishers. Based on the wrong key response profile and our neural distinguishers, we complete a 14-round key recovery attack of SIMON48/96 on a workstation configured with *Intel i9-10900K* and *Nvidia TITAN RTX*. Our attack takes about 1550s each time to recover the final subkey. The average time complexity is no more than $2^{22.21}$ 14-round encryption of SIMON48/96, and the average data complexity is about $2^{12.8}$, which is much lower than the complexity of traditional differential cryptanalysis. Similar to key recovery attacks on 14-round SIMON48/96, we perform a key recovery attacks on 13-round SIMON32/64, with a success rate of more than 90%. Fortunately, our attack is practical on 13-round SIMON32/64 and 14-round SIMON48/96. Summary of our results on SIMON32/64 and SIMON48/96 are shown in Table 2.

Table 2: Summary of our results on SIMON32/64 and SIMON48/96. For SIMON32/64, it is considered a success if the guess for the last subkey was incorrect in 2 bits. For SIMON32/64, it is considered a success if the guess for the last subkey was incorrect in 11 bits.

block cipher	Round	Time(s)	Time complexity	Data(CP)	Success Rate
SIMON32/64	13	23	$2^{16.4}$	$2^{12.5}$	93%
SIMON48/96	14	1550	$2^{22.21}$	$2^{12.8}$	59%

Organisation of the paper: The remaining of this paper is organised as follows. Section 2 gives a brief description of SIMON, illustrates how to use SAT in differential cryptanalysis and construct a neural distinguisher. In Section 3, we design an algorithm based on SAT to help us find high-accuracy neural differential distinguishers. Combining with the content of Section 3, we perform key recovery attacks on 14-round SIMON48/96 and 13-round SIMON32/64 in Section 4. Conclusions are drawn in Section 5 where we also suggest further work.

2 Preliminaries

Before introducing our architecture, we briefly review SIMON, SAT-Based automatic search for the differential characteristics and neural differential distinguisher.

2.1 Notations

SIMON$2n/mn$: SIMON acting on $2n$-bit plaintext blocks and using a mn-bit key
$\oplus$: bitwise XOR

$\&$: bitwise AND
$\vee$: bitwise OR
S^j : left circular shift by j bits
K : Master key
k_i : i-round subkey

2.2 Brief Description of SIMON

SIMON [4] is a lightweight block cipher proposed by the NSA. The aim of SIMON is to fill the need for secure, flexible, and analyzable lightweight block ciphers. It is a family of lightweight block ciphers with block sizes of 32, 48, 64, 96, and 128 bits. The constructions are Feistel ciphers using a word size n of 16, 24, 32, 48 or 64 bits, respectively. Table 3 makes explicit all parameter choices for all versions of SIMON.

Table 3: SIMON parameters

block size $2n$	key size mn	word size n	rounds T
32	64	16	32
48	72	24	36
	96	24	36
64	96	32	42
	128	32	44
96	96	48	52
	144	48	54
128	128	64	68
	192	64	69
	256	64	72

For SIMON$2n/mn$, the key-dependent SIMON$2n/mn$ round function is the map R_{k_i}:$GF(2)^n \times GF(2)^n \to GF(2)^n \times GF(2)^n$ defined by

$$R_{k_i}(x_{i+1}, x_i) = (x_i \oplus f(x_{i+1}) \oplus k_i, x_{i+1}),$$

where $f(x_{i+1}) = \left(S^1 x_{i+1} \& S^8 x_{i+1}\right) \oplus S^2 x_{i+1}$, $k_i(k_i \in GF(2)^n)$ is the round subkey. This round function is pictured in Fig. 1.

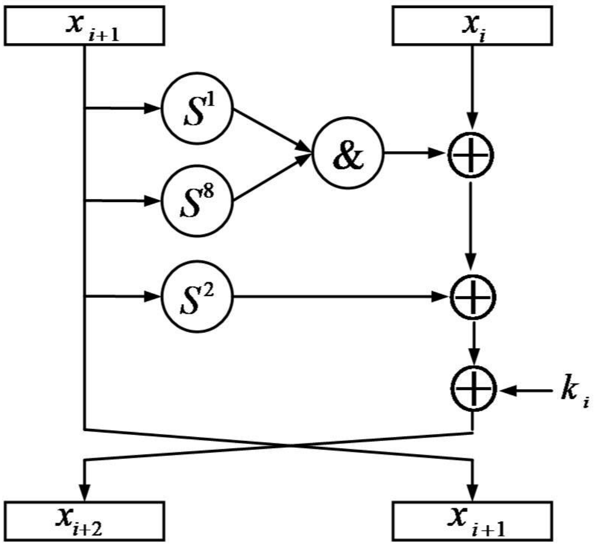


Fig. 1: Round function of SIMON

As it is out of scope for our purpose, we refer to [4] for the description of the key-scheduling.

Since the SIMON algorithm was proposed, cryptographers have analyzed its various versions. The earliest differential attack on SIMON was presented by Farzaneh Abed *et al.* in [18], which attacked 18-round SIMON32, 19-round SIMON48 and 26-round SIMON64. They described attacks on up to slightly more than half the number of rounds. Unfortunately, its complexity is high if we use their methods to attack low-round SIMON. At the FSE2014, Alex Biryukov *et al.* [19] applied automatic search for differential trails and attacked SIMON. At the same time, Wang Q *et al.* [20] studied the security of SIMON by using integral, zero-correlation linear and impossible differential cryptanalysis. In addition, Sun S *et al.* [21] got a better differential distinguisher by mixed integer linear programming (MILP). What's more, Raddum [22] gave efficient algebraic attacks on up to 16 rounds of the largest SIMON variants. Although there are many attacks, the complexity of their attack is very high. Further, if we want to use their methods for practical attacks, the number of rounds we can attack is low.

2.3 SAT-Based search for the differential characteristics on SIMON

SAT is the Boolean Satisfiability Problem. It is an NPC problem and considers whether there is a valid assignment to Boolean variables satisfying a given set of Boolean conditions. As the key issue of computer science and artificial intelligence, SAT solver has gained a lot of attention since it was proposed. It has great advantages with the open source, good interface, high efficiency and perfect compatibility. There are many cryptanalysis results based on SAT. In 2013, Mouha *et al.* [23] proposed a tool for finding optimal differential characteristics of ARX ciphers, where the tool can transform the problem of finding optimal differential characteristics of ARX cipher Salsa20 [24] to SAT model which can be solved by SAT/SMT solver. Later, Song *et al.* [5] utilized Mouha *et al.*'s framework to find differential characteristics of SPECK and LEA by adding a new method for constructing long characteristics from short. In CRYPTO 2015, Stefan *et al.* [6] derived efficiently computable and easily implementable expressions for the exact differential behaviour of SIMON-like round functions. At the same time, they used those expressions for a computer aided approach based on SAT/SMT solvers to find both optimal differential characteristics for SIMON. In this section, we mainly introduce those expressions. Stefan *et al.* gave a full formula for differentials as follows:

Theorem 1 [6] *Let $f(x) = S^a(x) \& S^b(x) \oplus S^c(x)$, where $gcd(n, a-b) = 1$, n even, $\vee$ denotes the bitwise OR operation, $\overline{x}$ denotes the bitwise negation of x, $wt(x)$ denotes the Hamming weight of x, and $a > b$ and let α and β be an input and output difference. Then with*

$$\begin{cases} varibits = S^a(\alpha) \vee S^b(\alpha) \\ doublebits = S^b(\alpha) \& \overline{S^a(\alpha)} \& S^{2a-b}(\alpha) \\ \gamma = \beta \oplus S^c(\alpha) \end{cases} \tag{1}$$

And the probability is given in (2) that difference α goes to difference β.

$$P(\alpha \to \beta) = \begin{cases} 2^{-n+1} & if\ \alpha = 1 and\ wt(\gamma) \equiv 0 mod 2 \\ 2^{-wt(varibits \oplus doublebits)} & if\ \alpha \neq 1 and\ \gamma\ \& \overline{varibits} = 0 \\ & and\ \left(\gamma \oplus S^{a-b}(\gamma)\right) \& doublebits = 0 \\ 0 & else \end{cases} \tag{2}$$

More precisely, the round function in SIMON corresponds to $(a, b, c) = (8, 1, 2)$. With Stefan's expressions, we can construct Algorithm 1 to search for optimal probability and optimal differential characteristics. The M refers to a SAT model of R-round differential characteristic of $SIMON2n/mn$ in Algorithm 1.

Algorithm 1 Search for optimal probability and optimal differential characteristics based on SAT

```
Input: Rounds $R$, Cipher SIMON$2n/mn$, $M$
Output: Optimal probability, Optimal differential characteristics.
 1: Path=[ ]
 2: flag=0
 3: For $p$ in range$(1, 2n)$ do#Search for optimal R-round probability
 4:   Flag=SAT($R$,p,$M$) #Use a SAT solver to determine whether M has a solution under R and p. If there is a
      solution, return to 'sat'
 5:   IF Flag == sat THEN
 6:     flag=1
 7:     BREAK
 8:   END IF
 9: END For
10: WHILE ((flag==1) and (Flag==sat))do#Search for optimal R-round differential characteristics
11:   path,Flag=SATpath($R$,p,$M$,Path)#Add elements in Path to constraints. Solve feasible solutions
12:   Path=Path||path
13: END WHILE
14: RETURN (Path,p)
```

By Theorem 1, the search for differential characteristics is transformed into SAT/SMT model. We use the open source solver Z3 [25] to complete the search for the differential characteristics. Using Z3, if the SAT/SMT model has a solution, it will return 'sat' and the solution model can be accessed to obtain the value of the solution; if the problem has no solution, it will return 'unsat', indicating that there is no solution under the current constraints. Algorithm 1 gives a special case of searching for differential characteristics, that is, searching for the optimal differential characteristics. By Theorem 1, the search for other differential characteristics with lower probability can also be completed. It is similar to Algorithm 1, so we don't describe it in detail.

2.4 Brief Description of Neural Differential Distinguisher for SIMON

Gohr convert the distinguisher of ciphertext pairs into a binary classification problem. His method is not only applicable to Speck, but also applicable to SIMON. With his method, we can construct a model of neural differential distinguisher for SIMON, which has not been trained. The method of constructing the data has been explained in [26]. As it is out of scope for our purpose, we refer to [26] for the description of the method of constructing the data.

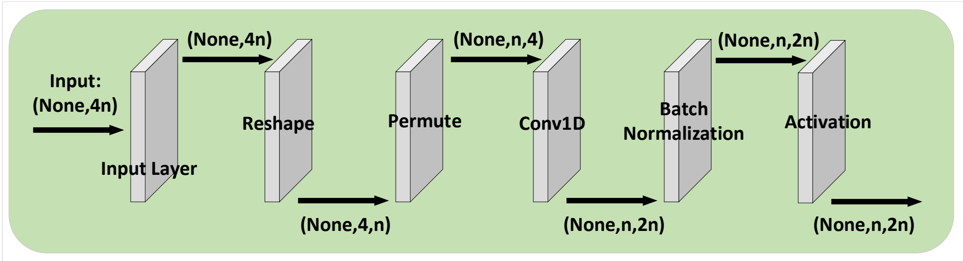
There are multiple neural networks available to train neural distinguishers, such as MIP, ResNet and so on. We choose the ResNet to help train a neural distinguisher. Our networks comprise three main components: input layer, iteration layer and predict layer. The diagram for our network is shown in Fig. 2. The n in Fig. 2 refers to the word size of SIMON$2n/mn$. The input layer receives training data with fixed length and applies reshape layer into the data. After reshape layer, we wish to permute input data. At the same time, we transpose and apply Conv1D into input data followed by batch normalization layer and activation layer so that we can expend the effect of each bit. After activation layer, data will be sent into iteration layer.

The iteration layer is built by classical residual learning framework and shown in Fig. 2(b). In each residual block, we use two Conv1D layers, and each Conv1D layers is followed by a batch normalization layer and a activation layer. Each Conv1D consists of $2n$ filters. The identity shortcuts can be directly used since the input of the first Conv1D layer and output of the second activation layer are of the same dimensions. At the end of the second activation layer, a skip connection then adds the output of the layer to the input of the first Conv1D layer and passes the result to the next residual block. Iteration layer will repeat 5 residual blocks in our experiments, and then the output layer will be following.

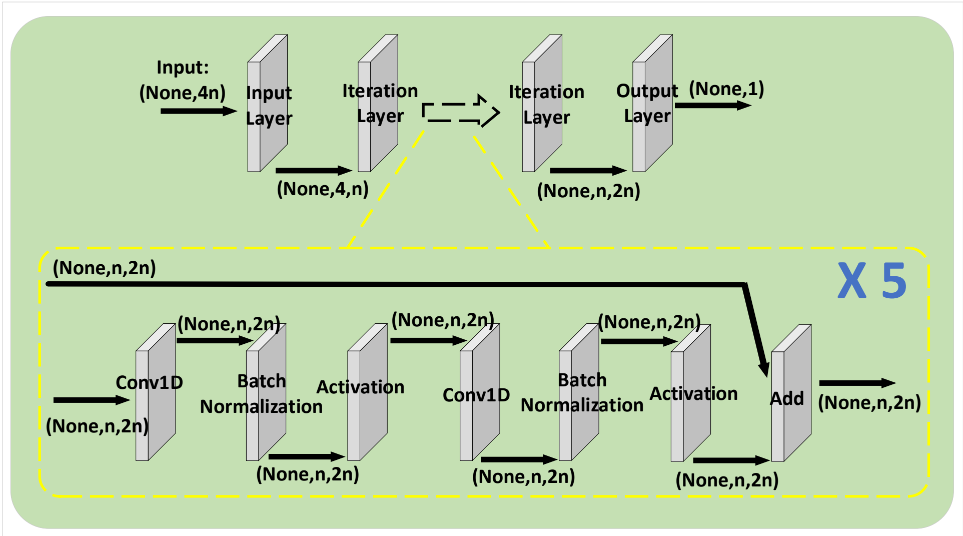
After flattening data from iteration layer, data will be sent into a fully connected layer. The fully connected layer consists of a hidden layer and a output unit. The first layer are Dense layers with $4n$ units, and followed by a batch normalization layer and a activation layer. The final layer consists of a single output unit using a sigmoid activation.

In our network, we choose that the kernel size of the first Conv1D layer is 1 and the kernel size of other Conv1D layer is 3. In addition, the number of filters in each convolutional layer is $2n$ and the padding method is 'same'. At last, we train our network based on L2 weights regularization to avoid overfitting. The other details of the hyper-parameters used are given in Table 4.

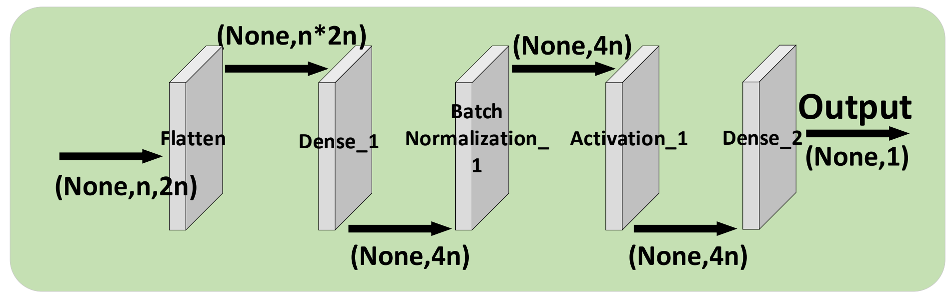
After the neural differential distinguisher is trained, we can use it to distinguish the output of SIMON with a given input difference from random data, if its accuracy is more than 51% on the test set. The higher its accuracy on the test set, the better it is to distinguish ciphertext data. In [26], Hou *et al.* investigate the influence of input difference on the accuracy of neural distinguisher. He explores the effect of two kinds of input difference on accuracy of neural distinguishers. But, unfortunately, he does not give an algorithm for finding the input difference to search for the high-accuracy neural differential distinguishers.



(a) Input Layer



(b) Iteration Layer



(c) Output Layer

Fig. 2: Network Architecture

Table 4: List of Hyper-parameters

Hyper-Parameters	Value
Batch Size	10000
Epochs	100
Train size	10^7
Validation size	10^5
Regularization parameter	10^{-4}
Optimizer	Adam
Loss function	MSE(mean-squared-error)

3 Search for Neural Differential Distinguisher Based on SAT

In traditional differential cryptanalysis, it is a primary task to find a high-probability differential characteristic, which takes advantage of the unevenness of the differential distribution. The distribution of output difference is different for different input difference. For neural differential distinguisher, it is actually to learn the distribution of output difference given a fixed input difference. Therefore, the input difference directly affects the accuracy of the neural differential distinguisher.

In this section, we will introduce our basic algorithm for search for high-accuracy neural differential distinguishers by searching for the optimal differential characteristics. With the basic algorithm, we search for the high-accuracy neural differential distinguishers for 9-round SIMON48/96. In addition, we compare the accuracy of the neural differential distinguisher obtained by using the input difference of the existing differential characteristic and our algorithm. Considering that some distinguishers are also effective, even if they are trained by input difference from some high-probability differential characteristics, instead of the optimal differential characteristics, we improve the basic algorithm to search for more high-accuracy neural distinguishers. We expand the search space of the distinguishers by expanding the probability space to $\left[2^{-\frac{n}{2}} \times P_{\max}, P_{\max}\right]$, where $P_{\max}$ is the optimal probability and n is the word size of SIMON. Using the improved algorithm, we search for the high-accuracy neural differential distinguishers for 9-round SIMON32/64, 10-round SIMON48/96 and 11-round SIMON64/128.

3.1 Basic algorithm for search for high-accuracy neural differential distinguisher

At present, the input difference of the neural differential distinguisher comes from the existing optimal differential characteristics, which are used in traditional differential cryptoanalysis. In [17], Yi Chen *et al.* choosed the input difference from [18] to train neural differential distinguisher of Speck32/64. However, these differential characteristics are not dedicated to neural differential distinguishers, so the neural differential distinguisher with high accuracy for the fixed rounds cannot be obtained by these input difference from differential characteristics.

Taking into account the unevenness of the distribution of out difference for different input difference, we decide to use the input difference of optimal differential characteristic as the candidate difference. We search for optimal differential characteristic by SAT-based automatic search, and train networks with these input difference of differential characteristics. We design an algorithm to help us search for neural differential distinguisher with higher accuracy, which is shown in Algorithm 2.

Algorithm 2 Search for neural differential distinguisher based on SAT

```
Input: Network Architecture $Net$, Cipher $C$ ($SIMON2n/mn$), Round $R$
Output: Neural differential distinguisher $ND$, Input difference of distinguisher $I_d$
1: Use Algorithm 1 to search for the optimal differential characteristics of $C$, and save their input difference as
   $DIFF$
2: $ND = [\,]$
3: $I_d = [\,]$
4: For $d$ in $DIFF$ do
5:     S = $C$(d,R) #Use d to generate data set
6:     V = $C$(d,R) #Use d to generate test set
7:     $D_d = Net$(S)#Pre-training using Net and S
8:     $acc_d$ = Evaluate($D_d$,V)#Get the accuracy of the model $D_d$
9:     IF $acc_d > 0.51$ THEN
10:        $ND = ND \mid\mid D_d$
11:        $I_d = I_d \mid\mid d$
12:    END IF
13: END For
14: RETURN ($ND$,$I_d$)
```

The network architecture Net refers to Fig. 2. With Algorithm 2, we can search for a neural differential distinguisher, which can distinguish random data and ciphertext pairs. At the same time, in order to save the time of searching distinguisher, we use the hyper-parameters in Table 5 for training and complete quick pre-training by reducing the number of training iterations. In Algorithm 2, we use d as input difference to train neural distinguisher D_d based on Sect. 2.4. After this, we evaluate D_d by Keras [27] to filter input difference. It is regarded as a better input difference if the accuracy of neural differential distinguisher exceeds 51%, which is effective for key recovery.

Table 5: List of Hyper-parameters for Algorithm 2

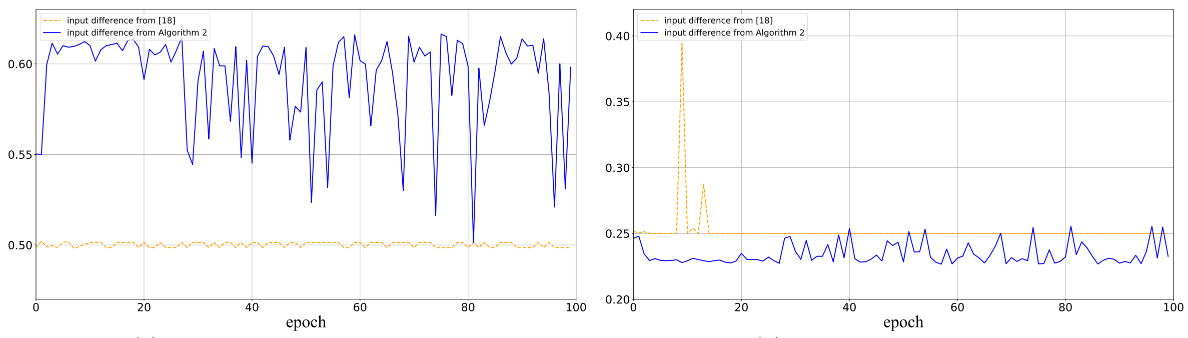
Hyper-Parameters	Value
Batch Size	10000
Epochs	10
Train size	10^7
Validation size	10^5
Regularization parameter	10^{-4}
Optimizer	Adam
Loss function	MSE(mean-squared-error)

It is found that the highest probability is 2^{-24} for 9-round SIMON48/96 by SAT. With the probability of 2^{-24}, we have a total of 96 input difference using SAT, which come from 9-round different differential characteristics. The input difference and corresponding accuracy obtained by our search are shown in Table 6. As we can see, we get 48 neural differential distinguishers with accuracy over 60%, where the highest accuracy is more than 61%.

Table 6: 9-round Input difference of SIMON48/96 with Algorithm 2

Input difference	Accuracy	Input difference	Accuracy	Input difference	Accuracy
(0x40000,0x110000)	0.61598998	(0x2000,0x8800)	0.60943002	(0x8000,0x22000)	0.60773998
(0x1000,0x4400)	0.58849001	(0x400000,0x100001)	0.58247	(0x1,0x400004)	0.58160001
(0x10000,0x44000)	0.58117002	(0x2,0x800008)	0.57898003	(0x800,0x2200)	0.57729
(0x80000,0x220000)	0.57688999	(0x200000,0x880000)	0.57665998	(0x20,0x88)	0.57538998
(0x400,0x1100)	0.57481003	(0x20000,0x88000)	0.57446003	(0x40,0x110)	0.57358998
(0x80,0x220)	0.57339001	(0x100,0x440)	0.57247001	(0x200,0x880)	0.57244003
(0x4000,0x11000)	0.57125002	(0x100000,0x440000)	0.57077003	(0x10,0x44)	0.56976998
(0x8,0x22)	0.56953001	(0x800000,0x200002)	0.56946999	(0x4,0x11)	0.56870002
(0x440000,0x1)	0.53143001	(0x22000,0x80000)	0.52972001	(0x11000,0x40000)	0.52842999
(0x200002,0x8)	0.52833998	(0x220,0x800)	0.52802002	(0x100001,0x4)	0.52728999
(0x110000,0x400000)	0.52667999	(0x220000,0x800000)	0.52667999	(0x110,0x400)	0.52596998
(0x4400,0x10000)	0.52525997	(0x400004,0x10)	0.52519	(0x88,0x200)	0.52478999
(0x88000,0x200000)	0.52468997	(0x800008,0x20)	0.52456999	(0x2200,0x8000)	0.52432001
(0x880,0x2000)	0.52342999	(0x11,0x40)	0.52337003	(0x1100,0x4000)	0.52332997
(0x880000,0x2)	0.52280003	(0x440,0x1000)	0.52270001	(0x22,0x80)	0.52197999
(0x8800,0x20000)	0.52187002	(0x44,0x100)	0.52144003	(0x44000,0x100000)	0.52140999
(0x200020,0x80088)	0.50421	(0x8880,0x20200)	0.50367999	(0x444,0x1010)	0.50345999
(0x440004,0x100010)	0.50344002	(0x80800,0x220200)	0.50329	(0x88800,0x202000)	0.50326997
(0x8080,0x22020)	0.50325	(0x808,0x2202)	0.50307	(0x2020,0x8808)	0.50297999
(0x880008,0x200020)	0.50292999	(0x888000,0x20002)	0.50287002	(0x10100,0x44040)	0.50265998
(0x110001,0x40004)	0.50261998	(0x404000,0x101001)	0.50261003	(0x111,0x404)	0.50260001
(0x11100,0x40400)	0.50248998	(0x20200,0x88080)	0.50247997	(0x404,0x1101)	0.50234002
(0x800088,0x202)	0.50230998	(0x222000,0x808000)	0.50226003	(0x800080,0x200220)	0.50222999
(0x40400,0x110100)	0.50208002	(0x20002,0x808008)	0.50199997	(0x444000,0x10001)	0.50199002
(0x1010,0x4404)	0.50193	(0x220002,0x80008)	0.50187999	(0x22200,0x80800)	0.50185001
(0x10001,0x404004)	0.50185001	(0x1110,0x4040)	0.50182998	(0x44400,0x101000)	0.50165999
(0x111000,0x404000)	0.50164998	(0x101000,0x440400)	0.50159001	(0x202,0x800880)	0.50156999
(0x222,0x808)	0.50155997	(0x888,0x2020)	0.50151998	(0x400044,0x101)	0.50150001
(0x400040,0x100110)	0.50147998	(0x200022,0x800080)	0.50142998	(0x808000,0x202002)	0.50138003
(0x4040,0x11010)	0.50129998	(0x100011,0x400040)	0.50125998	(0x80008,0x20022)	0.50111997
(0x101,0x400440)	0.50107998	(0x2220,0x8080)	0.50094998	(0x202000,0x880800)	0.50089997
(0x100010,0x40044)	0.50055999	(0x4440,0x10100)	0.50044	(0x40004,0x10011)	0.50041997

In order to show that Algorithm 2 is effective for searching for distinguishers with distinguishing effect, we use the input difference of the optimal differential characteristic in [18] to train the neural differential distinguisher with the same rounds. For SIMON48/96, we choose $(0x10100, 0x44040)$ in [18] as input difference to train 9-round neural distinguisher. The comparison between the input difference in Algorithm 2 and in [18] is shown in Fig. 3. Therefore, Algorithm 2 is effective in searching for neural differential distinguishers. Although both methods select the best characteristics, Algorithm 2 selects the rounds of the differential characteristics according to the rounds of neural differential distinguisher, however, the rounds of the existing optimal differential characteristics are longer than that of neural differential distinguisher. In other words, input difference from the existing optimal differential characteristics is not suitable for lower-round neural differential distinguisher.



(a) Validation accuracy by epoch (b) Validation loss by epoch

Fig. 3: Comparison between the input difference from Algorithm 2 and from [18]

Table 7: Comparison between using input difference in Algorithm 2 and [18]

Block Cipher	Source of input difference	Round	Input difference	Accuracy
SIMON48/96	[18]	9	(0x10100,0x44040)	50.22%
	Algorithm 2	9	(0x40000,0x110000)	61.60%

Similarly, we also try to use Algorithm 2 to construct neural distinguishers of SIMON48/96 with more rounds. We have a total of 48 input difference shown in Table 8, which come from ten-round different differential characteristics. With the hight accuracy, we choose $(0x111000, 0x404000)$ to train 10-round neural distinguishers, which is shown in Fig. 4. With accuracy over 50.1%, the ten-round distinguisher is effective to distinguish the output of SIMON48/96 with a given input difference from random data. But, unfortunately, the accuracy is not high enough to complete the key recovery. With the comparison of Table 6 and Table 8, we can see that the accuracy of the 10-round distinguisher is lower than the accuracy of the 9-round distinguisher. This shows that it is more difficult to search for high-accuracy neural distinguishers as the rounds increases. This is related to the weaker and weaker non-random features of the ciphertext pairs as the rounds increases.

Table 8: 10-round Input difference of SIMON48/96 with Algorithm 2

Input difference	Accuracy	Input difference	Accuracy	Input difference	Accuracy
(0x200,0x888)	0.50138998	(0x222,0x808)	0.50040001	(0x440004,0x100010)	0.50282001
(0x111000,0x404000)	0.50476003	(0x444000,0x10001)	0.50373	(0x222000,0x808000)	0.50134999
(0x22200,0x80800)	0.50221997	(0x8880,0x20200)	0.50119001	(0x888000,0x20002)	0.50145
(0x11100,0x40400)	0.50388002	(0x2220,0x8080)	0.50050002	(0x444,0x1010)	0.50121999
(0x880008,0x200020)	0.50186998	(0x220002,0x80008)	0.50277001	(0x8000,0x22200)	0.50234997
(0x200022,0x800080)	0.50199997	(0x110001,0x40004)	0.50186002	(0x40,0x111)	0.50303
(0x20000,0x88800)	0.50208002	(0x2000,0x8880)	0.50193	(0x4000,0x11100)	0.50182003
(0x111,0x404)	0.50283998	(0x20,0x800088)	0.50111997	(0x2,0x880008)	0.50285
(0x800000,0x220002)	0.50111002	(0x200000,0x888000)	0.50077999	(0x888,0x2020)	0.50146002
(0x100,0x444)	0.50186998	(0x800088,0x202)	0.50106001	(0x10000,0x44400)	0.50095999
(0x44400,0x101000)	0.50285	(0x40000,0x111000)	0.50351	(0x100011,0x400040)	0.50136
(0x80,0x222)	0.50139999	(0x80000,0x222000)	0.50213999	(0x400044,0x101)	0.50246
(0x88800,0x202000)	0.50186002	(0x1110,0x4040)	0.50193	(0x1,0x440004)	0.50256997
(0x4440,0x10100)	0.50173998	(0x400000,0x110001)	0.50203001	(0x1000,0x4440)	0.50244999
(0x800,0x2220)	0.50322002	(0x8,0x200022)	0.50234997	(0x4,0x100011)	0.50209999
(0x400,0x1110)	0.50410002	(0x10,0x400044)	0.50117999	(0x100000,0x444000)	0.50111997

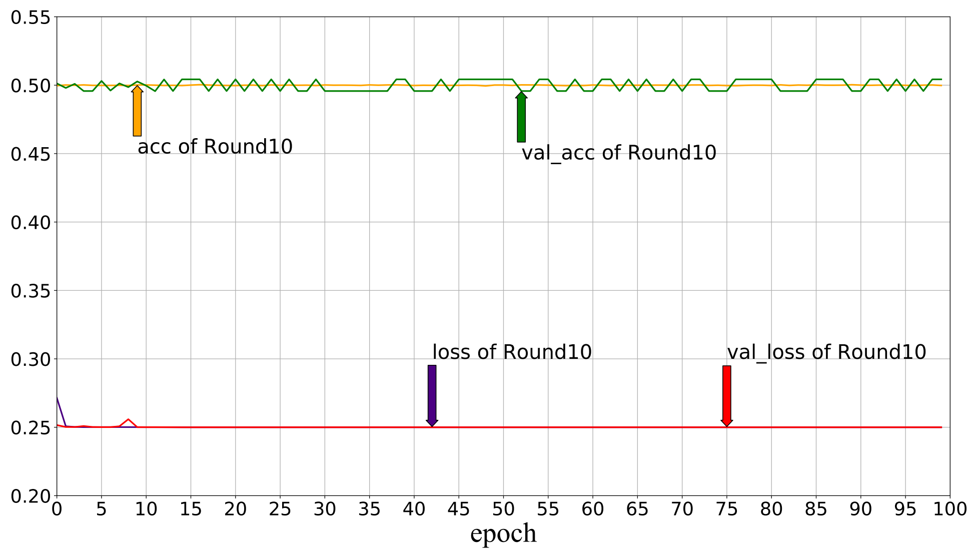


Fig. 4: 10-round neural differential distinguisher of SIMON48/96 with Algorithm 2

3.2 Improved algorithm for search for high-accuracy neural differential distinguisher

In our experiment, we find that some distinguishers are also effective, even if they are trained by input difference from some high-probability differential characteristics, instead of the optimal differential characteristics for fixed rounds. In other words, the accuracy from input difference of high-probability differential characteristics may be higher than the distinguisher from the optimal differential characteristics. In order to find a higher-accuracy distinguisher, we improved Algorithm 2 to Algorithm 3.

Algorithm 3 Improved search for neural differential distinguisher based on SAT

```
Input: Network Architecture $Net$, Cipher $C$ ($SIMON2n/mn$), Round $R$
Output: Neural differential distinguisher $ND$, Input difference of distinguisher $I_d$
 1: Search for the optimal probability as $P_{max}$
 2: Search for the differential characteristics with probability in $\left[2^{-\frac{n}{2}} \times P_{\max}, P_{\max}\right]$, and save their input difference
    as $DIFF$
 3: $ND = [\,]$
 4: $I_d = [\,]$
 5: For $d$ in $DIFF$ do
 6:    S = $C$(d,R) #Use d to generate data set
 7:    V = $C$(d,R) #Use d to generate test set
 8:    $D_d = Net$(S)#Pre-training using Net and S
 9:    $acc_d$ = Evaluate($D_d$,V)#Get the accuracy of the model $D_d$
10:    IF $acc_d > 0.51$ THEN
11:       $ND = ND \mid\mid D_d$
12:       $I_d = I_d \mid\mid d$
13:    END IF
14: END For
15: RETURN ($ND$,$I_d$)
```

In Algorithm 3, we expand the search space of input difference by expanding the range of the probability. We choose $2^{-\frac{n}{2}} \times P_{\max}$ as the lower bound of the probability, which is from experience. If differential probability is lower than $2^{-\frac{n}{2}} \times P_{\max}$, there is almost no neural differential distinguisher with distinguishing effect. Although Algorithm 3 increases the time, it expands the search space when the time allows, and it will be possible to search for a neural distinguishers with higher accuracy. By Algorithm 3, we searched for 10-round neural distinguisher of SIMON48/96 with accuracy about 57.89%, which shows that the improved algorithm is more effective in searching for high-accuracy neural distinguisher than the basic algorithm.

With Algorithm 3, we get the neural differential distinguishers of SIMON32/64, SIMON48/96 and SIMON64/128 in 9, 10 and 11 rounds respectively. This is the first time that there is a neural distinguisher of 11-round SIMON64/128. The results of the neural differential distinguishers are shown in Table 9.

Table 9: Results of neural differential distinguishers with three methods

Block Cipher	Source of neural differential distinguisher	Round	Input difference	Accuracy
SIMON32/64	[18]	9	(0x0,0x40)	59.07%
	Algorithm 3	9	(0x0,0x80)	59.77%
SIMON48/96	[18]	9	(0x10100,0x44040)	50.22%
	Algorithm 2	9	(0x40000,0x110000)	61.60%
	Algorithm 3	10	(0x0,0x100000)	57.89%
SIMON64/128	[18]	10	(0x100,0x440)	58.61%
	Algorithm 3	11	(0x0,0x10)	59.72%

In Table 9, we show the comparison of the accuracy from three methods of selecting the input difference. In Table 9, we select the high-accuracy neural differential distinguishers and their input difference obtained under three methods of searching for neural differential distinguishers. As we can see, compared with selecting the input difference in [18] and Algorithm 2, the neural distinguisher obtained by Algorithm 3 has higher accuracy. For SIMON32/64, the accuracy of input difference from Algorithm 3 is higher than from [18]. For SIMON48/96 and SIMON64/128, we get longer-round neural differential distinguishers, which helps us perform longer-round key recovery attack.

We also try to search for neural differential distinguisher with longer rounds. Unfortunately, as the rounds increases, the non-random features of the ciphertext pairs become weaker and weaker. So it is difficult for us to find a neural differential distinguisher with a higher round, even if using Algorithm 3. In addition, the higher the Hamming weight of the input difference, the more likely the output difference is to be evenly distributed in traditional differential cryptanalysis. So we can first search for input differences with lower Hamming weight for Algorithm 3, if time limit.

4 Practical key recovery attack of round-reduced SIMON

Using Algorithm 3, we choose $(0x0, 0x100000)$ as the input difference to train 9-round and 10-round neural differential distinguisher. With the automatic searches based on SAT, we extend our 9-round and 10-round distinguisher to a 11-round and 12-round distinguisher by prepending the 2-round differential characteristic $(0x400000, 0x100001) \xrightarrow{2^{-4}} (0x0, 0x100000)$. Using 11-round and 12-round distinguisher, we complete practical 14-round key recovery attack of SIMON48/96 on a workstation configured with *Intel i9-10900K* and *Nvidia TITAN RTX*. It takes about 1550s to recover the final subkey each time. Our attack only needs no more than $2^{22.21}$ 14-round encryption and the data complexity does not exceed $2^{12.8}$. With the traditional differential attack in [18], it requires 2^{35} plaintext pairs at least with the optimal 12-round optimal probability 2^{35} [28]. In addition, we perform key recovery attack of 13-round SIMON32/64 with a success rate of more than 93%. Therefore the data complexity and time complexity based on deep learning is far lower than that of the traditional differential cryptanalysis.

4.1 Gohr's Attack Scheme

The R-round neural differential distinguisher is applied to a key recovery attack on $(R+1)$-round Speck32/64. In attack, using the R-round neural differential distinguisher, the key candidate can be scored. A key guess is returned if its score exceeds threshold t. The key rank score is formulated as

$$v_k = \sum_{i=1}^{n} \log_2 \left(\frac{Z_i^k}{1 - Z_i^k} \right),$$

where k is the key candidate, and Z_i^k is the i_{th} $(R+1)$-round ciphertext pair's output signal given by the R-round neural differential distinguisher.

Since the time of scoring all k $(0 \leqslant k < 2^{16})$ and selecting the key with the largest score is huge, Gohr search for high-scoring candidate keys by iterative method. In order to generate new candidate keys in iterations, Gohr exploited the uneven distribution of the output signal corresponding to the wrong key. For $(R+1)$-round Speck32/64, let C_0 and $C_0^{'}$ are a pair of $(R+1)$-round ciphertext whose input difference is α, and k is the real $(R+1)_{th}$ subkey. For $\delta \in GF(2)^{16}$, there is $k^{'} = k \oplus \delta$. Use $k^{'}$ as a subkey to decrypt the ciphertext pair, and get the output signals of R-round neural distinguisher as

$$R_\delta \left(C_0, C_0^{'} \right) = N_R \left(E_{k^{'}}^{-1} \left(C_0, C_0^{'} \right) \right).$$

Then $R_\delta \left(C_0, C_0^{'} \right)$ can be regarded as a random variable related to δ, which follows a normal distribution with mean μ_δ and standard deviation σ_δ. Using the difference in the distribution of the wrong key, the guessed key can be generated. The iteration ends, if the score of a candidate key exceeds threshold t.

In the attack scheme above, Gohr's attack is shown as Algorithm 4.

Algorithm 4 Gohr's Key Research for $(R+1)$-round Speck32/64

```
Input: Ciphertext pairs set $C = \{C_0, C_1, ..., C_{n-1}\}$, $R$-round neural differential distinguisher $D$, number of candidates to be generated $t$, number of iterations $l$
Output: Key set $L$
 1: $S \leftarrow \{k_0, k_1, ..., k_{t-1}\}$, $k_i \neq k_j$ if $i \neq j$;
 2: $L \leftarrow \{\}$;
 3: For $j \in \{0, 1, ..., l-1\}$ do
 4:     $P_{i,k} \leftarrow Decrypt_1(C_i, k)$ for all $i \in \{0, 1, ..., n-1\}$ and $k \in S$
 5:     $v_{i,k} \leftarrow D(P_{i,k})$ for all $i \in \{0, 1, ..., n-1\}$ and $k \in S$
 6:     $w_{i,k} \leftarrow \log_2\left(\frac{v_{i,k}}{1-v_{i,k}}\right)$ for all $i \in \{0, 1, ..., n-1\}$ and $k \in S$
 7:     $L \leftarrow L||[(k, \sum_{i=0}^{n-1} w_{i,k})$ for $k \in S]$
 8:     $m_k \leftarrow \sum_{i=0}^{n-1} w_{i,k}/n$ for $k \in S$
 9:     $\lambda_k \leftarrow \sum_{i=0}^{t-1}\left(\frac{m_{k_i} - \mu_{k_i \oplus k}}{\sigma_{k_i \oplus k}}\right)^2$ for $k \in \{0, 1, ..., 2^{24}-1\}$
10:     $S \leftarrow argsort_k(\lambda)[0 : t-1]$
11: end for;
12: return $L$;
```

In addition, Gohr found that the output signal from R-round neural differential distinguisher will be rather weak. He therefore boost it by using k neutral bits to create from each plaintext pair a plaintext structure consisting of 2^k plaintext pairs that are expected to pass R-round neural differential distinguisher. Given k ciphertext structures, his algorithm is first tried on each structure. Then select the ciphertext structure, that can enhance output signal from R-round neural differential distinguisher, to perform the key recovery.

4.2 Key recovery attack of 14-round SIMON48/96

4.2.1 Overview

With over 57.5% accuracy, we choose $(0x0, 0x100000)$ as the input difference and train 9-round and 10-round neural distinguisher of SIMON48/96. With the automatic searches based on SAT, we extend our neural 9-round distinguisher to 11-round and 12-round distinguishers by prepending the 2-round differential characteristic $(0x400000, 0x100001) \xrightarrow{2^{-4}} (0x0, 0x100000)$. Due to the round function of the SIMON48/96, the key-addition occurs after the non-linear operation, we can construct 14-round key-recovery attack by adding 1 round before and after 12-round distinguisher.

Attack Pattern If the final subkey is correctly guessed, the probability that the intermediate state obtained by the correct last subkey and the correct second-to-last subkey passs through the 11-round distinguisher is the highest. That is to say, the response of the 11-round distinguisher is the highest if 2-round subkey are guessed correctly. This is due to the fact that the intermediate state decrypted by the error key is a random sequence for distinguisher given by the fixed difference, that is, the distinguisher will returns a value of approximately 0.5 if the guessed key is wrong. This can help us develop the key recovery attack of 14-round SIMON48/96. We guess possible keys in the last round, we use the guessed subkey to perform 1-round decryption, and use 12-round distinguisher to score and sort the guessed subkeys. If the score of a key exceeds the threshold $t1$, we use the key to decrypt one round. At the same time, guess the second-to-last subkey, and similarly, score and sort them. If the score exceeds the threshold $t2$, the last guessed subkey will be returned as the result.

We use Gohr's attack scheme to perform a practical key recovery attack on 14-round SIMON48/96. Our attack parameters are shown in Table 10.

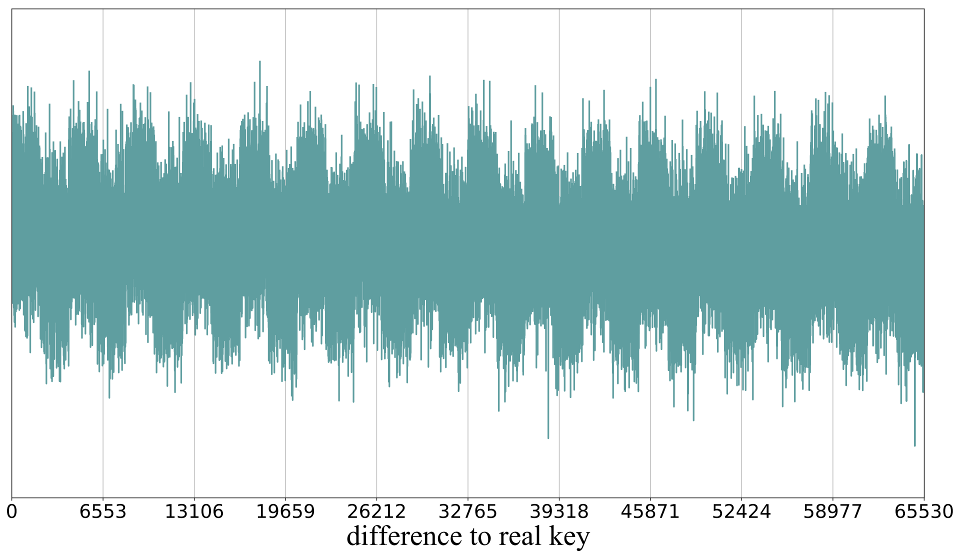
Table 10: Attack parameters for 14-round SIMON48/96

Parameter	Value
Initial difference	(0x400000,0x110001)
$t1$	20
$t2$	35
Neutral bit	[44,47,21,39,3,28]
Number of iterations	60
Experimental configuration	*Intel i9-10900K*, *Nvidia TITAN RTX*

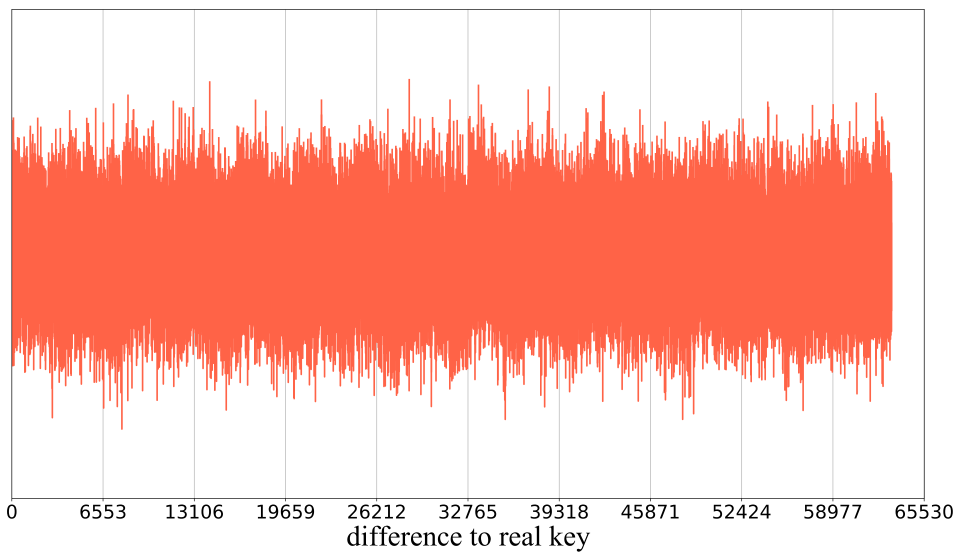
4.2.2 Wrong Key Randomization of the 10-round neural distinguisher

Let C_0 and $C_0^{'}$ are a pair of ciphertext whose input difference is $(0x0, 0x100000)$, and k is the real subkey of the last round. For $\delta \in GF(2)^{24}$, there is $k^{'} = k \oplus \delta$. Use $k^{'}$ as a subkey to decrypt the ciphertext pair, and get the response as $R_\delta\left(C_0, C_0^{'}\right) = N_{10}\left(E_{k^{'}}^{-1}\left(C_0, C_0^{'}\right)\right)$, where N_{10} is the 10-round neural distinguisher. Then $R_\delta\left(C_0, C_0^{'}\right)$ can be regarded as a random variable related to δ. We can assume that $R_\delta\left(C_0, C_0^{'}\right)$ follows a normal distribution with mean μ_δ and standard deviation σ_δ.

In order to compare the influence of wrong keys on the response of the neural distinguisher, we calculated the wrong key response profile for our 10-round distinguishers for SIMON48/96. For $\delta \in GF(2)^{24}$ and a random master key K, we generate 1 ciphertext pair $\left(C_0, C_0^{'}\right)$ whose input difference is $(0x0, 0x100000)$ and save its last real subkey rk. Calculate $k^{'} = rk \oplus \delta$ and use $k^{'}$ to decrypt $\left(C_0, C_0^{'}\right)$ for one round, we get the response value of the 10-round distinguisher. We repeat the above steps 2^{24} times and calculate the mean value and standard deviation of the response values. The part of wrong key response profile for ten rounds is shown in Fig.5. A lot of non-random structure is evident. The shape of the curves for σ_δ and for eight rounds is similar.



(a) Mean response for 10-round distinguisher



(b) Standard deviation for 10-round distinguisher

Fig. 5: Wrong key response profile for 10-round distinguisher

It can be seen from the figure above that, the mean value and standard deviation are larger when there are fewer error bits. This is a similar distribution for standard deviation σ_δ. This regular change helps recover the last key. Similar to 10-round neural distinguisher, we can get wrong key response profile for 9-round neural distinguisher.

4.2.3 Result and complexity analysis

Our attack only needs no more than 1550s each time, which does not exceed $2^{22.21}$ 14-round encryption. At the same time, the data complexity does not exceed $2^{12.8}$. With complexity analysis, we can see that this is a practical attack. Our result in 100 trials is shown in Table 11.

Table 11: Result of 14-round SIMON48/96

The number of error bits	0-2	3-5	6-8	9-11
Number of experiments	12	15	8	24

To show that the neural differential distinguisher has advantages in key recovery attacks, we use the traditional cryptanalysis to perform the recovery attacks for last key on 14-round SIMON48/96. Due to the round function of the SIMON48/96, the key-addition occurs after the non-linear operation, we can construct 14-round attack by adding one round before and after 12-round optimal differential characteristics with probability 2^{-35} [28]. With the traditional method, the time complexity is more than 2^{35} 14-round encryption, and the data complexity is 2^{35}.

4.3 Key recovery attack of 13-round SIMON32/64

Similar to the key recovery attack of 14-round SIMON48/96, we choose $(0x0, 0x80)$ as the input difference to train 8-round and 9-round neural differential distinguisher. With the automatic searches based on SAT, we extend our 8-round and 9-round distinguisher to a 10-round and 11-round distinguisher by prepending the 2-round differential characteristic $(0x200, 0x880) \xrightarrow{2^{-4}} (0x0, 0x80)$. Using 10-round and 11-round distinguisher, we complete practical 13-round key recovery attack of SIMON32/64. With the help of Gohr'method, the key recovery attack using our neural distinguisher is performed 100 times. Our attack parameters are shown in Table 12. In one hundred trials, the last subkey is correctly guessed in 60 cases and there are 24 cases that the guess for the last subkey was incorrect in one bit. There are only 9 cases where there is a difference of 2 bits from the correct last subkey. It takes about 23s each time to recover the final subkey, with a success rate of more than 90%. Our attack only needs no more than $2^{16.4}$ 13-round encryption and the data complexity does not exceed $2^{12.5}$.

Table 12: Attack parameters for 13-round SIMON32/64

Parameter	Value
Initial difference	$(0x200, 0x880)$
$t1$	10
$t2$	10
Neutral bit	[21, 30, 26, 5, 14, 18]
Number of iterations	100
Experimental configuration	*Intel i9-10900K*, *Nvidia TITAN RTX*

Using our algorithm, we choose input differences to train neural distinguishers. With our neural distinguishers, we perform key recovery attack on round-reduced SIMON32/64 and SIMON48/96. By comparing traditional differential cryptanalysis with differential cryptanalysis based on deep learning, we find that differential cryptanalysis based on deep learning have advantages in key recovery attacks. In traditional differential cryptanalysis, the traditional distinguisher uses a differential characteristic or multiple characteristics with given input difference and output difference. In contrast, the neural distinguisher only uses given input difference, which considers more the output differences effect under the same input difference. This makes the neural distinguisher to be more powerful in differential cryptanalysis.

5 Conclusion and future work

This paper investigates how to search for a neural differential distinguisher and performs key recovery attack on 13-round SIMON32/64 and 14-round SIMON48/96. We design an algorithm based on SAT to help us search for a good input difference so that the accuracy of the neural distinguisher can be as high as possible. We search and obtain the neural differential distinguishers of 9-round SIMON32/64, 10-round SIMON48/96 and aa-round SIMON64/128. For SIMON48/96, we train 9-round and 10-round neural distinguisher with $(0x0, 0x100000)$ as the input difference. In addition, we search for the best 2-round characteristic $(0x400000, 0x100001) \xrightarrow{2^{-4}} (0x0, 0x100000)$ to extend our neural 9-round and 10-round distinguishers to a 11-round and 12-round distinguishers by prepending the 2-round characteristic. With Gohr's attack scheme, we complete practical 14-round key recovery attack. Our attack only needs no more than 1550s each time, which does not exceed $2^{22.21}$ 14-round encryption. At the same time, the data complexity does not exceed $2^{12.8}$.

Our complexity is lower than traditional differential cryptanalysis. Similar to the key recovery attack of 14-round SIMON48/96, we perform 13-round key recovery attack of SIMON32/64. Our attack only needs no more than 23s each time, which does not exceed $2^{16.4}$ 13-round encryption. At the same time, the data complexity does not exceed $2^{12.5}$. To our surprise, the success rate for 13-round SIMON32/64 is more than 90%. Due to the similarity between SIMON48/72 and SIMON48/96, our work is also used to perform key recovery attack of 14-round SIMON48/72.

It is found that the choice of neutral bit directly affects the success rate of key recovery attack. We will research how to quickly select neutral bits to improve the success rate of key recovery attack. At the same time, it time is huge to get wrong key response profile for long-size block cipher. We will explore how to attack long-size block cipher.

References

1. Eli Biham, Adi Shamir. (1991) Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology, 4, 3-72.
2. Hans Dobbertin, Vincent Rijmen, Aleksandra Sowa. (2004) Advanced Encryption Standard - AES. Springer-Verlag, Berlin Heidelberg.
3. A BogdanovL, R Knudsen, G Leander, et al. (2007) PRESENT: An Ultra-Lightweight Block Cipher. Proceedings of the CHES 2007, Vienna, Austria, 10-13 September, pp.450-466. Springer, Berlin.
4. Beaulieu R, Shors D, Smith J, et al. (2013) The SIMON and SPECK Families of Lightweight Block Ciphers. Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference, San Francisco, USA, 8-12 June, pp.1-6. IEEE, New Jersey.
5. Ling Song, Zhangjie Huang, Qianqian Yang. (2016) Automatic Differential Analysis of ARX Block Ciphers with Application to SPECK and LEA. Proceedings of the ACISP2016, Melbourne, Australia, 4-6 July, pp.379-394. Springer, Cham.
6. Stefan Kolbl, Gregor Leander, Tyge Tiessen. (2015) Observations on the SIMON Block Cipher Family. Proceedings of the CRYPTO2015, Santa Barbara, USA, 16-20 August, pp.161-185. Springer, Berlin.
7. McCulloch, W.S. and Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5, 115-133.
8. Rosenblatt, F. (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review, 65, 386-408.
9. Rumelhart D, Hinton G. and Williams, R. (1986) Learning representations by back-propagating errors. Nature, 323, 533-536.
10. Hinton G.E., Osindero Simon, and Yee-Whye Teh. (2006) A fast learning algorithm for deep belief nets. Neural Computation, 18, 1527-1554.
11. Steve Lawrence, C. Lee Giles, Ah Chung Tsoi and Andrew D. Back. (1997) Face recognition: a convolutional neuralnetwork approach. IEEE Transactions on Neural Networks, 8, 98-113.
12. Williams R and Zipser D. (2014) A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. Neural Computation, 1, 270-280.
13. Haohao Li, Qiong Liu, Xiaoming Wei, Zhenhua Chai, Wenbai Chen. (2019) Facial Expression Recognition: Disentangling Expression Based on Self-attention Conditional Generative Adversarial Nets. Proceedings of the PRCV 2019, Xi'an, China, 8C11 November, pp.725-735. Springer, Cham.
14. Aron Gohr. (2019) Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning. Proceedings of the CRYPTO 2019, Santa Barbara, USA, 18-22 August, pp.150-179. Springer, Cham.
15. Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. (2016) Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 27-30 June, pp.770-778. IEEE, New Jersey.
16. Botao Hou, Yongqiang Li, Haoyue Zhao, Bin Wu. (2020) Linear Attack on Round-Reduced DES Using Deep Learning. Proceedings of the ESORICS 2020. Guildford, UK, 14C18 September, pp.131-145. Springer, Cham.
17. Yi Chen and Hongbo Yu. (2021) A New Neural Distinguisher Model Considering Derived Features from Multiple Ciphertext Pairs. IACR Cryptology ePrint Archive, 2021, 310.
18. Farzaneh Abed, Eik List, Stefan Lucks, Jakob Wenzel. (2014) Differential Cryptanalysis of Round-Reduced Simon and Speck. Proceedings of the FSE 2014. London, UK, 3-5 March, pp.525-545. Springer, Berlin.
19. Alex Biryukov, Arnab Roy, Vesselin Velichkov. (2014) Differential Analysis of Block Ciphers SIMON and SPECK. Proceedings of the FSE 2014. London, UK, 3-5 March, pp.546-570. Springer, Berlin.
20. Qingju Wang, Zhiqiang Liu, Kerem Varici, Yu Sasaki, Vincent Rijmen and Yosuke Todo. (2014) Cryptanalysis of Reduced-Round SIMON32 and SIMON48. Proceedings of the INDOCRYPT 2014. New Delhi, India, 14-17 December, pp.143-160. Springer, Cham.
21. Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma and Ling Song. (2014) Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. Proceedings of the ASIACRYPT 2014. Taiwan, China, 7-11 December, pp.158-178. Springer, Berlin.
22. Havard Raddum. (2015) Algebraic Analysis of the Simon Block Cipher Family. Proceedings of the LATINCRYPT 2015. Guadalajara, Mexico, 23-26 August, pp.157-169. Springer, Cham.
23. Mouha N and Preneel B. (2013) Towards Finding Optimal Differential Characteristics for ARX: Application to Salsa20. IACR Cryptology ePrint Archive, 2013, 328.
24. Daniel J Bernstein. (2008) The Salsa20 Family of Stream Ciphers. In: Robshaw M., Billet O. (eds), New Stream Cipher Designs. Springer, Berlin.
25. Leonardo de Moura and Nikolaj Bjorner. (2008) Z3: An Efficient SMT Solver. Proceedings of the TACAS 2008. Budapest, Hungary, March 29-April 6, pp.337-340. Springer, Berlin.

26. Zezhou Hou, Jiongjiong Ren and Shaozhen Chen. (2021) Cryptanalysis of Round-Reduced SIMON32 Based on Deep Learning. IACR Cryptology ePrint Archive, 2021, 362.
27. Nikhil Ketkar. (2017) Deep Learning with Python. Apress, Berkeley, CA.
28. Ling Sun, Wei Wang and Meiqin Wang. (2021) Accelerating the Search of Differential and Linear Characteristics with the SAT Method. IACR Cryptology ePrint Archive, 2021, 213.