Neural Aided Statistical Attack for Cryptanalysis

Yi Chen and Hongbo Yu

Department of Computer Science and Technology, Tsinghua University, P.R. China,
chenyi19@mails.tsinghua.edu.cn
yuhongbo@mail.tsinghua.edu.cn

Abstract. At CRYPTO'19, Gohr first proposed a Neural Distinguisher (ND) built on a plaintext difference. Based on the ND, Gohr developed a key recovery attack model and applied it to Speck32/64. The ND achieves a distinguishing accuracy higher than pure differential distinguishers at certain rounds. However, Gohr's attack model only works when there are plenty of neutral bits and relies purely on experiments for complexity evaluations. Taking a deeper analysis of the attack model, we find the above limitations are caused by one aspect of the black-box nature of neural networks. Can we take advantage of neural networks while avoiding their shortcomings?
In this paper, we prove that the above target can be achieved even if we take neural networks as a black-box. By exploiting the deterministic property of neural networks, we propose a neural aided statistical distinguisher. It not only exploits the advantage of the ND in accuracy, but also avoids the negative influence of its black-box nature. Then we develop a Neural Aided Statistical Attack (NASA) that is a universal attack model for neural aided cryptanalysis. It has no special requirements about the attacked cipher and allows us to estimate the theoretical complexities and success rate. For reducing the key space to be searched, we propose a *Bit Sensitivity Test* to identify which ciphertext bit is informative. Then specific key bits are recovered by building neural distinguishers on related ciphertext bits. Furthermore, since NASA is a universal attack model, it can be improved by properties related to the cipher, such as neutral bits. Applications in round reduced Speck32/64, Speck48/72, Speck48/96 demonstrate the correctness and superiorities of NASA.

Keywords: Cryptanalysis · Neural network · Normal distribution · Statistical attack · Bit sensitivity · Speck.

1 Introduction

Neural aided cryptanalysis is an interesting cryptanalysis technique that has received much expectation since the last century [16, 23]. Deep learning has shown its superiorities in various fields including computer vision [9, 19], natural language processing [5], and so on. But its application in the field of conventional cryptanalysis has been stagnant. A few valuable applications are only concentrated in the side-channel analysis [2, 18].

At CRYPTO'19, Gohr proposed a Neural Distinguisher (ND) and developed the first neural aided key recovery attack on 11-round Speck32/64 [15]. However, the attack proposed by Gohr is far from a generic cryptanalysis tool. Specifically, both the required data complexity and the number of key guesses to be searched are uncertain until practical experiments are finished. It leads to the fact that Gohr's attack can be applied only when the total time complexity is not high. To extend the encryption rounds of the attacked cipher, a differential can be placed before the ND. But Gohr's attack works only when enough neutral bits [12] exist in the differential. This further limits Gohr's attack. Furthermore, there is another open problem. Gohr proved that NDs against Speck32/64 captured some new features from ciphertext pairs. However, these new features are still unknown.

Here we need to introduce two properties of neural networks that are resulted from the black-box nature [8, 10, 21].

Property 1. Given an input X, we can not predict the precise output Z of a neural network.

Property 2. Given an input X and the output Z of a neural network, we do not know why the output of the neural network is Z.

The second property leads to the above open problem. Benamira et al [4] are still figuring out what new features are learned by the ND.

In fact, we can put this open problem aside for now. We find the limitations of Gohr's attack model are only related to *Property 1*. This finding implies that the limitations can be avoided since there is another important property.

Property 3. from a Bayesian perspective, solving binary classification problems with a neural network is equivalent to a posterior probability estimation [20].

We treat the output Z of the ND as a posterior probability. Once we choose a posterior probability threshold c, we create a stable classification hyperplane in the input space. For samples coming from a certain distribution, the probability $Pr(Z > c)$ that the posterior probability Z is higher than the threshold is stable and estimable.

Taking the ND as a black box, Gohr directly built the attack model on the output Z. Since Z is unpredictable, Gohr's attack model is limited seriously. Since Z is not related to the distinguishing accuracy, the advantage of the ND is not exploited either.

Our Contributions. In this paper, we still take the ND as a black box. It means that we do not focus on the features learned by the ND. We show how to build a new attack model by exploiting the deterministic property (the probability $Pr(Z > c)$) of the ND.

We propose a *neural aided statistical distinguisher* in Section 3. Based on this new distinguisher, we develop a Neural Aided Statistical Attack (NASA) for the key recovery in Section 4. NASA transforms the key recovery into the distinguishing between two different normal distributions. This transformation provides a theoretical framework to estimate the data complexity without performing practical experiments. Furthermore, NASA works without any extra requirements about the attacked cipher.

Gohr proposed a highly selective key search policy at a cost of not being able to estimate the number of key guesses to be searched. To make it feasible to estimate the theoretical computation complexity, we choose to traverse all possible key guesses. To significantly reduce the key guess space, a Bit Sensitivity Test (BST) is proposed to identify which ciphertext bit is informative for NDs (Section 5). Informative bits guide us build new NDs on partial ciphertext bits for recovering specific key bits.

To verify the performance of our NASA, we apply it to three round reduced Speck variants. The final results are listed in Table 1. Since the NASA is a universal but basic attack model, it can be enhanced by exploiting some special tricks related to the attacked cipher. Taking the neutral bits as an example, we present a method to reduce the data complexity of NASA. The improved attacks are also listed in Table 1.

Table 1. Summary of key recovery attacks on round reduced Speck. SD: neural aided statistical distinguisher. DD: differential distinguisher. CP: Chosen-Plaintext. DT: distinguisher type. DC: decryption complexity. PC: prediction complexity. The complexities of Gohr's attack on 11-round Speck32/64 are estimated again using our metrics.

cipher	DT	Rounds	DC	PC	Data	Time	Source
Speck32/64	DD	11	2^{46}	-	2^{14}CP	2^{46}	[11]
	ND	11	$2^{18.36}$	$2^{20.82}$	$2^{14.5}$CP	$2^{24.3}$	[15]
	SD	11	$2^{32.29}$	$2^{34.75}$	$2^{23.44}$CP	$2^{38.23}$	Section 6
	SD	11	$2^{25.5}$	$2^{27.96}$	$2^{20.39}$CP	$2^{31.44}$	Section 8
	DD	12	2^{51}	-	2^{19}CP	2^{51}	[11]
	ND	12	-	-	-	-	[15]
	SD	12	$2^{40.35}$	$2^{42.93}$	$2^{27.93}$CP	$2^{46.41}$	Section 6
	SD	12	$2^{37.39}$	$2^{39.97}$	$2^{24.92}$CP	$\mathbf{2^{43.45}}$	Section 8
	DD	13	2^{57}	-	2^{25}CP	2^{57}	[11]
	SD	13	2^{58}	$2^{60.7}$	$2^{28.7}$CP	$2^{64.18}$	Section 6
	SD	13	$2^{55.17}$	$2^{57.87}$	$2^{25.87}$CP	$2^{61.35}$	Section 8
Speck48/72 Speck48/96	DD	12	2^{43}	-	2^{43}CP	2^{43}	[6]
	SD	12	$2^{50.91}$	$2^{53.49}$	$\mathbf{2^{37.32}}$CP	$2^{56.97}$	Section 6
	SD	12	$2^{44.86}$	$2^{47.44}$	$\mathbf{2^{32.37}}$CP	$2^{50.92}$	Section 8

To fairly compare the complexity of various attacks, we introduce two new concepts: (1) Decryption Complexity (DC), (2) Prediction Complexity (PC). The total time complexity is DC $+\rho\times$ PC. In this article, $4.9 \leqslant \rho \leqslant 11$. More information can refer to Section 6.1.

Table 1 shows that NASA is more generic than Gohr's attack. Compared with classic attacks on 12-round Speck32/64 or Speck48/72 (Speck48/96), NASA significantly reduce the decryption complexity or data complexity.

2 Review of Gohr's Work

We briefly review Gohr's work and analyze the causes of limitations.

2.1 Neural Distinguisher

Let (P_0, P_1) denote a plaintext pair with difference $\varDelta P$. The corresponding intermediate states, ciphertexts are (S_0, S_1), (C_0, C_1). The target of the ND [15] is to distinguish two classes of ciphertext pairs (C_0, C_1)

$$Y = \begin{cases} 1, & if \ S_0 \oplus S_1 = \varDelta S \\ 0, & if \ S_0 \oplus S_1 \neq \varDelta S \end{cases} \tag{1}$$

$Y = 1$ or $Y = 0$ is the label of (C_0, C_1). If the difference between S_0 and S_1 is the target difference $\varDelta S$, the pair (C_0, C_1) is regarded as a positive sample drawn from the target distribution. Or (C_0, C_1) is regarded as a negative sample that comes from a uniform distribution.

A neural network is trained over $\frac{N}{2}$ positive samples and $\frac{N}{2}$ negative samples. The neural network is used as a distinguisher if the distinguishing accuracy over a testing database is higher than 0.5. Let ND_h denotes a ND against the cipher reduced to h rounds. Given a sample (C_0, C_1), the ND will output a score Z which is used as the posterior probability

$$Pr(Y = 1 \,|(C_0, C_1)\,) = Z = F(C_0, C_1), \quad 0 \leqslant Z \leqslant 1, \tag{2}$$

where $F(\cdot)$ stands for the posterior probability estimation function learned by the ND. When $Z > 0.5$, the predicted label of (C_0, C_1) is 1 [15]. For convenience, we also use $F(\cdot)$ to denote the ND in some cases.

2.2 Gohr's Key Recovery Attack Model

Algorithm 1 summarizes the core idea of the basic version (unaccelerated version) of Gohr's key recovery attack.

To understand Gohr's key recovery attack, we focus on ciphertext pairs with the label $Y = 1$. Decrypt such a ciphertext pair with a subkey guess kg, and feed the decrypted ciphertext pair into a neural distinguisher ND. If kg is the right subkey, denote the output of ND as Z_r. Or denote the out of ND as Z_w. According to the description of the ND, we know that $Z_r > Z_w$ will hold with a high probability. Then the subkey guesses are linked to the output signal of ND.

To increase the gap between the signal related to the right subkey and that related to the wrong subkeys, Gohr uses the following formula

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

to combine the signals Z_i of N decrypted ciphertext pairs into a rank score v_{kg} for the subkey guess kg. These N ciphertext pairs should satisfy $\varDelta P \rightarrow \varDelta S$

Algorithm 1 Basic version of Gohr's key recovery attack model

```
Require: $k$ neutral bits [12] that exist in the differential transition $\Delta P \to \Delta S$;
    A ND built from $\Delta S$, $F(\cdot)$;
    A key rank score threshold, $c_1$; A maximum number of iterations.
Ensure: A possible key candidate.
 1: repeat
 2:    Random generate a plaintext pair $(P_0^1, P_1^1), P_0^1 \oplus P_1^1 = \Delta P$;
 3:    Create a plaintext structure consisting of $2^k$ plaintext pairs by $k$ neutral bits;
 4:    Collect corresponding ciphertext pairs, $(C_0^i, C_1^i), i \in \{1, \cdots, 2^k\}$;
 5:    for each key guess $kg$ do
 6:       Partially decrypt $2^k$ ciphertext pairs with $kg$;
 7:       Feed decrypted ciphertext pairs to $F(\cdot)$ and collect the outputs;
 8:       Calculate the key rank score $v_{kg}$ based on collected outputs;
 9:       if $v_{kg} > c_1$ then
10:          stop the key search and return $kg$ as the key candidate;
11:       end if
12:    end for
13: until a key candidate is returned or the maximum number of iterations is reached.
```

simultaneously. Neutral bits [12] are adopted for gathering such N ciphertext pairs. When the rank score v_{kg} exceeds a threshold c_1, kg is very likely to be the right subkey.

It's clear that c_1, Z_r, Z_w determine the number N of required ciphertext pairs. However, the values of Z_r, Z_w are unpredictable. This further makes the choice of c_1 lack a theoretical basis. Then N is set to an experimental value. The final negative influence is that the theoretical data complexity N of Gohr's attack can not be estimated.

The above analysis brings two important inspirations. First, it's possible to break the limitations even if we do not figure out new features captured by NDs. Second, we should recover the right key by exploiting the determinstic properties of the ND.

3 Neural Aided Statistical Distinguisher

3.1 A Chosen Plaintext Statistical Distinguisher

Generate N chosen-plaintext pairs $(P_0^i, P_1^i), P_0^i \oplus P_1^i = \Delta P, i \in \{1, \cdots, N\}$ randomly and collect corresponding ciphertext pairs $\left(C_0^i, C_1^i\right), i \in \{1, \cdots, N\}$ using a cipher with a block size of L. Given a neural ND $F\left(C_0, C_1\right)$, the adversary needs to distinguish between this cipher and a random permutation.

The concrete process is as follows. For each ciphertext pair $\left(C_0^i, C_1^i\right)$, the adversary feeds it into the neural distinguisher and obtains its output Z_i. Setting a threshold value c_2, the adversary calculates the statistic T

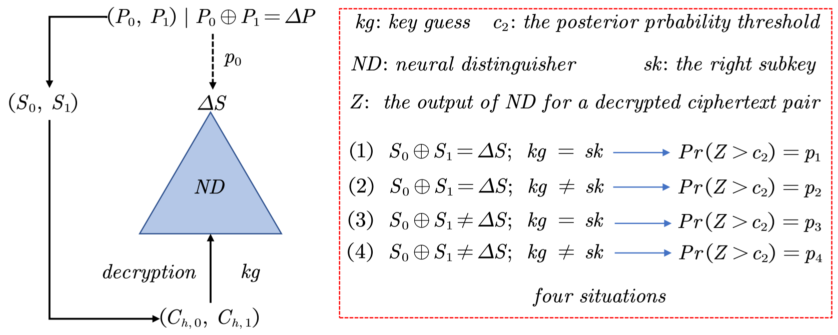
$$T = \sum_{i=1}^{N} \phi\left(Z_i\right), \quad \phi\left(Z_i\right) = \begin{cases} 1, if\ Z_i > c_2 \\ 0, if\ Z_i \leqslant c_2 \end{cases} \tag{4}$$

When the probability of the differential transition $\Delta P \to \Delta S$ is higher than 2^{-L}, it's expected that the value of the statistic T for the cipher is higher than that for a random permutation. In a key recovery setting, the right key will result in the statistic T being among the highest values for all candidate keys if N is large enough. In the sequel, we give this a theoretical analysis.

Remark 1. The value of the posterior probability threshold c_2 is selected experimentally in this paper. The value of c_2 has an indirect influence on the data complexity. Besides, the deep residual network [15] proposed by Gohr is also used to built NDs. But the depth of the residual block is set 1.

3.2 Distribution of the Statistic under Right and Wrong keys

First, we regard a ciphertext pair as a point in a high-dimensional space. For a given threshold, it's equivalent to creating a stable classification hyperplane in this space using a ND. Thus the classification over a random ciphertext pair is modeled as a Bernoulli experiment. It provides us with a theoretical analysis framework.



$(P_0, P_1) \mid P_0 \oplus P_1 = \Delta P$
p_0
(S_0, S_1)
ΔS
ND
decryption
kg
$(C_{h,0}, C_{h,1})$
kg: key guess
c_2: the posterior prbability threshold
ND: neural distinguisher
sk: the right subkey
Z: the output of ND for a decrypted ciphertext pair
(1) $S_0 \oplus S_1 = \Delta S$; $kg = sk$ → $Pr(Z > c_2) = p_1$
(2) $S_0 \oplus S_1 = \Delta S$; $kg \neq sk$ → $Pr(Z > c_2) = p_2$
(3) $S_0 \oplus S_1 \neq \Delta S$; $kg = sk$ → $Pr(Z > c_2) = p_3$
(4) $S_0 \oplus S_1 \neq \Delta S$; $kg \neq sk$ → $Pr(Z > c_2) = p_4$
four situations

Fig. 1. Four situations of decrypting a ciphertext pair with a key guess.

According to the key recovery process, there are four possible situations when we decrypt a ciphertext pair with a key guess as shown in Fig.1:

(1) Decrypting a positive sample with the right key: the ciphertext pair satisfies the differential transition and the key guess is right.

(2) Decrypting a positive sample with wrong keys: the ciphertext pair satisfies the differential transition but the key guess is wrong.

(3) Decrypting a negative sample with the right key: the ciphertext pair does not satisfy the differential transition but the key guess is right.

(4) Decrypting a negative sample with wrong keys: the ciphertext pair does not satisfy the differential transition and the key guess is wrong.

Given a ND, we denote the probability of $Z > c_2$ as p_1, p_2, p_3, p_4 for the four situations respectively. Then the distributions of the statistic (formula (4)) in these four situations are

$$\begin{array}{l} T_1 \sim \mathcal{N}(\mu_1, \sigma_1), \mu_1 = N_1 \times p_1, \sigma_1 = \sqrt{N_1 \times p_1 \times (1 - p_1)} \\ T_2 \sim \mathcal{N}(\mu_2, \sigma_2), \mu_2 = N_2 \times p_2, \sigma_2 = \sqrt{N_2 \times p_2 \times (1 - p_2)} \\ T_3 \sim \mathcal{N}(\mu_3, \sigma_3), \mu_3 = N_3 \times p_3, \sigma_3 = \sqrt{N_3 \times p_3 \times (1 - p_3)} \\ T_4 \sim \mathcal{N}(\mu_4, \sigma_4), \mu_4 = N_4 \times p_4, \sigma_4 = \sqrt{N_4 \times p_4 \times (1 - p_4)} \end{array} \tag{5}$$

if N_1, N_2, N_3, N_4 are high enough. $\mathcal{N}(\mu_i, \sigma_i)$ is a normal distribution with mean μ_i and standard deviation $\sigma_i, i \in \{1, 2, 3, 4\}$. An empirical condition is

$$N_i \times p_i > 5, \quad N_i \times (1 - p_i) > 5, \quad i \in \{1, 2, 3, 4\} \tag{6}$$

Now come back to the key recovery attack. If the probability of the differential transition $\Delta P \to \Delta S$ is p_0 and N ciphertext pairs are collected randomly, then

$$N_1 = N_2 = N \times p_0, \quad N_3 = N_4 = N \times (1 - p_0) \tag{7}$$

Besides, the distributions of the statistic (formula (4)) under the right key and wrong keys are both a mixture of two normal distributions.

Right key guess. This case contains two situations in which corresponding distributions are $\mathcal{N}(\mu_1, \sigma_1)$ and $\mathcal{N}(\mu_3, \sigma_3)$. Since a mixture of two independent normal distributions is still a normal distribution, the distribution of the statistic (formula (4)) under the right key guess is:

$$T_r = T_1 + T_3 \sim \mathcal{N}(\mu_r, \sigma_r) \tag{8}$$

$$\mu_r = N \times (p_0 p_1 + (1 - p_0) p_3) \tag{9}$$

$$\sigma_r = \sqrt{N \times p_0 \times p_1 \times (1 - p_1) + N (1 - p_0) \times p_3 \times (1 - p_3)} \tag{10}$$

Wrong key guess. This case also contains two situations in which corresponding distributions are $\mathcal{N}(\mu_2, \sigma_2)$ and $\mathcal{N}(\mu_4, \sigma_4)$. Then the distribution of the statistic (formula (4)) under wrong key guesses is:

$$T_w = T_2 + T_4 \sim \mathcal{N}(\mu_w, \sigma_w) \tag{11}$$

$$\mu_w = N \times (p_0 p_2 + (1 - p_0) p_4) \tag{12}$$

$$\sigma_w = \sqrt{N \times p_0 \times p_2 \times (1 - p_2) + N (1 - p_0) \times p_4 \times (1 - p_4)} \tag{13}$$

Negative samples in the high-dimensional space approximately obey uniform distribution, thus p_3 and p_4 are theoretically equal. $p_3 \approx p_4$ are also verified by experiments. Since the accuracy of NDs is higher than 0.5, $p_1 > p_2$ also holds with a high probability. When we set $c_2 = 0.5$, we can ensure $p_1 > p_2$. Thus $\mu_r > \mu_w$ also holds.

From the analysis above, we know that the distributions of T_r, T_w are different. Then the key recovery attack is performed based on this finding.

3.3 Distinguishing between Two Normal Distributions

The distinguishing between two normal distributions is also adopted in zero correlation cryptanalysis [7]. Here, we still give the details.

Consider two normal distributions: $\mathcal{N}(\mu_r, \sigma_r)$, and $\mathcal{N}(\mu_w, \sigma_w)$. A sample s is sampled from either $\mathcal{N}(\mu_r, \sigma_r)$ or $\mathcal{N}(\mu_w, \sigma_w)$. We have to decide if this sample is from $\mathcal{N}(\mu_r, \sigma_r)$ or $\mathcal{N}(\mu_w, \sigma_w)$. The decision is made by comparing the value s to some threshold t. Without loss of generality, assume that $\mu_r > \mu_w$. If $s \geqslant t$, the decision is $s \in \mathcal{N}(\mu_r, \sigma_r)$. If $s < t$, the decision is $s \in \mathcal{N}(\mu_w, \sigma_w)$. Then there are error probabilities of two types:

$$\begin{aligned}\beta_r &= Pr\left\{\mathcal{N}(\mu_w, \sigma_w) \,|s \in \mathcal{N}(\mu_r, \sigma_r)\right\}, \\ \beta_w &= Pr\left\{\mathcal{N}(\mu_r, \sigma_r) \,|s \in \mathcal{N}(\mu_w, \sigma_w)\right\}.\end{aligned} \tag{14}$$

When a sample s is sampled from $\mathcal{N}(\mu_r, \sigma_r)$, the probability that the decision is $s \in \mathcal{N}(\mu_w, \sigma_w)$ is β_r.

Here a condition is given on μ_r, μ_w, σ_r, σ_w such that the error probabilities are β_r and β_w. The proof can refer to related research [13, 14].

Proposition 1. *For the test to have error probabilities of at most β_r and β_w, the parameters of the normal distribution $N(\mu_r, \sigma_r)$ and $N(\mu_w, \sigma_w)$ with $\mu_r \neq \mu_w$ have to be such that*

$$\frac{z_{1-\beta_r} \times \sigma_r + z_{1-\beta_w} \times \sigma_w}{|\mu_r - \mu_w|} = 1 \tag{15}$$

where $z_{1-\beta_r}$ and $z_{1-\beta_w}$ are the quantiles of the standard normal distribution.

3.4 Data Complexity of the Statistical Distinguisher

Based on Proposition 1, one obtains the condition:

$$\frac{z_{1-\beta_r}\sigma_r + z_{1-\beta_w}\sigma_w}{\mu_r - \mu_w} = 1 \tag{16}$$

where the values of $\mu_r, \sigma_r, \mu_w, \sigma_w$ refer to formula (9), (10), (12), (13) respectively. In a key recovery setting, $1 - \beta_r$ is the minimum probability that the right key survives, β_w is the maximum probability that wrong keys survive.

Since we can not know the real classification hyperplane learned by the ND, p_1, p_2, p_3, and p_4 are estimated experimentally. Then the estimated values of p_3 and p_4 will be slightly different even they should be theoretically equal. When the probability p_0 of the differential transition is very low, the slight distinction $p_3 - p_4$ may dominate $\mu_r - \mu_w$, which is wrong. Thus we neglect the minor difference and replace p_3, p_4 with p_n.

Then the final condition can be simplified:

$$\sqrt{N} = \frac{z_{1-\beta_r} \times \sqrt{p_0a_1 + (1-p_0)a_3} + z_{1-\beta_w} \times \sqrt{p_0a_2 + (1-p_0)a_3}}{(p_1 - p_2) \times p_0}, \tag{17}$$

$$a_1 = p_1(1-p_1), \quad a_2 = p_2(1-p_2), \quad a_3 = p_n(1-p_n). \tag{18}$$

The data complexity N is directly calculated when β_r and β_w are set. N is affected by p_0, p_1, p_2, p_n simultaneously. The impacts of p_0, p_1, p_2, p_n on N are $\mathcal{O}(p_0^{-2})$, $\mathcal{O}((p_1 - p_2)^{-2})$, $\mathcal{O}(p_n)$ respectively. The proof can be found in Supplementary Materials.

The decision threshold t is:

$$t = \mu_r - z_{1-\beta_r}\sigma_r = \mu_w + z_{1-\beta_w}\sigma_w. \tag{19}$$

3.5 Estimation of p_1, p_n

Consider a ND $F(\cdot)$ against a cipher reduced to h rounds, the values of p_1, p_n are estimated as:

1. Randomly generate M positive (negative) samples and decrypt them for 1 round with the right (random) subkeys.
2. Feed partially decrypted samples into $F(\cdot)$.
3. Calculate the final ratio of $Z > c_2$.

The ratio is the statistical expectation of p_1 or p_n. A large M can help make the statistical expectation accurate enough.

3.6 Further Analysis and Estimation of p_2

When we decrypt a positive sample with a wrong key guess (Fig.1(2)), the final value of p_2 is rather complex and related to characteristics of the wrong key guess. Such a phenomenon is based on *Property 1* and *Property 2.*

Property 4. Decrypt a ciphertext for one round with two different subkeys,

$$C^1_{h-1} = DecOneRound(C_h, kg_1), \quad C^2_{h-1} = DecOneRound(C_h, kg_2).$$

If kg_1 and kg_2 are only different at a few bits (e.g. just 1 bit or 2 bits), the Hamming distance between C^1_{h-1} and C^2_{h-1} will be very small in high probability.

Property 5. Consider a neural network $F(\cdot)$ solving a binary classification problem. If two input samples X_1, X_2 are very close to each other in the input space, two outputs $F(X_1), F(X_2)$ of the neural network may satisfy $F(X_1) \approx F(X_2)$ with a high probability.

Although the distance metric in the input space of neural networks is complex and unknown, the Hamming distance is still a good alternative. Thus it is expected that p_2 is related to the Hamming distance between the right key and wrong key guesses.

Suppose we decrypt a positive sample $(C_{h+x,0}, C_{h+x,1})$ with x subkey guesses simultaneously

$$C_{h+j-1,0/1} = DecOneRound(C_{h+j,0/1}, kg_{h+j}), \quad j \in [1, x] \tag{20}$$

where kg_{h+j} is the key guess of the $(h+j)$-th round. $(C_{h,0}, C_{h,1})$ is fed into an h-round ND for estimating p_2.

When the last $x-1$ key guesses $kg_{h+j}, j \in [2, x]$ are all right, $(C_{h+1,0}, C_{h+1,1})$ is a positive sample. The probability of $Z > c_2$ is p_2. If $kg_{h+j}, j \in \{2, \cdots, x\}$ are not all right, then $(C_{h+1,0}, C_{h+1,1})$ is not a positive sample anymore. The resulted probability of $Z > c_2$ is closer to p_n.

Since x key guesses have different influences on the probability of $Z > c_2$, we consider x Hamming distances for estimating p_2. Let d_j denotes the Hamming distance between the right key and key guess in the $(h+j)$-th round, and $p_{2|d_1,\cdots,d_x}$ denotes the probability of $Z > c_2$. Algorithm 2 is proposed for the estimation of $p_{2|d_1,\cdots,d_x}$.

Algorithm 2 Estimation of $p_{2|d_1,\cdots,d_x}$

```
Require: a cipher with a subkey size of $L$;
    a ND against this cipher reduced to $r$ rounds, $F(\cdot)$;
    $M$ random plaintext pairs, $(P_0^i, P_1^i), P_0^i \oplus P_1^i = \Delta P, i \in \{1, \cdots, M\}$;
    $M$ random master keys, $MK_i, i \in \{1, \cdots, M\}$;
    The threshold $c_2$.
Ensure: $p_{2|d_1,\cdots,d_x}$ .
 1: Encrypt each plaintext pair $(P_0^i, P_1^i)$ with a master key $MK_i$ for $h+x$ rounds;
 2: Save resulting ciphertext pair $(C_0^i, C_1^i)$;
 3: Save the $(h+j)$-th subkey $sk_{h+j}^i, j \in \{1, \cdots, x\}$;
 4: for $d_1 = 0$ to $L$, $\cdots$, $d_x = 0$ to $L$ do
 5:     for $i = 1$ to $M$ do
 6:         Randomly draw $x$ key guesses $kg_j^i, j \in \{1, \cdots, x\}$ where the Hamming distance
            between $kg_j^i$ and $sk_{h+j}^i$ is $d_j$;
 7:         Decrypt $(C_0^i, C_1^i)$ with $kg_j^i, j \in \{1, \cdots, x\}$ for $x$ rounds;
 8:         Feed the decrypted ciphertext pair into $F(\cdot)$ and save the output as $Z_{i|d_1,\cdots,d_x}$;
 9:     end for
10:     Count the number of $Z_{i|d_1,\cdots,d_x} > c_2$, and denote it as $T_{d_1,\cdots,d_x}$;
11:     Save $p_{2|d_1,\cdots,d_x} = \frac{T_{d_1,\cdots,d_x}}{M}$.
12: end for
```

Verification. We have performed tests on 5 NDs against round reduced Speck32/64. The difference constraint is $\Delta S = (0x0040, 0)$. Let $M = 10^7$, Table 2 and Table 3 show the estimation results of $p_{2|d_1}$ and $p_{2|d_1,d_2}$ respectively.

Tests above have verified the analysis of p_2. Besides, when two subkeys are guessed simultaneously, $p_{2|d_1,d_2}$ will decrease sharply even if the key guess of the last round is wrong at only 1 bit.

Thus, **the choice of p_2 depends on the target of the key recovery attack**. If we think the attack is successful as long as the Hamming distance between the key guess and the right key is not higher than a threshold d, the value of p_2 should be

$$p_2 = \max\left\{p_{2|d_1,\cdots,d_x} | d_1 + \cdots + d_x > d\right\} \tag{21}$$

Table 2. The estimation of $p_{2|d_1}$ of the NDs against round reduced Speck32/64. For ND_4, ND_5, ND_6, ND_7, $c_2 = 0.55$. For ND_8, $c_2 = 0.5$. $p_{2|d_1=0} = p_1$.

ND_4	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.995	0.5065	0.2815	0.1686	0.1088	0.0735	0.0521	0.039	$\leqslant 0.0301$
ND_5	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.8889	0.5151	0.3213	0.2168	0.1556	0.1189	0.0956	0.08	$\leqslant 0.0694$
ND_6	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.6785	0.4429	0.3135	0.2384	0.1947	0.1684	0.1518	0.1408	$\leqslant 0.1334$
ND_7	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.4183	0.3369	0.2884	0.2607	0.2442	0.234	0.2276	0.2236	$\leqslant 0.2211$
ND_8	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.5184	0.5056	0.4993	0.4957	0.4939	0.4927	0.4925	0.4918	$\leqslant 0.4917$

Table 3. The estimation of $p_{2|d_1,d_2}$ of the 7-round ND against Speck32/64. $c_2 = 0.55$. The columns correspond to d_2. The rows correspond to d_1. All results only retain two decimal places. The same value is replaced by an uppercase letter. $Y = 0.21$, $E = 0.22$, $J = 0.23$, $U = 0.25$, and $V = 0.26$.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0.42	V	E	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
1	0.33	U	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	E
2	0.29	J	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
3	V	J	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
4	J	J	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
5	E	E	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
6	E	Y	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
7	E	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	E
8	E	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
$9 \sim 16$	$\leqslant Y$																

This choice is based on the following truth. By setting a proper threshold c_2 such as $c_2 \geqslant 0.5$, we can ensure

$$p_{2|d_1,\cdots,d_x} \leqslant 0.5, \quad if \quad d_1 + \cdots + d_x > d \tag{22}$$

According to formula (18), the higher p_2 is, the higher the required data complexity is. The decision threshold also increases when p_2 increases. Thus we only need to focus on the highest data complexity needed for filtering wrong keys.

Take the 7-round ND as an example. Let $d = 2$, it means that the attack is successful if the recovered key is different from the right key at most 2 bits. Then $p_2 = p_{2|3} = 0.2607$ or $p_2 = p_{2|0,1} = p_{2|3,0} = 0.26$.

4 Neural Aided Statistical Attack

4.1 Key Recovery Model

This neural aided statistical distinguisher can be used to determine whether a key guess may be the right key. This is done by the *Statistical Test* as shown in Algorithm 3. Algorithm 4 summarizes the basic attack model based on the neural aided statistical distinguisher.

Algorithm 3 Statistical test for a key guess

```
Require: A ND;  A key guess, $kg$;
    A posterior probability threshold, $c_2$;  The decision threshold, $t$;
    $N$ ciphertext pairs $(C_0^i, C_1^i)$ encrypted from $(P_0^i, P_1^i), P_0^i \oplus P_1^i = \Delta P, i \in [1, N]$.
1: Decrypt $N$ ciphertext pairs with $kg$;
2: Feed decrypted ciphertext pairs into the ND, and collect the outputs $Z_i, i \in [1, N]$;
3: Calculate the statistic $T$ in formula (4);
4: if $T \geqslant t$ then
5:     Return $kg$ as a key candidate.
6: end if
```

Algorithm 4 The general model of our neural aided statistical attack

```
Require: The attacked cipher;
    The differential transition with a probability of $p_0$, $\Delta P \to \Delta S$;
    Two maximum error probabilities, $\beta_r, \beta_w$;
    A posterior probability threshold, $c_2$.
Ensure: All possible key candidates.
1: Train a ND based on $\Delta S$, $F(C_0, C_1)$;
2: Estimate $p_1, p_n, p_2$ using $F(C_0, C_1)$ (Section 3.5, Algorithm 2);
3: Calculate the data complexity $N$ and the decision threshold $t$ (Section 3.4);
4: Randomly generate $N$ plaintext pairs $(P_0^i, P_1^i), P_0^i \oplus P_1^i = \Delta P, i \in \{1, \cdots, N\}$;
5: Collect corresponding $N$ ciphertext pairs, $(C_0^i, C_1^i), i \in \{1, \cdots, N\}$;
6: for each key guess $kg$ do
7:     Perform the statistical test (Algorithm 3);
8: end for
9: Test surviving key candidates against a necessary number of plaintext-ciphertext
   pairs according to the unicity distance for the attacked cipher.
```

4.2 Verification

To verify our NASA, various practical attacks on Speck32/64 have been performed. NASA should work as long as the ND has a distinguishing accuracy higher than 0.5. Besides, the data complexity should be correctly estimated once $\Delta P \xrightarrow{p_0} \Delta S$, ND, d, β_r, and β_w are provided.

Thus, the verification plan consists of three steps:

1. Set the value of β_r and β_w. Calculate the data complexity N.
2. Perform the NASA 100 times with N samples.
3. Check the following observation indexes:
   (a) The ratio that the right key ($d_1 = 0$) passes the statistical test.
   (b) The average number of surviving keys in 100 trails.
   (c) The ratio that the number of surviving keys is smaller than the expected upper bound.

Attack 1: recover sk_{10} of 10-round Speck32/64. Table 4 summarizes the attack setting. It's expected that returned key guesses are different from the right key at most $d = 2$ bits.

Table 4. The attack setting of **Attack 1**.

$\Delta P \xrightarrow{p_0} \Delta S$	ND	c_2	p_1	d	p_2	p_n	β_r	β_w
$0x211/0xa04 \xrightarrow{p_0=2^{-2}} 0x2800/0x10$	ND_7	0.55	0.4183	2	0.2607	0.2162	0.005	2^{-16}

The prepended differential is a 1-round differential with a probability of 2^{-2}. Since no key addition happens in Speck before the first nonlinear operation, this differential is extended to a 2-round differential without probability loss. The ND adopted in this attack is ND_7.

In this attack setting, we get $N = 5274 \approx 2^{12.34}$ and $t = 1325$ (see formula (18)). The decision threshold is $t = 1325$. The right key ($d_1 = 0$) should survive with a $1 - \beta_r = 0.995$ probability at least. Wrong keys ($d_1 \geqslant 3$) should survive with a $\beta_w = 2^{-16}$ probability at most. The number of surviving keys shouldn't exceed $137 \times 0.995 + (2^{16} - 137) \times 2^{-16} = 137.31$.

After performing this attack 100 times with 5274 plaintext pairs, we find:

1. The right key ($d_1 = 0$) has passed the test in 99 experiments.
2. The average number of surviving keys is 63.54 that is smaller than 137.31.
3. The number of surviving keys is smaller than 137.31 in 98 experiments.

Attack 2: recover sk_{10} of 10-round Speck32/64. Table 7 summarizes the attack setting. ND_8 is a very weak distinguisher. Its distinguishing accuracy is only about 0.518.

Table 5. The attack setting of **Attack 2**.

$\Delta P \xrightarrow{p_0} \Delta S$	ND	c_2	p_1	d	p_2	p_n	β_r	β_w
-	ND_8	0.5	0.5184	2	0.4957	0.4914	0.001	2^{-16}

In this attack setting, $N = 25680 \approx 2^{14.65}$ and $t = 13064$. The number of surviving keys shouldn't exceed 137.998. After performing this attack 100 times with 25680 plaintext pairs, we find:

1. The right key ($d_1 = 0$) has passed the test in all the 100 experiments.
2. The average number of surviving keys is 77.47 that is smaller than 137.998.
3. The number of surviving keys is smaller than 137.998 in 85 experiments.

5 Reduce the Key Space

So far we still need to guess all the bits of the subkey simultaneously, since the ND takes the complete ciphertext pairs (C_0, C_1) as the input. When the subkey has a large size, this is a serious bottleneck.

5.1 An Intuitive Method for Reducing the Key Space

An intuitive method for reducing the key space is building the ND on partial ciphertext bits

$$C_i = C_i[L-1]||\cdots||C_i[0], i \in [0,1] \tag{23}$$

$$\Gamma = \{x_1, x_2, \cdots, x_k\}, x_1 > \cdots > x_k, k <= L \tag{24}$$

$$\varphi(C_i, \Gamma) = C_i[x_1]||C_i[x_2]||C_i[x_k], i \in [0,1] \tag{25}$$

$$Y(\varphi(C_0, \mathit{\Gamma}), \varphi(C_1, \mathit{\Gamma})) = \begin{cases} 1, & if\ S_0 \oplus S_1 = \mathit{\Delta} S \\ 0, & if\ S_0 \oplus S_1 \neq \mathit{\Delta} S \end{cases} \tag{26}$$

$$Pr(Y = 1\,|(\varphi(C_0, \mathit{\Gamma}), \varphi(C_1, \mathit{\Gamma}))\,) = Z = f(\varphi(C_0, \mathit{\Gamma}), \varphi(C_1, \mathit{\Gamma})) \tag{27}$$

where $C_i[0]$ is the least significant bit of the ciphertext C_i, $\mathit{\Gamma}$ is the subscript set of selected ciphertext bits, $f(\cdot)$ is the ND built on selected ciphertext bits.

Such a method significantly reduces the key space to be searched. But **which ciphertext bits should we select for building $f(\cdot)$? Can we develop a generic and efficient framework for guiding this selection?** In order to better introduce our work for solving these problems, three new concepts are proposed first.

Definition 1 *An **informative bit** is the ciphertext bit that is helpful to distinguish the cipher and a pseudo-random permutation.*

Definition 2 *For a cipher reduced to h rounds, the ND $F(\cdot)$ trained on the complete ciphertexts (C_0, C_1) is denoted as the **teacher distinguisher** ND_h^t, the ND $f(\cdot)$ trained on the selected ciphertext bits $(\varphi(C_0, \mathit{\Gamma}), \varphi(C_1, \mathit{\Gamma}))$ is denoted as the **student distinguisher** ND_h^s. The teacher distinguisher is viewed as a special student distinguisher.*

5.2 Identify Informative Bits by Bit Sensitivity Test

It's clear that student distinguishers should be built on informative bits. However, it's hard to identify informative bits according to **Definition 1**. Thus we propose an approximate definition of the informative bit.

Definition 3 *For a teacher distinguisher $F(C_0, C_1)$, if the distinguishing accuracy is greatly affected by the j-th bit of C_0, C_1, the j-th ciphertext bit is an **informative bit**.*

A teacher distinguisher works since it has learned knowledge from ciphertext bits. According to **Definition 1**, only the informative bit can provide knowledge. Thus the ciphertext bit that has a significant influence on the distinguishing accuracy of the teacher distinguisher must be the *informative bit*.

Definition 3 can not ensure each informative bit that obeys the **Definition 1** is identified successfully. But we only care about informative bits that are captured by a teacher distinguisher. This approximate definition can help develop a simple but effective framework for identifying informative bits.

The proposed framework is named **Bit Sensitivity Test** (BST). Its core idea is to test whether the teacher distinguisher's distinguishing accuracy drops after we remove some knowledge related to the specific bit.

Gohr in [15] has proved that teacher distinguishers capture the knowledge about the ciphertext difference and some unknown features. Consider the j-th ciphertext bit. We remove the knowledge about the j-th ciphertext bit's difference by

$$C_0 = C_0 \oplus (\eta << j) \quad or \quad C_1 = C_1 \oplus (\eta << j) \tag{28}$$

where η is a random mask that could be 0 or 1.

We have performed an extreme test on teacher distinguishers against Speck32/64 reduced to 4/5/6/7/8 rounds. If we XOR each bit of C_0 or C_1 with a random mask, teacher distinguishers can not distinguish positive samples and negative samples anymore. These tests imply that knowledge about unknown features is also removed by one of the two operations above.

After the knowledge related to a ciphertext bit is removed, the decrease of the teacher distinguisher's accuracy is named **Bit Sensitivity**, which is used to identify informative bits. Algorithm 5 summarizes the BST.

Algorithm 5 Bit Sensitivity Test

```
Require: a cipher with a block size of $L$;
    a teacher distinguisher against this cipher, $F(C_0, C_1)$;
    a test dataset consisting of $\frac{M}{2}$ positive samples and $\frac{M}{2}$ negative samples.
Ensure: An array $sen$ that saves the bit sensitivity of $L$ ciphertext bits.
 1: Test the distinguishing accuracy of the ND on the test dataset. Save it to $sen[L]$;
 2: for $j = 0$ to $L - 1$ do
 3:     for $i = 1$ to $M$ do
 4:         Generate a random mask $\eta \in \{0, 1\}$;
 5:         $C_0^{i,new} = C_0^i \oplus (\eta << j)$;
 6:         Feed the new sample $(C_0^{i,new}, C_1^i)$ to the ND;
 7:     end for
 8:     Count the current accuracy $cp$;
 9:     $sen[j] = sen[L] - cp$;
10: end for
```

Examples and analysis. We have applied the BST to three teacher distinguishers against Speck32/64. Table 10 shows the results sen_0, sen_1, $sen_{0,1}$ of the bit sensitivity test under three scenarios.

According to Table 10, we observe that $sen_0 \approx sen_1$. This proves that $C_0 \oplus (\eta << j)$ is equivalent to $C_1 \oplus (\eta << j)$. Besides, we know

1. If $sen_0[j] > 0$, the j-th ciphertext bit is an informative bit.
2. If $sen_{0,1}[j] > 0$, the j-th ciphertext bit provides some useful unknown features. Since the knowledge about the bit difference is not removed, then only useful unknown features can lead to a decrease in the accuracy.
3. If $sen_0[j] \approx sen_{0,1}[j]$, the j-th ciphertext bit's difference has little influence on the ND.

Reverse verification about identified informative bits. To further verify **Definition 3**, a reverse verification about identified informative bits is performed. First, select some informative bits. Second, train a student distinguisher on selected informative bits and observe the distinguishing accuracy.

Taking the ND_7^t against Speck32/64 as an example, we have performed the reverse verification based on results in Table 10. Table 11 shows the distinguishing accuracies under two settings. For Speck32/64, the j-th and $(j + 16)$-th bit are directly related to the same subkey bit. Thus the 8-th and 1st ciphertext bits are also considered.

Table 6. Results of *Bit Sensitivity Test* of teacher distinguishers against Speck32/64 under three scenarios, $M = 10^6$. sen_0 is the results of performing $C_0 \oplus (\eta << j)$, sen_1 is the results of performing $C_1 \oplus (\eta << j)$, $sen_{0,1}$ is the results of performing two operations simultaneously, $j \in \{0, \cdots, 31\}$. All results are only to three decimal places.

Bit index	ND_7^t			ND_6^t			ND_5^t		
	sen_0	sen_1	$sen_{0,1}$	sen_0	sen_1	$sen_{0,1}$	sen_0	sen_1	$sen_{0,1}$
0	0	0	0	0	0	0	0.001	0.001	0
1	0	0	0	0	0	0	0.004	0.004	0
2	0.001	0.001	0.001	0.028	0.028	0.018	0.168	0.169	0.071
3	0.006	0.006	0.003	0.111	0.111	0.022	0.222	0.221	0.119
4	0.054	0.053	0.004	0.15	0.15	0.017	0.174	0.174	0.028
5	0.056	0.056	0.001	0.006	0.006	0.003	0.142	0.141	0
6	0	0	0	0	0	0	0	0	0
7	0.001	0.001	0	0	0	0	0	0	0
8	0	0	0	0.001	0.001	0.001	0.002	0.003	0.002
9	0.002	0.002	0.001	0.007	0.007	0.007	0.021	0.021	0.005
10	0.006	0.006	0.006	0.039	0.039	0.022	0.205	0.205	0.043
11	0.023	0.023	0.021	0.138	0.138	0.034	0.167	0.167	0.061
12	0.054	0.054	0.022	0.116	0.116	0.041	0.162	0.162	0.037
13	0.053	0.053	0.016	0.084	0.084	0.028	0.098	0.098	0.031
14	0.045	0.045	0	0.075	0.075	0	0.089	0.089	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0.026	0.026	0	0.058	0.058	0
17	0.001	0.001	0	0.082	0.082	0	0.216	0.216	0
18	0.007	0.007	0.001	0.096	0.096	0.018	0.2	0.2	0.071
19	0.014	0.014	0.003	0.134	0.134	0.023	0.223	0.223	0.119
20	0.054	0.054	0.004	0.15	0.15	0.017	0.174	0.174	0.028
21	0.056	0.056	0	0.112	0.112	0	0.142	0.141	0
22	0	0	0	0.001	0.001	0	0	0.001	0
23	0.001	0.001	0	0.002	0.002	0.001	0.002	0.002	0
24	0.002	0.002	0.001	0.009	0.009	0.002	0.018	0.018	0.002
25	0.015	0.015	0.003	0.047	0.047	0.009	0.088	0.089	0.005
26	0.038	0.038	0.011	0.102	0.102	0.022	0.214	0.214	0.043
27	0.058	0.058	0.02	0.154	0.154	0.034	0.188	0.188	0.06
28	0.072	0.072	0.022	0.136	0.136	0.04	0.18	0.18	0.037
29	0.053	0.053	0.016	0.084	0.084	0.028	0.098	0.098	0.031
30	0.045	0.045	0	0.075	0.075	0	0.089	0.089	0
31	0	0	0	0	0	0	0	0	0

Table 7. Accuracies of NDs trained on selected ciphertext bits

Γ	$\{30 \sim 23, 14 \sim 7\}$	$\{30 \sim 23, 21 \sim 17, 14 \sim 7, 5 \sim 1\}$	$\{31 \sim 0\}$
Accuracy	0.5414	0.6065	0.6067

The accuracy of the teacher distinguisher is 0.6067. When all the identified informative bits are considered, the resulted student distinguisher obtains a distinguishing accuracy of 0.6065, which is almost the same as 0.6067. Such an experiment shows that **Definition 3** can help identify all the ciphertext bits that have a significant influence on teacher distinguishers.

5.3 Verification

Attack 3: recover sk_9, sk_{10} of 10-round Speck32/64. By setting $\Gamma = \{30 \sim 23, 14 \sim 7\}$, we have trained two student distinguishers ND_7^s, ND_6^s. Let $c_2 = 0.55$, Table 8 shows the estimation of $p_{2|d_1}$ of two student distinguishers.

Table 8. The estimation of $p_{2|d_1}$ of ND_6^s, ND_7^s against Speck32/64.

ND_6^s	d_1	0	1	2	3	4	5	6	7	8
	$p_{2\|d_1}$	0.5132	0.4057	0.3402	0.3025	0.2817	0.2706	0.265	0.2617	0.2594
ND_7^s	d_1	0	1	2	3	4	5	6	7	8
	$p_{2\|d_1}$	0.3576	0.3230	0.3036	0.2940	0.2893	0.2873	0.2866	0.2863	0.2862

Then this attack is divided into four stages. Table 9 summarizes the attack settings of four stages.

Table 9. Attack settings of the key recovery attack on 10-round Speck32/64. SD_i^t/SD_i^s: the neural aided statistical distinguisher built on ND_i^t/ND_i^s.

stage	SD	β_r	β_w	d	c_2	p_0	p_1	p_2	p_n	N	t	Related keys
1	SD_7^s	0.005	2^{-8}	2	0.55	2^{-2}	0.3576	0.2940	0.2863	22540	6677	$sk_{10}[7 \sim 0]$
2	SD_7^t	0.005	2^{-16}	2	0.55	2^{-2}	0.4183	0.2607	0.2162	5274	1325	sk_{10}
3	SD_6^s	0.001	2^{-14}	1	0.55	2^{-2}	0.5132	0.3402	0.2603	5228	1589	$sk_{10}, sk_9[7 \sim 0]$
4	SD_6^t	0.001	2^{-16}	1	0.55	2^{-2}	0.6785	0.3135	0.1164	830	181	sk_{10}, sk_9

In stage 1, we guess the 8 key bits $sk_{10}[7 \sim 0]$. In stage 2, we guess the complete sk_{10} based on each surviving $sk_{10}[7 \sim 0]$. Based on **Attack 1**, we know the number of surviving sk_{10} is about 2^6. In stage 3, we guess $sk_9[7 \sim 0]$ and test each possible pair $(sk_{10}, sk_9[7 \sim 0])$. After stage 4, it's expected that sk_{10} is recovered and sk_9 has at most 1 wrong bit. Finally, the number of surviving (sk_{10}, sk_9) shouldn't exceed $1 + 16 + (2^{16} - 17) \times 2^{-16} \approx 18$. In stage 2, 3, 4, the required samples are randomly drawn from 22540 samples.

After performing the attack 100 times with 22540 plaintext pairs, we find:

1. The right subkey pair (sk_{10}, sk_9) survives in 99 experiments.
2. In four stages, the average numbers of surviving subkey guesses are 12.53, 50.43, 20.52, and 14.73 respectively. Due to the student distinguishers, the average key space (in stage 1 and 2) is reduced from 2^{16} to $2^8 + 12.53 \times 2^8 \approx 2^{11.76}$. In stage 3 and 4, the average key space is reduced from $51 \times 2^{16} \approx 2^{21.67}$ to $51 \times 2^8 + 20.52 \times 2^8 \approx 2^{14.16}$.
3. In 99 cases, only the pair which has the right sk_{10} survives the attack. In the remining 1 case, 1 pair which the guess of sk_{10} has 1 wrong bit also survives.

6 Applications

Our **NASA** is applied to round-reduced Speck32/64, Speck48/72, Speck48/96. To attack the Speck32/64 or Speck48/96, we need to recover the last 4 subkeys. As for Speck48/72, we need to recover 3 subkeys. This is based on the key schedule inversion algorithm [11].

6.1 Time Complexity

The basic operation of a key recovery attack usually contains two parts: (1) the decryption of a ciphertext with a key guess, (2) the verification of a decrypted ciphertext. In classic cryptanalysis, the verification (eg. finding the right ciphertext pair) is usually much simpler than the decryption. Then the time consumption of the second part is neglected.

For neural aided cryptanalysis, the verification is performed by neural networks that contain massive computations. Table 10 shows the time consumption of NDs against round-reduced Speck32/64. A SIMD-parallelized implementation of Speck32/64 is used.

Table 10. Time consumption of NDs against round reduced Speck32/64. M: the number of tested ciphertexts. t_1: the average time of decrypting M ciphertexts for 1 round. t_2: the average time of distinguishers' verification for M ciphertexts. The CPU is Intel(R) Core(TM) i5-7500. One graphics card (NVIDIA GeForce GTX 1060(6GB)) is used. The number(batch size) of ciphertext pairs fed into a ND each time is 2^{18}. All the results are based on 100 experiments.

M	2^{21}	2^{23}	2^{32}
t_1(seconds)	0.029	0.121	60.392
t_2(seconds, ND^t)	1.779	7.091	3643.392
t_2(seconds, ND^s)	0.945	3.725	1935.36

The time consumption of neural networks is higher than that of the decryption. However, except for the hardware, neural networks can be accelerated by many techniques [17,22]. Some techniques are not applicable to the decryption.

To fairly evaluate the time complexity of the attack on a h-round cipher, we focus on two basic operations: (1) decrypting a ciphertext for h rounds, (2) making a prediction for an input using a ND. If an attack needs to perform the first operation N_1 times, we define N_1 as the Decryption Complexity (DC). If an attack needs to perform the second operation N_2 times, we define N_2 as the Prediction Complexity (PC). The final time complexity is $N_1 + N_2 \times \rho$ where ρ is a constant.

Take Table 10 as an example. One time unit related to DC is $x_1 = \frac{t_1 \times h}{M}$ and one time unit related to PC is $x_2 = \frac{t_2 \times 2}{M}$. We should notice that the input of NDs is a ciphertext pair. Then $\rho = \frac{x_2}{x_1} = \frac{2 \times t_2}{t_1 \times h}$. In this paper, $h \in \{11, 12, 13\}$ and $5 \leqslant \rho \leqslant 11$.

It's clear that N_1, N_2 are determined by the data complexity and the key space. Thus, they are deterministic. Actually, the real time consumption is also determined by N_1, N_2.

6.2 Key Recovery Attacks on Round Reduced Speck32/64.

To attack Speck32/64 reduced to $11, 12, 13$ rounds, the following 8 neural aided statistical distinguishers are built as Table 11 shows.

ND_7^s, ND_6^s, ND_5^s are trained using Algorithm 6. The subscript set of selected ciphertext bits is $\Gamma = \{30 \sim 23, 14 \sim 7\}$. The number of encryption rounds covered by $\Delta P \to \Delta S$ is 2. It is extended to 3 rounds without loss of probability.

Table 11. Eight neural aided statistical distinguishers for attacking Speck32/64. p_2 is selected based on d, Table 2/12. p_1, p_n are estimated with $M = 10^7$(Section 3.5).

SD	$\Delta P \xrightarrow{p_0} \Delta S$	ND	c_2	p_1	d	p_2	p_n
SD_8^t	$(0x211, 0xa04) \xrightarrow{p_0=2^{-6}} (0x0040, 0)$	ND_8^t	0.5	0.5184	1	0.4993	0.4914
SD_7^s		ND_7^s	0.55	0.3576	2	0.2940	0.2863
SD_7^t		ND_7^t	0.55	0.4183	2	0.2607	0.2162
SD_6^s		ND_6^s	0.55	0.5132	1	0.3402	0.2603
SD_6^t		ND_6^t	0.55	0.6785	1	0.3135	0.1161
SD_5^s		ND_5^s	0.55	0.7192	0	0.5498	0.1291
SD_5^t		ND_5^t	0.55	0.8889	0	0.5151	0.0384
SD_4^t		ND_4^t	0.55	0.995	0	0.5065	0.0069

Key recovery attack on 11-round Speck32/64. To recover the last 4 subkeys, the attack is divided into seven stages. Table 12 shows the attack settings.

Table 12. Settings of the key recovery attack on 11-round Speck32/64. EUB: expected upper bound.

$stage$	SD	β_r	β_w	N	key space	related keys	surviving key space
1	SD_7^s	0.005	2^{-8}	$2^{22.44}$	2^8	$sk_{11}[7 \sim 0]$	2^4(Attack 1)
2	SD_7^t	0.005	2^{-16}	$2^{20.28}$	2^{4+8}	sk_{11}	2^6(Attack 1)
3	SD_6^s	0.001	2^{-14}	$2^{20.28}$	2^{6+8}	$sk_{11}, sk_{10}[7 \sim 0]$	2^4(EUB)
4	SD_6^t	0.001	2^{-16}	$2^{17.37}$	2^{4+8}	sk_{11}, sk_{10}	2^4(EUB)
5	SD_5^s	0.001	2^{-12}	$2^{19.43}$	2^{4+8}	$sk_{11}, sk_{10}, sk_9[7 \sim 0]$	2(EUB)
6	SD_5^t	0.001	2^{-16}	$2^{15.89}$	2^{1+8}	sk_{11}, sk_{10}, sk_9	2(EUB)
7	SD_4^t	0.001	2^{-17}	$2^{13.04}$	2^{1+16}	$sk_{11}, sk_{10}, sk_9, sk_8$	2(EUB)

The probability that the right master key survives is about $(1-0.005)^2 \times (1-0.001)^5 \approx 0.985$. In stages 2, 3, 5, the DC is $(2^{21.28} \times (2^{14} + 2^{12}) + 2^{20.43} \times 2^{12}) \times \frac{1}{11} = 2^{32.29}$. the PC is $2^{20.28} \times (2^{14} + 2^{12}) + 2^{19.43} \times 2^{12} = 2^{34.75}$ In stage $1, 4, 6, 7$, the DC and PC are negligible. Thus the total DC is $2^{32.29}$, the total PC is $2^{34.75}$, and the data complexity is $2^{23.44}$.

Key recovery attack on 12-round Speck32/64. Table 13 shows the attack settings.

Table 13. Settings of the key recovery attack on 12-round Speck32/64.

$stage$	SD	β_r	β_w	N	key space	related keys	surviving key space
1	SD_8^t	0.005	2^{-16}	$2^{26.93}$	2^{16}	sk_{12}	2^4(EUB)
2	SD_7^s	0.005	2^{-12}	$2^{22.86}$	2^{4+8}	$sk_{12}, sk_{11}[7 \sim 0]$	2^7(EUB)
3	SD_7^t	0.005	2^{-16}	$2^{20.28}$	2^{7+8}	sk_{12}, sk_{11}	2^8(EUB)
4	SD_6^s	0.001	2^{-16}	$2^{20.41}$	2^{8+8}	$sk_{12}, sk_{11}, sk_{10}[7 \sim 0]$	2^4(EUB)
5	SD_6^t	0.001	2^{-16}	$2^{17.37}$	2^{4+8}	$sk_{12}, sk_{11}, sk_{10}$	2^4(EUB)
6	SD_5^t	0.001	2^{-20}	$2^{16.12}$	2^{4+16}	$sk_{12}, sk_{11}, sk_{10}, sk_9$	2(EUB)

The probability that the right master key survives is about $(1-0.005)^3 \times (1-0.001)^3 \approx 0.982$. The total DC is about $2^{40.35}$, the total PC is about 42.93, and the data complexity is $2^{27.93}$.

Key recovery attack on 13-round Speck32/64. Table 23 shows the attack settings.

Table 14. Settings of the key recovery attack on 13-round Speck32/64.

stage	*SD*	β_r	β_w	N	key space	related keys	surviving key space
1	SD_8^t	0.005	2^{-32}	$2^{27.7}$	2^{32}	sk_{12}, sk_{11}	2^5(EUB)
2	SD_7^t	0.005	2^{-21}	$2^{20.58}$	2^{5+16}	$sk_{12}, sk_{11}, sk_{10}$	2^8(EUB)
3	SD_6^t	0.001	2^{-24}	$2^{17.78}$	2^{8+16}	$sk_{12}, sk_{11}, sk_{10}, sk_9$	2^5(EUB)

The probability that the right master key survives is about $(1-0.005)^2 \times (1-0.001) \approx 0.989$. The total DC is about 2^{58}, the total PC is about $2^{60.7}$ and the data complexity is $2^{28.7}$.

6.3 Key Recovery Attacks on Round Reduced Speck48/X.

To attack round-reduced Speck48/X, we train three teacher distinguishers ND_5^t, ND_4^t, ND_3^t. The number of training sample is 2×10^7. The difference constraint is $\varDelta S = (0x808000, 0x808004)$ [1].

After performing the BST against three teacher distinguishers, we obtain two student distinguishers ND_5^s, ND_4^s by letting $\varGamma = \{47 \sim 32, 23 \sim 8\}$. Let $c_3 = 0.55$, Table 15 shows the estimation of $p_{2|d_1}$ of the five NDs.

Table 15. The estimation of $p_{2|d_1}$ of five neural distinguishers against Speck48/X

ND_3^t	d_1	0	1	2	3	4	5	6	7	$8 \sim 24$
	$p_{2\|d_1}$	0.9871	0.5885	0.3508	0.2158	0.1397	0.095	0.0678	0.051	< 0.05
ND_4^t	d_1	0	1	2	3	4	5	6	7	$8 \sim 24$
	$p_{2\|d_1}$	0.7494	0.5531	0.4189	0.329	0.2688	0.2299	0.2035	0.1849	< 0.18
ND_4^s	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.6555	0.5057	0.4152	0.3588	0.323	0.3	0.2849	0.2747	< 0.27
ND_5^t	d_1	0	1	2	3	4	5	6	7	$8 \sim 24$
	$p_{2\|d_1}$	0.3304	0.2906	0.2629	0.2448	0.2312	0.2226	0.2162	0.2106	< 0.21
ND_5^s	d_1	0	1	2	3	4	5	6	7	$8 \sim 16$
	$p_{2\|d_1}$	0.2942	0.2696	0.2545	0.2454	0.2407	0.2369	0.2354	0.2334	< 0.2334

Based on these five NDs above, the following 5 neural aided statistical distinguishers are built as Table 16 shows.

Key recovery attack on 12-round Speck48/72. Table 17 shows the attack settings.

The probability that the right master key survives is about $(1-0.005) \times (1-0.001)^4 \approx 0.991$. The total DC is about $2^{50.91}$, the total PC is about $2^{53.49}$, and the data complexity is $2^{37.32}$.

Table 16. Five neural aided statistical distinguishers for attacking Speck48/X. $\Delta P \rightarrow \Delta S = 0x400052/0x504200 \rightarrow 0x808000/0x808004$.

SD	p_0	ND	c_3	p_1	d	p_2	p_n
SD_5^s	2^{-12}	ND_5^s	0.55	0.2942	1	0.2545	0.2304
SD_5^t		ND_5^t		0.3304	1	0.2629	0.1999
SD_4^s		ND_4^s		0.6555	1	0.4152	0.2352
SD_4^t		ND_4^t		0.7494	0	0.5531	0.1349
SD_3^t		ND_3^t		0.9871	0	0.5885	0.0102

Table 17. Settings of the key recovery attack on 12-round Speck48/72 with $p_0 = 2^{-12}$.EUB: expected upper bound.

stage	SD	β_p	β_n	N	key space	related keys	surviving key space
1	SD_5^s	0.005	2^{-16}	$2^{36.32}$	2^{16}	$sk_{12}[15 \sim 0]$	2^4(EUB)
2	SD_5^t	0.001	2^{-24}	$2^{35.27}$	2^{4+8}	sk_{12}	2^5(EUB)
3	SD_4^s	0.001	2^{-21}	$2^{31.64}$	2^{5+16}	$sk_{12}, sk_{11}[15 \sim 0]$	2^5(EUB)
4	SD_4^t	0.001	2^{-24}	$2^{31.73}$	2^{5+8}	sk_{12}, sk_{11}	2(EUB)
5	SD_3^t	0.001	2^{-25}	$2^{26.21}$	2^{1+24}	$sk_{12}, sk_{11}, sk_{10}$	2(EUB)

Key recovery attack on 12-round Speck48/96. The 12-round Speck48/96 can be attacked by recovering the last 4 subkeys. The attack setting for 12-round Speck48/72 is adopted for recovering the last 3 subkeys of Speck48/96. We recover sk_9 by adopting a teacher distinguisher ND_2^t. And the extra computation complexity is negligible.

Thus, the total DC is about $2^{50.91}$, the total PC is about $2^{53.49}$, and the data complexity is $2^{37.32}$.

7 Reduce the Data Complexity

NASA is a universal attack model. But some unique properties related to the attacked cipher can be used to enhance NASA. In this section, we provide an instance.

The impact of p_0 on the data complexity is $\mathcal{O}(p_0^{-2})$, which is one of the main bottlenecks for the raw NASA. Actually, we can reduce the data complexity by exploiting neutral bits.

7.1 Overview of the Idea

Similarly, we also take the key recovery attack with 1-round decryption as an example. Figure 2 summarizes the scheme of reducing the data complexity of the NASA.

The core idea is dividing the differential transition into two parts: $\Delta P \rightarrow \Delta S_1$ with a probability of p, $\Delta S_1 \rightarrow \Delta S_2$ with a probability of q. Suppose that $p_0 = p \times q$. The statistical distinguisher only covers the second part.

When we perform a key recovery attack using the statistical distinguisher, the data complexity is calculated based on q and the ND. Let's denote the data

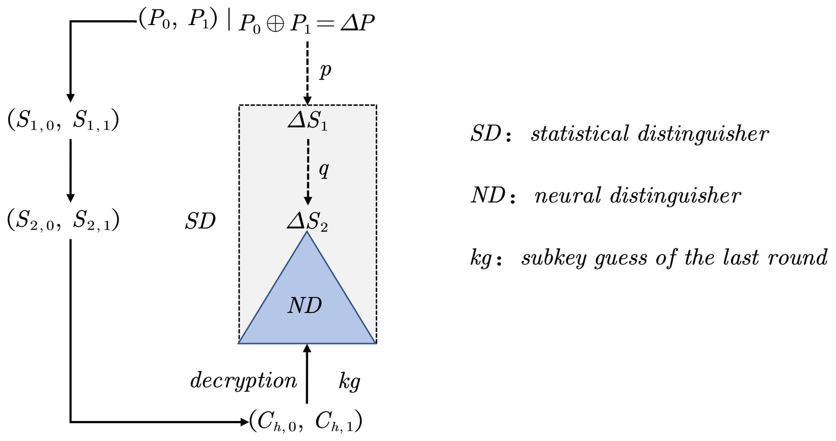


Fig. 2. The scheme of reducing the data complexity of NASA.

complexity as N_1. Now, the first task is collecting N_1 samples that pass the first differential transition $\Delta P \to \Delta S_1$.

To solve this task, we propose a concept named **Homogeneous Set**.

Definition 4 *Consider a differential transition $\Delta P \to \Delta S_1$. A homogeneous set is a special set $\mathcal{S}$ consisting of M plaintext pairs*

$$\mathcal{S} = \{(P_0^0, P_1^0), \cdots, (P_0^M, P_1^M)\}, \quad P_0^i \oplus P_1^i = \Delta P, \quad i \in [1, M] \tag{29}$$

Without loss of generality, we assume that the M plaintext pairs are derived from (P_0^0, P_1^0) using a certain rule. After encryption, these M plaintext pairs can pass the differential transition together in a high probability if (P_0^0, P_1^0) passes the differential transition.

It's clear that neutral bits can be used to generate homogeneous sets. In fact, there are more techniques that are also applicable, such as the affine subspace introduced for improving the differential-linear attack [3]. If the homogeneous set passes the differential, we denote it as a **valid homogeneous set**. Or we denote it as an **invalid homogeneous set**. Then the target of gathering N_1 samples becomes gathering enough valid homogeneous sets.

7.2 Identify the Valid Homogeneous Set

Adopting the idea of NASA, the valid homogeneous set is identified as:

1. Generate a homogeneous set $\mathcal{S}$ randomly.
2. Encrypt $\mathcal{S}$ with the attacked cipher.
3. Collect corresponding ciphertext pairs $(C_0^i, C_1^i), i \in [1, M]$.
4. Decrypt M ciphertext pairs with a subkey guess kg.
5. Feed partially decrypted ciphertext pairs into a ND.
6. Collect the ND's outputs and calculate the statistic T (formula (4)).

After the statistic T is obtained, we can identify homogeneous sets.

Distribution of the statistic under valid homogeneous sets. The probability of the differential $\Delta S_1 \to \Delta S_2$ is q. Then there are about $M \times q$ positive samples and $M \times (1-q)$ negative samples.

When kg is the right subkey, the distribution of the statistic is

$$T_{V_1} = T_1 + T_3 \sim \mathcal{N}(\mu_{V_1}, \sigma_{V_1}) \tag{30}$$

$$\mu_{V_1} = M(qp_1 + (1-q)p_3), \quad \sigma_{V_1} = \sqrt{Mqp_1(1-p_1) + M(1-q)p_3(1-p_3)}. \tag{31}$$

If kg is a wrong subkey guess, the distribution of the statistic is

$$T_{V_0} = T_2 + T_3 \sim \mathcal{N}(\mu_{V_0}, \sigma_{V_0}) \tag{32}$$

$$\mu_{V_0} = M(qp_2 + (1-q)p_3), \quad \sigma_{V_0} = \sqrt{Mqp_2(1-p_2) + M(1-q)p_3(1-p_3)}. \tag{33}$$

For convenience, we denote p_1, p_2 as the same parameter p_V. Then T_{V_1}/T_{V_0} is represented in the same form

$$T_V \sim \mathcal{N}(\mu_V, \sigma_V) \tag{34}$$

$$\mu_V = M[qp_V + (1-q)p_3], \quad \sigma_V = \sqrt{Mqp_V(1-p_V) + M(1-q)p_3(1-p_3)} \tag{35}$$

Distribution of the statistic under invalid homogeneous sets. When $\mathcal{S}$ is an invalid homogeneous set, the distribution of the statistic(formula (4)) is

$$T_I = T_3 \sim \mathcal{N}(\mu_I, \sigma_I) \tag{36}$$

$$\mu_I = Mp_3, \quad \sigma_I = \sqrt{Mp_3(1-p_3)} \tag{37}$$

Distinguishing between T_V and T_I. Since T_V and T_I are two different normal distributions, the technique in Section 3.3 is used to distinguish these two distributions. According to **Proposition 1**, the condition for distinguishing T_V and T_I is

$$\frac{z_{1-\beta_V} \times \sigma_V + z_{1-\beta_I} \times \sigma_I}{\mu_V - \mu_I} = 1 \tag{38}$$

where

$$\begin{aligned} \beta_V &= Pr\left\{s \in \mathcal{N}\left(\mu_I, \sigma_I\right) | s \in \mathcal{N}\left(\mu_V, \sigma_V\right)\right\} \\ \beta_I &= Pr\left\{s \in \mathcal{N}\left(\mu_V, \sigma_V\right) | s \in \mathcal{N}\left(\mu_I, \sigma_I\right)\right\} \end{aligned} \tag{39}$$

For invalid homogeneous sets, the maximum probability that subkey guesses survive the NASA is β_I. For valid homogeneous sets, the minimum probability that subkey guesses survive the NASA is $1 - \beta_V$.

The lower bound of the homogeneous set's size. According to the distinguishing condition presented in Equation (38), the lower bound of the homogeneous set's size M and the decision threshold t_M are

$$a_V = p_V(1-p_V), \quad a_3 = p_3(1-p_3) \tag{40}$$

$$\sqrt{M} = \frac{z_{1-\beta_V} \times \sqrt{qa_V + (1-q)a_3} + z_{1-\beta_I} \times \sqrt{a_3}}{(p_V - p_3) \times q} \tag{41}$$

$$t_M = \mu_V - z_{1-\beta_V}\sigma_V = \mu_I + z_{1-\beta_I}\sigma_I \tag{42}$$

If there are no enough neutral bits, the distinguishing between T_V and T_I will fail. Besides, the choice of p_V is related to $p_{2|d_1}$ as p_2. A deeper analysis about p_V is presented later.

Identify valid homogeneous sets by counting surviving keys. Let $\mathcal{K}$ denote the set of all possible subkey guesses

$$\mathcal{K} = \mathcal{K}_{\mathcal{R}} + \mathcal{K}_{\mathcal{W}}$$

where $\mathcal{K}_{\mathcal{R}}$ is the set of subkey guesses in the target subspace, and $\mathcal{K}_{\mathcal{W}}$ is the set of subkey guesses in other subspaces. Then the lower bound of the number of surviving subkeys is $|\mathcal{K}_{\mathcal{R}}| \times (1-\beta_V)$ when $\mathcal{S}$ is a valid homogeneous set. The upper bound of the number of surviving subkeys is $|\mathcal{K}| \times \beta_I$ when $\mathcal{S}$ is an invalid homogeneous set.

By setting two proper error probabilities β_V, β_I, we ensure the following condition always holds

$$|\mathcal{K}_{\mathcal{R}}| \times (1-\beta_V) >> |\mathcal{K}| \times \beta_I \tag{43}$$

Then we set another decision threshold t_S that satisfies the condition

$$|\mathcal{K}_{\mathcal{R}}| \times (1-\beta_V) \geqslant t_S >> |\mathcal{K}| \times \beta_I \tag{44}$$

It's expected that we can always identify valid homogeneous sets by comparing the number of surviving subkey guesses with t_S. Algorithm 6 summarizes the concrete process of identifying valid homogeneous sets.

Further analysis about p_V. When p_V increases, M (Equation (41)) will decrease since

$$\begin{aligned}\sqrt{M} &= \frac{z_{1-\beta_V} \times \sqrt{qp_V(1-p_V) + (1-q)a_3} + z_{1-\beta_I} \times \sqrt{a_3}}{(p_V - p_3) \times q} \\ &= \frac{z_{1-\beta_V} \times \sqrt{q(1-p_V) + \frac{(1-q)a_3}{p_V}} + \frac{z_{1-\beta_I} \times \sqrt{a_3}}{p_V}}{(\sqrt{p_V} - \frac{p_3}{\sqrt{p_V}}) \times q}.\end{aligned}$$

If p_V increases, the numerator will decrease and the denominator will increase. Then M will decrease.

Algorithm 6 Identify valid homogeneous sets

```
Require: a homogeneous set $\mathcal{S}$ with a size of $M$(formula 23);
    a statistical distinguisher that covers a differential and a neural distinguisher $ND$;
    the posterior probability threshold $c$;
    a decision threshold $t_M$ for filtering subkey guesses;
    a decision threshold $t_S$ for identifying valid homogeneous sets.
Ensure: the classification of the homogeneous set $\mathcal{S}$.
 1: Encrypt $M$ plaintext pairs of the homogeneous set $\mathcal{S}$ and collect the ciphertexts;
 2: Initialize a counter $cp \leftarrow 0$;
 3: for each possible subkey guess $kg$ do
 4:    Decrypt $M$ ciphertext pairs with $kg$;
 5:    Feed partially decrypted ciphertext pairs into the neural distinguisher $ND$;
 6:    Save the outputs of the neural distinguisher, $Z_i, i \in [1, M]$;
 7:    Count the number of $Z_i > c$, and denote it as $T_M$;
 8:    if $T_M > t_M$ then
 9:       $cp \leftarrow cp + 1$;
10:    end if
11: end for
12: if $cp \geqslant t_S$ then
13:    Return 1 ($\mathcal{S}$ is a valid homogeneous set).
14: else
15:    Return 0 ($\mathcal{S}$ is an invalid homogeneous set).
16: end if
```

When the Hamming distance d_1 between the right subkey sk and subkey guess kg increases, $p_{2|d_1}$ will decrease in high probability. But the number of subkey guesses in the subspace may increase sharply when d_1 increases. As long as the condition (formula (44)) holds, we advise $p_V = p_{2|d_1}$ where d_1 should be as small as possible.

8 Improved Applications

8.1 Improved Attacks on Round Reduced Speck32/64

To improve the three key recovery attacks presented in Section 6.2, the 2-round differential is divided into two 1-round differentials:

$$\Delta P = (0x211, 0xa04) \xrightarrow{p=2^{-4}} \Delta S_1 = (0x2800, 0x10) \xrightarrow{q=2^{-2}} \Delta S_2 = (0x0040, 0).$$

Compared with the attacks in Section 6.2, the technique for reducing the data complexity brings two following changes:

1. In each stage, N, t need to be estimated based on q instead of $p_0 = pq$.
2. The total data complexity and computation complexity contains two parts: gathering enough valid homogeneous sets, key recovery.

Table 18. Improved data complexity in each stage of attacks on Speck32/64. Nr: the number of encryption rounds.

Nr	stage	1	2	3	4	5	6	7
11	N	$2^{14.94}$	$2^{12.94}$	$2^{12.2}$	$2^{9.7}$	$2^{11.73}$	$2^{8.78}$	$2^{6.98}$
12		$2^{18.93}$	$2^{15.44}$	$2^{12.94}$	$2^{12.2}$	$2^{9.7}$	$2^{9.03}$	-
13		$2^{19.7}$	$2^{13.28}$	$2^{9.97}$	-	-	-	-

Other settings including the surviving key space are unchanged. Table 18 summarizes the improved data complexity in each stage of three attacks.

Table 19 summarizes the settings for identifying valid homogeneous sets that pass the 1-round differential $\Delta P \to \Delta S_1$.

Table 19. The settings used in Algorithm 6 for identifying valid homogeneous sets.

Nr	ND	c	d	$p_V = p_{2\|d_1=d}$	p_3	β_V	β_I	M	t_M	t_s
11	ND_7^s	0.55	1	0.323	0.2863	0.1	2^{-8}	$2^{15.21}$	11096	8
12	ND_8^t	0.5	1	0.5056	0.4914	0.1	2^{-16}	$2^{19.17}$	291188	16
13	ND_8^t	0.5	1	0.5056	0.4914	0.1	2^{-32}	$2^{20.09}$	553207	16

Taking the first row as an example, we explain the settings presented in Table 19. p_V, p_3 are related to ND_7^s. M is the size of homogeneous sets, which is calculated based on Equation (41). Since $d = 1$, it's expected that the Hamming distance between subkeys belonging to $\mathcal{K}_R$ and the right subkey is $d = 1$. Thus, the lower bound of the number of surviving subkey guesses for valid homogeneous sets is $|\mathcal{K}_R| \times (1 - \beta_V) = 8 \times (1 - 0.1) = 7.2 \approx 8$. Thus, let the decision threshold for distinguishing valid homogeneous sets be $t_S = 8$.

Look at Table 18 and Table 19 simultaneously. We find that $2^{14.94} < 2^{15.21}$, $2^{18.93} < 2^{19.17}$, and $2^{19.7} < 2^{20.09}$. This means that we just need to find *one* valid homogeneous set.

In order to generate homogeneous sets, three sets of probabilistic neutral bits are adopted respectively

$$\begin{aligned}
&\mathcal{B}_1 = \{0, 1, 3 \sim 5, 11, 14, 15, 20 \sim 24, 26 \sim 28\}, |\mathcal{B}_1| = 16;\\
&\mathcal{B}_2 = \{0, 1, 3 \sim 7, 11, 14, 15, 20 \sim 24, 26 \sim 30\}, |\mathcal{B}_2| = 20;\\
&\mathcal{B}_3 = \{0, 1, 3 \sim 8, 11, 14, 15, 20 \sim 24, 26 \sim 30\}, |\mathcal{B}_3| = 21.
\end{aligned} \tag{45}$$

Generate homogeneous sets based on ΔP and $\mathcal{B}_i, i \in [1, 3]$. The corresponding probabilities that a homogeneous set is a valid homogeneous set are about $2^{-4.18}, 2^{-4.75}, 2^{-5.17}$ respectively. These three probabilities are experimentally estimated.

Improved key recovery attack on 11-round Speck32/64. Thus the total theoretical data complexity is

$$N = 2^{4.18} \times M = 2^{4.18+15.21+1} = 2^{20.39} \tag{46}$$

The complexities both contain two parts. The DC of finding a valid homogeneous set is about $2^{20.39} \times 2^8 \times \frac{1}{11} \approx 2^{24.93}$. The corresponding PC is about $2^{27.39}$. It's worth noticing that $sk_{11}[7 \sim 0]$ has already been filtered during the process of gathering the valid homogeneous set. Thus, the DC of recovering the right key is about $2^{23.89}$. The corresponding PC is about $2^{26.35}$. Then the total DC is $2^{24.93} + 2^{23.89} \approx 2^{25.5}$, and the total PC is about $2^{27.96}$.

Practical experiments. To verify the theoretical analysis of the improved key recovery attack on 11-round Speck32/64, we have performed practical experiments according to the attack setting above. The target is to recover sk_{10}, sk_{11}, which contains the first 4 stages. For each experiment, we first find a valid homogeneous set using algorithm 6. After performing this experiment 100 times, we find

1. In 48 experiments, there are no survivng subkey guess pairs (kg_{10}, kg_{11}).
2. In the remaining 52 experiments, the right subkeys (sk_{10}, sk_{11}) survives.
3. If we consider all the 100 experiments, the average numbers of surviving subkey guesses are 6.38 (stage 1), 24.31 (stage 2), 13.93 (stage 3), 9.8 (stage 4). If we focus on the 52 experiments, the average numbers of surviving subkey guesses are 11.01 (stage 1), 46.03 (stage 2), 26.46 (stage 3), 18.84 (stage 4).

The sampling randomness is a very important point. When we generate homogeneous sets with neutral bits, the sampling randomness is likely to be decreased when the first differential covers 1 founds. Thus, for a valid homogeneous set, resulting N intermediate state pairs $(S^i_{1,0}, S^i_{1,1}), i \in \{1, \cdots, N\}$ will be very close to each other in the high-dimensional space.

This will cause a chain reaction. First, corresponding ciphertext pairs may also be close to each other. When we decrypt these ciphertext pairs using several subkey guesses, partially decrypted ciphertext pairs may also possess the similarity. The resulting negative influence is that these subkey guesses will simultaneously survive the attack or be filtered in a high probability.

Although this negative influence exists, it doesn't break the correctness of our improved **NASA**. This is based on following clues

1. The estimation of data complexity for recovering the right subkey is accurate.
2. The number of surviving keys basically meets expectations when all the 100 experiments are considered.

We have performed more experiments about the negative influence presented above. Generally speaking, the negative influence will be alleviated (even tackled) when the first differential or the statistical distinguisher covers more rounds. Thus when we estimate theoretical complexities of attacks based on the improved **NASA**, it's acceptable to neglect the negative influence.

Improved key recovery attack on 12-round Speck32/64. The total theoretical data complexity is

$$N = 2^{4.75} \times M \times 2 = 2^{4.75+19.17+1} = 2^{24.92} \tag{47}$$

The DC of finding a valid homogeneous set is about $2^{24.92} \times 2^{16} \times \frac{1}{12} \approx 2^{37.34}$. The DC of the key recovery is about $2^{32.39}$. Thus the total DC is $2^{37.34} + 2^{32.39} \approx 2^{37.39}$, the total PC is about $2^{39.97}$.

Improved key recovery attack on 13-round Speck32/64. Thus the theoretical data complexity is

$$N = 2^{5.17} \times M \times 2 = 2^{5.17+19.7+1} = 2^{25.87} \tag{48}$$

The DC of finding a valid homogeneous set is about $2^{25.87} \times 2^{32} \times \frac{2}{13} \approx 2^{55.17}$. The DC of the key recovery is about $2^{32.65}$. Thus the total DC is $2^{55.17} + 2^{32.65} \approx 2^{55.17}$, and the total PC is about $2^{57.87}$.

8.2 Improved Attacks on Round Reduced Speck48/X

To improve the two attacks presented in Section 6.3, the 5-round differential is divided into a 1-round differential and 4-round differential

$$\begin{aligned}
\varDelta P = (0x400052, 0x504200) &\xrightarrow{p=2^{-5}} \varDelta S_1 = (0x820200, 0x1202) \\
\varDelta S_1 = (0x820200, 0x1202) &\xrightarrow{q=2^{-7}} \varDelta S_2 = (0x808000, 0x808004)
\end{aligned} \tag{49}$$

Based on $q = 2^{-7}$, we estimate the data complexity in each stage of Table 17 again. The improved data complexities are $2^{26.32}, 2^{25.27}, 2^{21.64}, 2^{21.74}, 2^{16.36}$ respectively.

Improved Key Recovery Attack on 12-round Speck48/72 The student distinguisher ND_5^s is adopted for identifying valid homogeneous sets.

The following settings are used to run Algorithm 6

$$c = 0.55,\ \ d = 1,\ \ p_V = p_{2|d_1=d} = 0.2696,\ \ p_3 = 0.2304,\ \ \beta_V = 0.1,\ \ \beta_I = 2^{-16}$$

Besides, $M = 56194311 \approx 2^{25.74}$.

Since $M > 2^{25}$, we need to select 26 neutral bits for generating a homogeneous set. Since $M < 2^{26.32} < 2 \times M$, we need to gather 2 valid homogeneous sets. However, by exploiting a plaintext bit which has a neutrality of 1, we can generate $2^{26.32}$ samples with 1 valid homogeneous set. Let the set of probabilistic neutral bits be

$$\mathcal{B}_4 = \{0 \sim 4, 10 \sim 12, 18, 19, 21, 23, 25 \sim 27, 29, 31 \sim 37, 42 \sim 44\}.$$

The probability that a homogeneous set is a valid homogeneous set is about $2^{-5.62}$.

It's worth noticing that the ninth plaintext bit has a neutrality of 1. After one valid homogeneous set is found, we generate the left $2^{26.32} - M$ samples by flipping the ninth plaintext bit of M plaintext pairs. Thus the total data complexity is

$$N = 2^{5.62} \times M \times 2 + (2^{26.32} - M) \times 2 \approx 2^{32.37} \tag{50}$$

The DC of finding a valid homogeneous set is about $2^{5.62} \times M \times 2 \times 2^{16} \times \frac{1}{12} \approx 2^{44.78}$. The DC of the key recovery is about $2^{40.58}$. Thus the total DC is $2^{44.78} + 2^{40.58} \approx 2^{44.86}$, the total PC is about $2^{47.44}$.

Improved Key Recovery Attack on 12-round Speck48/96. All the attack settings related to the attack on 12-round Speck48/72 is applied to this attack. Thus, the data complexity is $N \approx 2^{32.37}$, the total DC is about $2^{44.86}$, and the total PC is about $2^{47.44}$.

9 Conclusions

In this paper, we propose a neural aided statistical attack (NASA) that is a universal framework for neural aided cryptanalysis. It has no extra requirements about the attacked cipher except for a high probabilistic differential transition. Besides, it provides a theoretical framework for estimating the needed attack complexities and success rate. In order to reduce the key space to be searched in the key recovery attack, a bit sensitivity test is proposed to help build NDs flexibly. Applications to round reduced Speck and DES have proved the correctness and superiorities of our NASA. Special properties related to the attacked cipher can also be used to improve NASA. Experiments based on neutral bits prove it successfully. This implies that NASA is a cryptanalysis technique with great potential.

But there is still more work to do. The first is exploring the properties of the ND. If we know how to build NDs against more rounds, the statistical attack model can be greatly improved. The second is understanding the knowledge learned by the ND. It can help us get rid of the dependence on neural distinguishers and improve the theoretical framework. The practical time consumption can also be reduced significantly. Besides, the concept of the informative bit is full of potential. We believe neural aided cryptanalysis has great potential for better assessing the security of ciphers.

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential cryptanalysis of round-reduced simon and speck. In: International Workshop on Fast Software Encryption. pp. 525–545. Springer (2014)
2. Batina, L., Bhasin, S., Jap, D., Picek, S.: Poster: Recovering the input of neural networks via single shot side-channel attacks. computer and communications security pp. 2657–2659 (2019)

3. Beierle, C., Leander, G., Todo, Y.: Improved differential-linear attacks with applications to arx ciphers. In: Annual International Cryptology Conference. pp. 329–358. Springer (2020)
4. Benamira, A., Gerault, D., Peyrin, T., Tan, Q.Q.: A deeper look at machine learning-based cryptanalysis. IACR Cryptol. ePrint Arch **287**, 2021 (2021)
5. Bengio, Y., Ducharme, R., Vincent, P.: A neural probabilistic language model. Neural Information Processing Systems (NeurIPS) pp. 932–938 (2000)
6. Biryukov, A., Roy, A., Velichkov, V.: Differential analysis of block ciphers simon and speck. In: International Workshop on Fast Software Encryption. pp. 546–570. Springer (2014)
7. Bogdanov, A., Wang, M.: Zero correlation linear cryptanalysis with reduced data complexity. fast software encryption pp. 29–48 (2012)
8. Castelvecchi, D.: Can we open the black box of ai? Nature News **538**(7623), 20 (2016)
9. Chen, Y., Yu, L., Ota, K., Dong, M.: Robust activity recognition for aging society. IEEE Journal of Biomedical and Health Informatics **22**(6), 1754–1764 (2018)
10. Dayhoff, J.E., DeLeo, J.M.: Artificial neural networks: opening the black box. Cancer: Interdisciplinary International Journal of the American Cancer Society **91**(S8), 1615–1635 (2001)
11. Dinur, I.: Improved differential cryptanalysis of round-reduced speck. international conference on selected areas in cryptography pp. 147–164 (2014)
12. Eli, B., Rafi, C.: Near-collisions of sha-0. Annual International Cryptology Conference pp. 290–305 (2004)
13. Feller, W.: An introduction to probability theory and its applications. vol. ii. Population **23**(2), 375 (1968)
14. Gisselquist, R., Hoel, P.G., Port, S.C., Stone, C.J.: Introduction to probability theory. American Mathematical Monthly **81**(9), 1041 (1974)
15. Gohr, A.: Improving attacks on round-reduced speck32/64 using deep learning. international cryptology conference pp. 150–179 (2019)
16. Greydanus, S.: Learning the enigma with recurrent neural networks. arXiv: Neural and Evolutionary Computing (2017)
17. Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M.A., Dally, W.J.: Eie: efficient inference engine on compressed deep neural network. ACM SIGARCH Computer Architecture News **44**(3), 243–254 (2016)
18. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: Make some noise: Unleashing the power of convolutional neural networks for profiled side-channel analysis. cryptographic hardware and embedded systems **2019**(3), 148–179 (2019)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems (NeurIPS) pp. 1097–1105 (2012)
20. MacKay, D.J.: Introduction to gaussian processes. NATO ASI Series F Computer and Systems Sciences **168**, 133–166 (1998)
21. Olden, J.D., Jackson, D.A.: Illuminating the "black box": a randomization approach for understanding variable contributions in artificial neural networks. Ecological modelling **154**(1-2), 135–150 (2002)
22. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems. pp. 8026–8037 (2019)
23. Ronald, L.R.: Cryptography and machine learning. International Conference on the Theory and Application of Cryptology pp. 427–439 (1991)

A Analysis of the Data Complexity

For the conventional differential cryptanalysis, the data complexity is only related to the differential. For the **NASA**, the data complexity is also related to the neural distinguisher. In this appendix, we present the analysis of each part's impact on the data complexity.

A.1 The Differential's Impact on the Data Complexity

According to formula (17), the data complexity N is affected by the probability p_0 of the differential transition as

$$\begin{aligned}\sqrt{N} &= \frac{z_{1-\beta_r} \times \sqrt{p_0 a_1 + (1-p_0)a_3} + z_{1-\beta_w} \times \sqrt{p_0 a_2 + (1-p_0)a_3}}{(p_1 - p_2) \times p_0} \\ &\propto \frac{z_{1-\beta_r} \times \sqrt{a_3 + (a_1 - a_3)p_0} + z_{1-\beta_w} \times \sqrt{a_3 + (a_2 - a_3)p_0}}{p_0} \\ &\propto \frac{\sqrt{a_3 + (a_1 - a_3)p_0} + a_4 \times \sqrt{a_3 + (a_2 - a_3)p_0}}{p_0}\end{aligned}$$

where $a_4 = \frac{z_{1-\beta_w}}{z_{1-\beta_r}}$. We further know

$$N \propto p_0^{-2}\left[\boldsymbol{a_3 + a_4^2 a_3} + (a_1 - a_3 + a_4^2 a_2 - a_4^2 a_3)p_0 + a_5\right]$$

where $a_5 = 2 \times a_4 \times \sqrt{a_3 + (a_1 - a_3)p_0} \times \sqrt{a_3 + (a_2 - a_3)p_0}$. Thus the impact of the probability p_0 of the differential transition is $\mathcal{O}(p_0^{-2})$.

A.2 The Neural Distinguisher's Impact on the Data Complexity

Three probabilities p_1, p_2, p_n are related to the neural distinguisher. Since p_n is related to negative samples and p_1, p_2 are related to positive samples, we can discuss p_n separately.

The impact of $\boldsymbol{p_n}$. In formula (17), only a_3 is related to p_n.

$$\begin{aligned}\sqrt{N} &= \frac{z_{1-\beta_r} \times \sqrt{p_0 a_1 + (1-p_0)a_3} + z_{1-\beta_w} \times \sqrt{p_0 a_2 + (1-p_0)a_3}}{(p_1 - p_2) \times p_0} \\ &\propto \sqrt{p_0 a_1 + (1-p_0)a_3} + a_4 \times \sqrt{p_0 a_2 + (1-p_0)a_3} \\ \Rightarrow N &\propto p_0 \times a_1 + a_4^2 \times p_0 \times a_2 + (1-p_0)(1+a_4^2)a_3 + a_5\end{aligned}$$

Next, we will focus on attack scenarios where the p_0 is very low. This can make the discussion easier and more concise. This simplification is also reasonable. Because when we attack a cipher for more rounds, the probability of the differential transition is generally small.

When $p_0 \to 0$, $a_5 \to 2 \times a_4 \times a_3$. Thus the impact of a_3 on the data complexity is $\mathcal{O}(a_3)$. Since $p_n < 1$ always holds and $a_3 = p_n - p_n^2$, the impact of p_n on the data complexity is also $\mathcal{O}(p_n)$.

The impact of p_1, p_2. In formula (17), a_1, a_2 are related to p_1, p_2 respectively. The impacts of a_1, a_2 are adjusted by the differential transition's probability p_0. Due to this property, we can also focus on attack scenarios where $p_0 \to 0$.

$$\begin{aligned}\sqrt{N} &= \frac{z_{1-\beta_r} \times \sqrt{p_0 a_1 + (1-p_0)a_3} + z_{1-\beta_w} \times \sqrt{p_0 a_2 + (1-p_0)a_3}}{(p_1 - p_2) \times p_0} \\ &\approx \frac{z_{1-\beta_r} \times \sqrt{(1-p_0)a_3} + z_{1-\beta_w} \times \sqrt{(1-p_0)a_3}}{(p_1 - p_2) \times p_0} \propto (p_1 - p_2)^{-1}\end{aligned}$$

Thus the impact of p_1, p_2 on the data complexity is $\mathcal{O}((p_1 - p_2)^{-2})$.