Conditional Differential-Neural Cryptanalysis

Zhenzhen Bao[1], Jian Guo[1], Meicheng Liu[2], Li Ma[2], and Yi Tu[1]

[1] Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore.
{zzbao,guojian}@ntu.edu.sg, tuyi0002@e.ntu.edu.sg

[2] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, China.
{meicheng.liu,skloismary}@gmail.com

Abstract. In CRYPTO 2019, Gohr introduced deep learning into cryptanalysis, and for the first time successfully applied it to key recovery attacks on Speck32/64 reduced to 11 and 12 rounds, with complexities comparable with traditional differential cryptanalysis. In this paper, we introduce the technique of generalized neutral bits into Gohr's framework, and successfully mount the first practical key recovery attacks against 13-round Speck32/64 with time 2^{48} and data 2^{29} for a success rate of 0.21. Compared against the best differential attacks in literature with time 2^{51} for 12 rounds or impractical time 2^{57} on a single GPU for 13 rounds, the full implementation of our 13-round attack is able to complete execution within 3 days. We also extend the framework to Simon32/64, and reduce the data complexity for the practical 16-round attack from 1/6 of the codebook to 2^{21}. This is arguably the first time to witness deep learning based cryptanalysis having a considerable advantage over traditional methods.

Keywords: Neural Distinguisher, Key Recovery Attack, Differential Cryptanalysis, Simon, Speck, Generalized Neutral Bits, Bayesian Search

1 Introduction

Whether machine learning is useful for cryptanalysis has been a long standing open problem to the cryptography research community, especially given its breakthrough in applications like image classification, autonomous vehicles, and board games like chess. In CRYPTO 2019, Gohr [8] made the first successful attempt to Speck32/64, where complexities comparable to that by the traditional cryptanalysis were obtained for 11 and 12 rounds. Gohr's attack tries to mimic the traditional differential cryptanalysis, where the underlying distinguisher with a high probability differential was replaced by a neural-network based distinguisher and the key recovery phase by Bayesian search. It was observed that, the success rate for a pure neural distinguisher (ND) will decrease drastically when rounds increase, Gohr overcame this problem by training a short ND and prepending a traditional high probability differential path (DP) to it. This approach is similar to differential-linear cryptanalysis [15]. The DP and ND have to connect to each other, *i.e.*, the output difference of DP has to match the expected input difference of ND, and the two essentially form a long differential path. Due to the prepended DP with a probability, not every pair of input will follow the DP. These data will be not usable and become noise, and hence reduce the success rate of the succeeding ND. Neutral bits were introduced to group the conforming pairs of the DP, thus increase the density at a local point. They are neutral in the sense that a given conforming pair of the DP will result in additional conforming pairs with probability one at no cost by flipping the neutral bits. The increased success rate in turn reduces the overall attack complexity and extends the attack to even more rounds.

The idea of using neutral bits to boost the conforming pairs of differentials can be tracked back to 2004 by Biham and Chen [3] on a collision attack against SHA-0. This was extended to probabilistic neutral bits in [5], where the additional conforming pairs are valid with some probability after flipping the neutral bits. The essential effect is similar to auxiliary differential paths [13] used to attack SHA-1, where the additional conforming pairs are valid when the auxiliary differential paths are followed. In 2010 [14], conditional differential was proposed for cryptanalysis of NLFSR-based crypto-systems.

In this paper, we extend Gohr's attack framework in the following ways.

– Firstly, in addition to single neutral bit used in [8], we find there exists a more sophisticated combination of bits, which allows to find additional conforming pairs by flipping the set simultaneously, and we call them "simultaneous neutral bit-set" (SNBS).
– Secondly, in addition to neutral bits that find the additional conforming pairs with probability one, we are also interested in those with lower probabilities. Some of the probabilistic transitions can be fulfilled deterministically by prefixing some bits in the plaintext chosen, which brings the probability of the neutral bits (close) to one. We call it "conditional neutral bit" (CNB), or "conditional simultaneous neutral bit-set" when the condition is on a set of simultaneous neutral bits.
– Thirdly, we note the output difference of DP matters to ND, but not the input difference. Hence, more than one differential paths can be prepended to ND, as long as they share the same output difference. Surprisingly, some neutral bits can be shared by multiple such differential paths.
– Lastly, besides the Residual Network (ResNet) [10] considered by Gohr, other neutral networks developed in recent years have also demonstrated their advantages. Dense Network (DenseNet) [12] shows advantages in parameter efficiency, implicit deep supervision, and feature reuse. Squeeze-and-Excitation Network (SENet) [11], which won the first place in the *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC 2017) classification task, reduced the Top-5 error to 2.251%. It can also be combined with existing deep architectures to boost their performance at minimal additional computational cost. On top of the choices of different neutral networks, the ND can also be trained by different ways including key-averaging and staged training, in this paper we pick the best combination with the highest accuracy.

Both SPECK32/64 and SIMON32/64 have a block size of 32 bits, which are small enough for us to find *all* SNBS (flipping simultaneously up to 4 bits) in a bruteforce search or in an algebraic method basing on Gröbner basis. For finding CNB, the manual analysis supported by experiments was applied. Together with a search of multiple DPs using a solver named CryptoSMT [16], we find the first practical key recovery attack against 13-round SPECK32/64 and also achieved lower data complexities for 12-round SPECK32/64 and 16-round SIMON32/64 when compared with the best attacks by traditional methods. These results are summarized in Table 1.

Table 1: Summary of key-recovery attacks on SPECK32/64 and SIMON32/64

Target	#R	Time (GPU h)	Time (CPU h)	Time (#Enc)	Data	Succ. Rate	Dist.	Ref.
SPECK32/64	11	-	-	2^{46}	2^{14}	-	DD	[6]
		-	0.139	2^{38*}	$2^{13.6}(2^{14.5})$	0.52 (1)	ND	[8]
	12	-	-	2^{51}	2^{19}	-	DD	[6]
		<1	12	$2^{43.40*}$	$2^{22.97}$	0.40	ND	[8]
		-	4	$2^{41.81*}$	$2^{18.6}$	0.32	ND	Sect. 4.3
	13	-	-	2^{57}	2^{25}	-	DD	[6]
		64	-	$2^{45.81^*+r}$	2^{29}	0.21	ND	Sect. 4.2
		32	-	$2^{44.81^*+r}$	2^{28}	0.12	ND	Sect. 4.2
SIMON32/64	16	-	-	$2^{26.48}$	$2^{29.48}$	0.62	DD	[1]
		4	-	$2^{41.81^*+r}$	2^{21}	0.49	ND	Sect. 6.2

* Under the assumption that one second equals the time of 2^{28} executions of SPECK32/64 or SIMON32/64 on a CPU.
r : $\log_2(cpu/gpu)$, where *cpu* is the CPU time and *gpu* is the GPU time running an attack. In our computing systems, $r = 2.4$ (The worse case execution time of the core of the 12-round attack on SPECK32/64 (without guessing the one key bit of k_0) took 6637 seconds on CPU and 1265 seconds on GPU).

ORGANIZATION. The rest of the paper is organized as follows. Section 2 gives the preliminary on machine learning based differential cryptanalysis, and introduces the design of SIMON and SPECK. Section 3 introduces the generalized neutral bits technique and the framework of our conditional differential-neutral cryptanalysis. The applications to SPECK32/64 and SIMON32/64 are presented in Section 4 and 6, respectively. Besides, Section 5 presented various of neural distinguishers on SIMON32/64 reduced up to 11 rounds. Section 7 concludes the paper.

2 Preliminary

2.1 Brief Description of Speck and Simon

Notations. Denote by n the word size in bits, $2n$ the state size in bits. Denote by (x_r, y_r) the left and right branches of a state after the encryption of r rounds. Denote by $x[i]$ (resp. $y[i]$) the i-th bit of x (resp. y) counted starting from 0; Denote by $[j]$ the index of the j-th bit of the state, *i.e.*,, the concatenation of x and y, where $y[0]$ is the 0-th bit, and $x[0]$ is the 16-th bit. Denotes by $\oplus$ the bit-wise XOR, $\boxplus$ the addition modulo 2^n, $\cdot$ or & the bit-wise AND, $x^{\lll s}$ or $x \lll s$ the bit-wise left rotation by s positions, $x^{\ggg s}$ or $x \ggg s$ the bit-wise right rotation by s positions. Denote by F_k (resp. F_k^{-1}) the round function (resp. inverse of the round function) using subkey k of the encryption.

Brief Description of Speck32/64 **and** Simon32/64**.** Speck32/64 and Simon32/64 are small members in the lightweight block cipher family Speck and Simon [2] designed by researchers from the National Security Agency (NSA) of the USA. Both Speck32/64 and Simon32/64 are of Feistel constructions, has 32-bit block and 64-bit key. The round functions use combinations of rotation, XOR, and addition modulo 2^{16} (Speck) or bit-wise AND (Simon). Speck32/64 has 22 rounds and Simon32/64 has 32 rounds. The encryption algorithm of Speck32/64 and Simon32/64 are listed in Algorithms 1 and 2. The subkeys of 16-bit for each round are generated from a master key of 64-bit by the non-linear key schedule using the same round function (Speck32/64), or linear functions of simple rotation and XOR (Simon32/64).

Algorithm 1: Encryption of Simon32/64

```
  Input: $P = (x_0, y_0)$ and $\{k_0, \cdots, k_{31}\}$
  Output: $C = (x_{32}, y_{32})$
1 for $r = 0$ to 31 do
2     $x_{r+1} \leftarrow (x_r^{\lll 1} \cdot x_r^{\lll 8}) \oplus x_r^{\lll 2} \oplus y_r \oplus k_r$
3     $y_{r+1} \leftarrow x_r$
4 end
```

Algorithm 2: Encryption of Speck32/64

```
  Input: $P = (x_0, y_0)$ and $\{k_0, \cdots, k_{21}\}$
  Output: $C = (x_{22}, y_{22})$
1 for $r = 0$ to 21 do
2     $x_{r+1} \leftarrow x_r^{\ggg 7} \boxplus y_r \oplus k_r$
3     $y_{r+1} \leftarrow y_r^{\lll 2} \oplus x_{r+1}$
4 end
```

2.2 Differential-based Neural Distinguishers

The work in [8] shows that neural network could be trained to capture the non-randomness of the distribution of values of output pairs when the input pairs to round-reduced Speck32/64 are of specific difference, and thus play the role of distinguisher in cryptanalysis. Although, this is still in the framework of the traditional differential cryptanalysis, the employed neural network is a non-traditional component. This differential-based neural distinguisher is the first known machine learning model that successfully performed cryptanalysis task on modern ciphers (beyond the applications on side-channel attacks).

In the following, the way of training the differential-based neural distinguisher introduced in [8] is briefly recalled.

The Training Data and Input Representation. For a target cipher, the neural network is to be trained to distinguish between examples of ciphertext pairs corresponding to plaintext pairs with particular difference and those corresponding to random plaintext pairs. Thus, each of the training data is a data pair of the form (C, C') together with a label taking a value 0 or 1, where 0 means the corresponding plaintext pair is generated randomly, and 1 from a particular plaintext difference Δ_I. For SPECK32/64, the Δ_I is chosen to be of a single active bit, *i.e.*, $(\texttt{0x0040}, \texttt{0000})$, which is the intermediate difference lying in a known best differential characteristic.

The state of SPECK32/64 has left and right parts, thus, a pair of data is transformed into a quadruple of words (x, y, x', y') where $C = x \| y$ and $C' = x' \| y'$. The word quadruple is then interpreted into a 4×16-matrix with each word as a row-vector before fed into the neural network with an input layer consists of 64 units. Among the set of training data and verification data, half are positive examples labelled by 1, and the other half are negative examples labelled by 0.

Training Schemes. The neural network structure used in [8] is a deep residual network (refer to [7, 8] for more details). There are three training schemes proposed in [8]. The first is a basic training scheme that is sufficient for successfully training short round distinguishers (*i.e.*, 5-round and 6-round). The second is an improved training scheme for r-round distinguishers that simulate the output of the KEYAVERAGING algorithm used with an $(r-1)$-round distinguisher. Using the second scheme, the best neural distinguisher on 7-round SPECK32/64 was achieved in [8]. The third is a staged training method that turns an already trained $(r-1)$-round distinguisher into an r-round distinguisher in several stages. In the first stage, the positive data examples are output pairs of $(r-i)$-round, which correspond to input pairs with difference that appears with the highest probability from Δ_I after i rounds, where i is small, *e.g.*, $i = 3$. In the latter stages, the positive data examples are the output pairs of r-round, which correspond to input pairs with difference Δ_I, while the learning rate drops from stage to stage. Using the third scheme, the longest neural distinguisher on SPECK32/64, which is an 8-round one was achieved.

2.3 Upper Confidence Bounds and Bayesian Optimization

Besides a basic key-recovery attack, an improved attack using both elements in specifics of the targeted cipher and elements in reinforcement learning was proposed in [8].

The improved key-recovery attack employs an r-round main and an $(r-1)$-round helper neural distinguisher trained with data pairs corresponding to input pairs with difference Δ_I; a short s-round differential, $\Delta_{I'} \to \Delta_I$ with probability denoted by 2^{-p}, is prepended on top of the neural distinguishers (refer to Fig. 1 for an illustration of the components of the key-recovery attack.) About $c \cdot 2^p$ (denoted by n_{cts}) data pairs with difference $\Delta_{I'}$ are randomly generated (where c is a small constant); Neutral bits of the s-round differential are used to expand each data pair to a structure of n_b data pairs. The resulted n_{cts} structures of data pairs are decrypt by one round with 0 as the subkey to get plaintext structures. All plaintext structures are queried to obtain the corresponding ciphertext structures.

Each ciphertext structure is to be used to generate candidates of the last subkey by the r-round main neural distinguisher (and latter of the second to last subkey by the $(r-1)$-round helper neural distinguisher) with a highly selective key search policy based on a variant of Bayesian optimization.

More specifically, the key search policy depends on an important observation that the expected response of the distinguisher upon wrong-key decryption will depend on the bitwise difference between the trial key and the real key. This *wrong key response profile*, which can be precomputed, is used to recommend new candidate values for the key from previous candidate values with minimizing the weighted Euclidean distance as the criteria in an BayesianKeySearch Algorithm 3. It recommends a set of subkeys and provides their scores without exhaustively performing trail decryptions.

The use of ciphertext structures is also highly selective using a standard exploration-exploitation technique, namely *Upper Confidence Bounds* (UCB). Each ciphertext structure is assigned a priority according to the scores of the subkeys they recommended and the visited times of them.

An important detail in the BayesianKeySearch Algorithm 3 is that the responses from the neural distinguisher on ciphertext pairs in the ciphertext structure are combined using the Formula 1 and used as the score of the recommended subkey. This score is highly decisive for the execution time and success rate of the attack. It will determine whether the recommended subkey will be further treated as it score pass or fail to pass the cutoff and also determine the priority of ciphertext structures to be visited. The number of ciphertext pairs in each structure is decisive when the neural distinguisher has a low accuracy.

$$s_k := \sum_{i=0}^{n_b-1} \log_2\left(\frac{v_{i,k}}{1-v_{i,k}}\right) \tag{1}$$

3 Deep Exploring of Neutral Bits

3.1 Motivation of Neutral Bits

In general, the more rounds the neural distinguisher covers, the lower its accuracy. Theoretically, a neural distinguisher with accuracy higher than 0.5 means

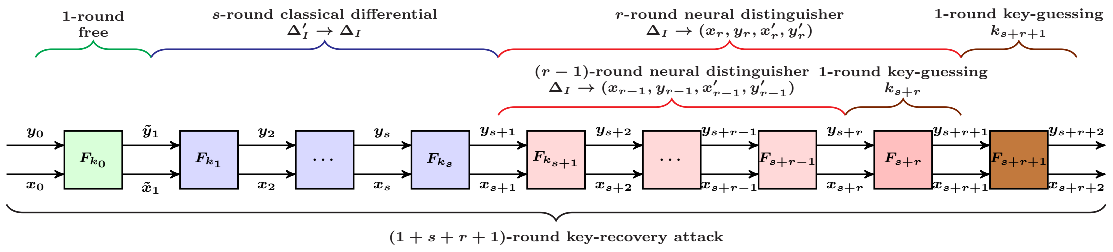

1-round free
s-round classical differential $\Delta'_I \to \Delta_I$
r-round neural distinguisher $\Delta_I \to (x_r, y_r, x'_r, y'_r)$
1-round key-guessing k_{s+r+1}
$(r-1)$-round neural distinguisher $\Delta_I \to (x_{r-1}, y_{r-1}, x'_{r-1}, y'_{r-1})$
1-round key-guessing k_{s+r}
y_0 x_0 F_{k_0} $\tilde{y}_1$ $\tilde{x}_1$ F_{k_1} y_2 x_2 $\cdots$ y_s x_s F_{k_s} y_{s+1} x_{s+1} $F_{k_{s+1}}$ y_{s+2} x_{s+2} $\cdots$ y_{s+r-1} x_{s+r-1} F_{s+r-1} y_{s+r} x_{s+r} F_{s+r} y_{s+r+1} x_{s+r+1} F_{s+r+1} y_{s+r+2} x_{s+r+2}
$(1+s+r+1)$-round key-recovery attack

Fig. 1: Components of the key-recovery attacks

Algorithm 3: BayesianKeySearch Algorithm [8]

```
/* The description of this BayesianKeySearch Algorithm in [8] has a
   small typo and is inconsistent with that in the implementation
   codes [7], the description here corrects it according to [7]. */
Input: Ciphertext structure $\mathcal{C} := \{C_0, \cdots, C_{n_b-1}\}$, a neural distinguisher $\mathcal{ND}$,
       and its wrong key response profile $\mu$ and $\sigma$, the number of candidates
       to be generated within each iteration $n_{cand}$, the number of iterations
       $n_{byit}$
Output: The list $L$ of tuples of recommended keys and their scores
1  $S := \{k_0, k_1, \ldots, k_{n_{cand}-1}\} \leftarrow$ choose $n_{cand}$ values at random without
   replacement from the set of all subkey candidates.
2  $L \leftarrow \{\}$
3  for $t = 1$ to $n_{byit}$ do
4      for $\forall k_i \in S$ do
5          for $j = 0$ to $n_b - 1$ do
6              $C'_{j,k_i} = F^{-1}_{k_i}(C_j)$
7              $v_{j,k_i} = \mathcal{ND}(C'_{j,k_i})$
8              $s_{j,k_i} = \log_2(v_{j,k_i}/(1 - v_{j,k_i}))$
9          end
10         $s_{k_i} = \sum_{j=0}^{n_b-1} s_{j,k_i}$ ;                /* the combined score of $k_i$ */
11         $L \leftarrow L||(k_i, s_{k_i})$
12         $m_{k_i} = \sum_{j=0}^{n_b-1} v_{j,k_i}/n_b$
13     end
14     for $k \in \{0, 1, \cdots, 2^{16} - 1\}$ do
15         $\lambda_k = \sum_{i=0}^{n_{cand}-1} (m_{k_i} - \mu_{k_i \oplus k})^2 / \sigma^2_{k_i \oplus k}$
16     end
17     $S \leftarrow \text{argsort}_k(\lambda)[0 : n_{cand} - 1]$ ;     /* Pick $n_{cand}$ keys with the $n_{cand}$
        smallest score to form the new set of candidate keys $S$ */
18 end
19 return $L$
```

some distinguishing advantage over a random distinguisher. However, when the accuracy being marginally higher than 0.5, it is hard to be used in practical key recovery attack. Thus, Gohr in [8] used the combined response (using Formula 1) of the neural distinguisher over large number of samples of same distribution as a distinguisher (named as combined-score-distinguisher). By doing so, the signal from the neural distinguisher is amplified and the distinguishability is increased. For a combined-score-distinguisher built on top of a weak neural distinguisher to reach its most potential with respect to distinguishability, the number of samples of the same distribution should be sufficiently large. It is observed that, the required number of samples on which the response from a neural distinguisher are to be combined is closely related with the bias of the accuracy of the neural distinguisher. We have the following Conjecture 1, which is supported by the data in our attacks to be presented in the sequel.

Conjecture 1. For a combined-score distinguisher derived from a neural distinguisher with accuracy lager than 0.5 and of bias ϵ to be successfully used in the improved attack using Upper Confidence Bound and BayesianKeySearch Algorithm, the size of samples from the same distribution should be about $c \times 1/(2\epsilon)^2$, where c is a small constant.

For the hybird differential distinguisher used in the key-recovery attack in [8], it is not straightforward to aggregate enough number of samples of same distribution fed to the neural distinguisher due to the prepended classical differential. To overcome this problem, Gohr in [8] used the neutral bits of the classical differential, which is a notion introduced in collision attacks on hash function [3] and frequently used in previous attacks of different types. That is, changing the values at the neutral bits of an input pair does not change the conformability for the differential. The more the neutral bits of the prepended differential, the larger the number of samples of same distribution could be generated and fed into the neural distinguisher. However, in general, the longer the classical differential, the lesser the number of neutral bits.

Finding enough neutral bits for prepending a long differential over a long but weak neural distinguisher becomes a difficult problem for devising a key-recovery to cover more rounds.

Thus, the first part of this work focuses on finding new types of neutral bits.

3.2 Neutral Bits and Generalized Neutral Bits

Notations. Let $\varDelta_{in} \to \varDelta_{out}$ be a differential with input difference $\varDelta_{in}$ and output difference $\varDelta_{out}$ of an r-round encryption F^r. Let (P, P') be the input pair and $(C, C' \mid C = F^r(P), C' = F^r(P'))$ be the output pair, where $P \oplus P' = \varDelta_{in}$. If $C \oplus C' = \varDelta_{out}$, (P, P') is said to be conforming the differential $\varDelta_{in} \to \varDelta_{out}$.

The primary notion of neutral bits can be interpreted as follows. Let $e_0, e_1, \ldots, e_{n-1}$ be the standard basis of $\mathbb{F}_2^n$. Let i be an index of a bit (starting from 0). The i-th bit is a *neutral bit* for the differential $\varDelta_{in} \to \varDelta_{out}$, if for any conforming pair (P, P'), $(P \oplus e_i, P' \oplus e_i)$ is also a conforming pair.

Let $\{i_1, i_2, \ldots, i_n\}$ be the set of neutral bits of a differential $\varDelta_{in} \to \varDelta_{out}$. Denote the subspace of $\mathbb{F}_2^n$ with basis $\{e_{i_1}, e_{i_2}, \ldots, e_{i_n}\}$ by $\mathcal{S}$. Then, from one input pair (P, P') where $P \oplus P' = \varDelta_{in}$, one can generate a set $\{(P_i, P'_i) \mid P_i \in$

$P \oplus \mathcal{S}, P'_i = P_i \oplus \Delta_{in}\}$ that forms a data structure with the same conformability for the differential.

For a differential $\Delta_{in} \to \Delta_{out}$ of F^r, in the view of system of equations defined on the derivative function of F^r, *i.e.*, $D_{\Delta_{in}} F^r(P) = \Delta_{out}$, a set of neutral bits $\mathcal{NB}$ partitions the solution space of $D_{\Delta_{in}} F^r(x) = \Delta_{out}$ into equivalence classes. It can be seen that, the more neutral bits for a differential, the better structured the solution space.

Generalization of Neutral Bits. In this work, two types of generalized neutral bits are considered beyond the neutral bits considered in [8]. The first type, named as simultaneous-neutral bit-set (SNBS for short), has already been introduced together with the notion of neutral bit in [3], that is, for an input pair, complementing the values of a set of bits simultaneously does not change its conformability for the differential. Formally, it can be defined as follows.

Definition 1 (Simultaneous-neutral bit-sets [3]). *Let $I_s = \{i_1, i_2, \ldots, i_s\}$ be a set of bit indices. Denote $f_{I_s} = \bigoplus_{i \in I_s} e_i$. The bit-set I_s is a simultaneous-neutral bit-set for the differential $\Delta_{in} \to \Delta_{out}$, if for any conforming pair (P, P'), $(P \oplus f_{I_s}, P' \oplus f_{I_s})$ is also a conforming pair.*

The second type, which is a natural generalization, is named in this work, as conditional (simultaneous-) neutral bit(-set)s (CSNBS for short), that is, the bits or bit-sets are neutral for input pairs fulfilling specific conditions. Formally, it can be defined as follows.

Definition 2 (Conditional (simultaneous-) neutral bit(-set)s). *let $I_s = \{i_1, i_2, \ldots, i_s\}$ be a set of bit indices. Denote $f_{I_s} = \bigoplus_{i \in I_s} e_i$. Let $\mathcal{C}$ be a set of constraints on the value of an input P, and $\mathcal{P}_{\mathcal{C}}$ be the set of input that fulfilling constraints $\mathcal{C}$. The bit-set I_s is a conditional simultaneous-neutral bit-set for the differential $\Delta_{in} \to \Delta_{out}$, if for any conforming pair $(P, P' \mid P \in \mathcal{P}_{\mathcal{C}})$, $(P \oplus f_{I_s}, P' \oplus f_{I_s})$ is also a conforming pair.*

The most straightforward constraints can be that some bit values of P are fixed. However, the constraints on the values of input P can be more involved system of linear or non-linear equations, and correspondingly named as *linear-conditional (simultaneous-) neutral bit(-set)s* (LCSNBS for short) and *nonlinear-conditional (simultaneous-) neutral bit(-set)s* (NCSNBS for short).

Specifically, in this work for Simon32/64 and Speck32/64, conditional neutral bits are slightly different in the following ways:

- for Speck32/64, a set of bits is neutral only when the value of some specific bits are fixed to a particular value. Thus, one chooses particular data instead of random one to form ciphertext-structure, but always uses the same set of neutral bit-sets (refer to Sect. 4.1).
- for Simon32/64, depending on the value of specific bits, one can always obtain a neutral bit-sets by grouping different bits. Thus, one randomly generates a data pair, then selects different neutral bit-sets depending on the values of specific bits of the random pair to generate a ciphertext-structure (refer to Sect. A.1).

Besides, an observation on neutral bits of reduced-round differential of Speck32/64 is that, the lower the Hamming weight of a differential, the more the number of neutral bits. Thus, for some of the sequel key-recovery attacks, differentials with sub-optimal probability but with low Hamming weight are of interests and eventually used.

Remark 1. The neutrality of the CSNBS depends on values of some particular bits. The selected data is at intermediate round in our attacks in this work, although the difference does not depend on the round-key, the values do. Thus, using CSNBS, the attack requires to guess some key bits of the first round.

Remark 2. In general, neutral bits of non-trivial differentials are scarce. In [8], because of the scarce of neutral bits for the 2-round prepended differential of Speck32/64, probabilistic neutral bits (PNB for short) are exploited. Formally, it can be defined as follows. Let i be an index of a bit (starting from 0). The i-th bit is a *p-probabilistic neutral bit* for the differential $\Delta_{in} \to \Delta_{out}$, if take a random P and let P' be $P \oplus \Delta_{in}$, (P, P') and $(P \oplus e_i, P' \oplus e_i)$ conform the differential at the same time with probability p. The higher the probability p is, the more useful the neutral bit becomes. For convenience, the neutral bits are said to be *complete neutral bit* when $p = 1$. In other view, the probabilistic neutral bits can be seen as conditional neutral bits with unknown conditionals.

In the sequel attacks on Simon32/64, with involved analysis, conditions on the neutral bits are explicit, and thus, under the known conditions, all used (generalized) neutral bits are complete. For Speck32/64, because of the complicated modular addition, some of the used (generalized) neutral bits, including those conditional ones, are in sense of with high probability. That means, besides those explicit conditions observed, there are still some hidden conditions that can be fulfilled with high probability.

3.3 Exploiting Multiple Differentials Sharing Same Neutral Bits

For the prepended classical differential, the goal is to propagate more rounds with as less plaintext requirement as possible while leaving enough positive samples to the neural distinguisher.

From the connecting difference between the classical differential and the neural distinguisher propagating upward, there might be multiple similar differentials with equally good probability. The observation is that, these similar differentials are likely to share many neutral bits. When a shared neutral bit happens to be exactly the difference between input differences of two differentials, one can re-group ciphertext pairs within each ciphertext structure corresponding to one differential, and obtain ciphertext structures corresponding to the other differential without additional queries, *i.e.*, doubling the number of ciphertext structures for free.

Formally, let $D_1 = \Delta_{in_1} \to \Delta_{out}$ and $D_2 = \Delta_{in_2} \to \Delta_{out}$ be two differentials with input differences satisfying $\Delta_{in_1} \oplus \Delta_{in_2} = \Delta_{nb_i}$ and with the same output difference. Suppose nb_i is a neutral bit for both D_1 and D_2. Then, once a pair of input pair $\{(P, P \oplus \Delta_{in_1}), (P \oplus \Delta_{nb_i}, P \oplus \Delta_{in_1} \oplus \Delta_{nb_i})\}$ is generated for differential D_1, one can re-pair the inputs as $\{(P, P \oplus \Delta_{in_1} \oplus \Delta_{nb_i}), (P \oplus \Delta_{nb_i}, P \oplus \Delta_{in_1})\}$ and obtain a pair of input pair for differential D_2. Thus, by re-pairing the corresponding ciphertext pairs, the number of ciphertext structures are doubled. Such a pair of differentials are said to be matched differentials.

This can reduce the data complexity by half, but is only of interest when the two differentials are with almost equally good probability and share enough other neutral bits to be used in key-recovery attacks.

An example can be found in Sect. 4.1. One useful differential might match with many useful differentials in this sense. The more matched differentials found, the lower the final data complexity will be.

Remark 3. There is an implicit relation between neutral bits of a differential and high-order differential. A simultaneous-neutral bit-set I_s of a differential $\Delta_{in} \to \Delta_{out}$ defines a special high-order differential $\Delta_{a_1,a_2} \to 0$, where $a_1 = \Delta_{in}$ and $a_2 = \bigoplus_{i \in I_s} e_i$.

Besides, there is an interesting relation between neutral bits and the mixture-differential distinguisher of AES. Some neutral bits found for Speck32/64 and Simon32/64 in this work can result in some bit level mixture quadruples.

4 Key Recovery Attack on Round-Reduced Speck32/64

This sections shows that the neural distinguishers have not reached their full potential in the key-recovery attacks in [8]. They could be harnessed to cooperate with classical cryptanalytic tools and perform key-recovery attacks that are more competitive to the attacks devised purely by classical cryptanalysis techniques.

In the following, we present key-recovery attacks employing the same neural distinguishers used in the 11-round and 12-round attacks on Speck32/64 in [7,8]. The first 13-round attack and an improved 12-round attack that use neural distinguishers on Speck32/64 were obtained.

The improved attacks follow the same framework of the improved key-recovery attacks on Speck32/64 in [8]. An r-round main and an $(r-1)$-round helper neural distinguishers are employed and an s-round classical differential is prepended. The key guessing procedure applies a simple reinforcement learning procedure. The last subkey and the second to last subkey are to be recovered without exhaustively using all candidate values to do one-round decryption. Instead, Bayesian key search employing wrong key response profile is to be used.

The prepended classical differentials to be used in the improved attacks includes the same 2-round differential used in the attack in [8] and four new 3-round differentials. The preliminary is to find enough NB of these differentials to obtain enough samples of same distribution, so that to use the combined response from the neural distinguishers. In the following, the simultanous neutral bit-sets and CNB introduced in Sect. 3 are to be found.

4.1 Finding CSNBS for Speck32/64

For finding NB of the differential of round-reduced Speck32/64, we used an exhaustive search for empirical results because of the complexity brought by the carry of modular addition.

Finding SNBS for 2-round Differential. For the prepended 2-round differential on top of the neural distinguishers, one can experimentally obtain 3 complete NB and 2 SNBS (simultaneously complementing up to 4 bits) using exhaustive search. Besides, bits and bit-sets that are (simultaneous-)neutral with high probabilities ($\geq 80\%$) are also detected. Concretely, for the 2-round differential $(\texttt{0x0211}, \texttt{0x0a04}) \to (\texttt{0x0040}, \texttt{0x0000})$, bits and bit-sets that are (probabilistically) (simultaneous-)neutral are summarized in Table 2.

Table 2: (Probabilistic) SNBS for 2-round differential $(\texttt{0x0211}, \texttt{0x0a04}) \to (\texttt{0x0040}, \texttt{0x0000})$ of Speck32/64

NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.
[20]	1	[21]	1	[22]	1	[9, 16]	1	[2, 11, 25]	1		
[14]	0.965	[15]	0.938	[6, 29]	0.91	[23]	0.812	[30]	0.809	[7]	0.806

Find SNBS for 3-round Differential. The 2-round differential $(\texttt{0x0211}, \texttt{0x0a04}) \to (\texttt{0x0040}, \texttt{0x0000})$ can be extended to two optimal (prob. $\approx 2^{-11}$) 3-round differentials, *i.e.*,

$$(\texttt{0x0a20}, \texttt{0x4205}) \to (\texttt{0x0040}, \texttt{0x0000}), (\texttt{0x0a60}, \texttt{0x4205}) \to (\texttt{0x0040}, \texttt{0x0000}).$$

However, the NB/SNBS of these two optimal differential are very scarce. There are four sub-optimal (prob. $\approx 2^{-12}$) 3-round differential, *i.e.*,

$$(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000}), (\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000}),$$
$$(\texttt{0x8021}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000}), (\texttt{0x8061}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000}).$$

For these sub-optimal 3-round differentials, the hamming weights of the input differences are low, and they have more SNBS. Still, the numbers of SNBS are not enough for appending a weak neural network distinguisher. Thus, conditional ones were searched. The concrete approach for finding CNB/CNBS is empirical.

At a high-level, the empirical approach is as follows. First, the sufficient conditions for a bit or a set of bits to be neutral are observed. Next, the necessity of the sufficient conditions is tested. Concretely, let $(\tilde{x}, \tilde{y})$ be the chosen data for the 3-round differential. Because the 3-round differential will be neutrally extended one round to the backward in the key-recovery attack, in the real encryption, $(x, y) = (\tilde{x} \oplus k_0, \tilde{y} \oplus k_0)$ is the real input to the 3-round differential (refer to Fig. 3). The considered sufficient conditions are on the values of each bit of the following four variables, *i.e.*, x, y, $(x \ggg 7) \oplus y$, $(x \ggg 7) \cdot y$. All bits of these variables are examined to see if any of them keeps as a constant 0 or 1 among all correct pairs in the structure generated by each candidate CNBS. Concerning values of x and y is for examining the conditions on the values of the inputs; Concerning the values of the later two is for examining the conditions on the values that will be involved in the modular addition. We observed that for some bits/bits-sets that are neutral with relatively high probabilities, some bits p_i's of $(x \ggg 7) \oplus y$ for the correct pairs are always b ($b \in \{0, 1\}$), from which we obtained the sufficient conditions for the bits/bits-sets to be neutral. We then fixed the corresponding bits p_i's to be b, and examined the probabilities for the bits/bits-sets to be neutral. Exploited experimental results are summarized in Table 3. Besides, we observed that for each of the four sub-optimal differentials, there are three sufficient (linear) conditions for a pair $((x, y), (x', y'))$ to conform the 3-round differential, as listed in Eq. 2.

$$
\begin{array}{c|c|c|c}
\begin{array}{c}(\texttt{0x8020}, \texttt{0x4101}) \\ \downarrow \\ (\texttt{0x0040}, \texttt{0x0000})\end{array} &
\begin{array}{c}(\texttt{0x8060}, \texttt{0x4101}) \\ \downarrow \\ (\texttt{0x0040}, \texttt{0x0000})\end{array} &
\begin{array}{c}(\texttt{0x8021}, \texttt{0x4101}) \\ \downarrow \\ (\texttt{0x0040}, \texttt{0x0000})\end{array} &
\begin{array}{c}(\texttt{0x8061}, \texttt{0x4101}) \\ \downarrow \\ (\texttt{0x0040}, \texttt{0x0000})\end{array} \\
\hline
\begin{cases} x[7] = 0, \\ x[5] \oplus y[14] = 1, \\ x[15] \oplus y[8] = 0. \end{cases} &
\begin{cases} x[7] = 0, \\ x[5] \oplus y[14] = 0, \\ x[15] \oplus y[8] = 0. \end{cases} &
\begin{cases} x[7] = 0, \\ x[5] \oplus y[14] = 1, \\ x[15] \oplus y[8] = 1. \end{cases} &
\begin{cases} x[7] = 0, \\ x[5] \oplus y[14] = 0, \\ x[15] \oplus y[8] = 1. \end{cases}
\end{array}
\tag{2}
$$

Notice that, the first condition $x[7] = 0$ on conforming pairs are shared among the four differentials, while for the other two conditions, they are complementary. Because the conditions are linear, fulfilling each condition, the probability is 2^{-1}. Thus, under the three conditions, the probability of the four differentials are 2^{-9}. However, in the key-recovery attacks, because of the extended one round on top of these 3-round differentials, these conditions cannot be fulfilled by chosen data without guessing corresponding bits of k_0.

Exploiting Multiple Differentials. The four differentials share most of the high-probabilistic NB and the conditions on the NB (except for the [30], [0, 8, 31]). Besides, the neutral bit [22] makes $(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ and $(\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ matched differentials, and $(\texttt{0x8021}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ and $(\texttt{0x8061}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ also matched differentials as introduced in Sect 3.3. More specifically, take $(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ and $(\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ for example, they share neutral bit [22] and all other useful NB. Since $(\texttt{0x8020}, \texttt{0x4101}) \oplus (\texttt{0x8060}, \texttt{0x4101}) = (\texttt{0x0040}, \texttt{0000})$, while the neutral bit [22] corresponds to difference $\mathit{\Delta}_{22} = (\texttt{0x0040}, \texttt{0000})$, ciphertext structures for $(\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ can be directly obtained from that of $(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ (refer to Sect. 3.3). Thus, using a pair of matched differentials (as in the following attack $\mathcal{A}^{\textsc{Speck}_{13R}}$ on the 13-round SPECK32/64), one can generate half of the required data pairs for free. Accordingly, the data complexity to get one pair of ciphertexts is one instead of two.

Table 3: (Probabilistic) (simultaneous-)neutral bit/bit-sets for 3-round differential $(\texttt{0x8020}, \texttt{0x4101}) \rightarrow (\texttt{0x0040}, \texttt{0x0000})$, $(\texttt{0x8060}, \texttt{0x4101}) \rightarrow (\texttt{0x0040}, \texttt{0x0000})$, $(\texttt{0x8021}, \texttt{0x4101}) \rightarrow (\texttt{0x0040}, \texttt{0x0000})$, and $(\texttt{0x8061}, \texttt{0x4101}) \rightarrow (\texttt{0x0040}, \texttt{0x0000})$ of Speck32/64

	(8020, 4101)		(8060, 4101)		(8021, 4101)		(8061, 4101)		
Bit-set	Pre.	Post.	Pre.	Post.	Pre.	Post.	Pre.	Post.	Condition
[22]	0.995	1.000	0.995	1.000	0.996	1.000	0.997	1.000	–
[20]	0.986	1.000	0.997	1.000	0.996	1.000	0.995	1.000	–
[13]	0.986	1.000	0.989	1.000	0.988	1.000	0.992	1.000	–
[12, 19]	0.986	1.000	0.995	1.000	0.993	1.000	0.986	1.000	–
[14, 21]	0.855	0.860	0.874	0.871	0.881	0.873	0.881	0.876	–
[6, 29]	0.901	0.902	0.898	0.893	0.721	0.706	0.721	0.723	–
[30]	0.803	0.818	0.818	0.860	0.442	0.442	0.412	0.407	–
[0, 8, 31]	0.855	0.859	0.858	0.881	0.000	0.000	0.000	0.000	–
[5, 28]	0.495	1.000	0.495	1.000	0.481	1.000	0.469	1.000	$x[12] \oplus y[5] = 1$
[15, 24]	0.482	1.000	0.542	1.000	0.498	1.000	0.496	1.000	$y[1] = 1$
[6, 11, 12, 18]	0.445	0.903	0.456	0.906	0.333	0.701	0.382	0.726	$x[2] \oplus y[11] = 0$
[4, 27, 29]	0.672	0.916	0.648	0.905	0.535	0.736	0.536	0.718	$x[11] \oplus y[4] = 1$

Pre.: probability obtained using 1000 correct pairs without fulfilling the conditions.
Post.: probability obtained using with 1000 correct pairs and fulfilling all the four conditions in the last column.
🟦: Neutral bit(-set)s used in the 13-round attack $\mathcal{A}^{\text{Speck}_{13R}}$ on Speck32/64.
🟩: Neutral bit(-set)s used in the 12-round attack $\mathcal{A}^{\text{Speck}_{12R}}$ on Speck32/64.

For the ease of notation, let us denote $(\texttt{0x8020}, \texttt{0x4101})$ as example difference Δ_E^1, and $(\texttt{0x8021}, \texttt{0x4101})$ as Δ_E^2. Six queries of a plaintext structure consisting of $(P,\ P \oplus \Delta_E^1,\ P \oplus \Delta_{22},\ P \oplus \Delta_E^1 \oplus \Delta_{22},\ P \oplus \Delta_E^2,\ P \oplus \Delta_E^2 \oplus \Delta_{22})$ result in eight pairs to be used in the upcoming attack $\mathcal{A}^{\text{Speck}_{12R}}$ on the 12-round Speck32/64. The eight pairs are two pairs $(P, P \oplus \Delta_E^1)$ and $(P \oplus \Delta_{22}, P \oplus \Delta_E^1 \oplus \Delta_{22})$ following input difference Δ_E^1, two pairs $(P, P \oplus \Delta_E^1 \oplus \Delta_{22})$, $(P \oplus \Delta_{22}, P \oplus \Delta_E^1)$ following input difference $\Delta_E^1 \oplus \Delta_{22}$, two pairs $(P, P \oplus \Delta_E^2)$, $(P \oplus \Delta_{22}, P \oplus \Delta_E^2 \oplus \Delta_{22})$ following input difference Δ_E^2, and two pairs $(P, P \oplus \Delta_E^2 \oplus \Delta_{22})$, $(P \oplus \Delta_{22}, P \oplus \Delta_E^2)$ following input difference $\Delta_E^2 \oplus \Delta_{22}$. In such a way, the average data complexity to get one pair of ciphertexts reduces from 2 to 3/4, equivalent with the saving by a factor of $2^{1.42}$.

Note that, to use these CNBS (in the following $\mathcal{A}^{\text{Speck}_{13R}}$), one has to guess the value corresponding to the conditions, *i.e.*, some key bits or their linear combinations. For example, guessing 4 linear combinations of key bits, one can additionally get 4 more NBS of high probability; The more the guessed bits, the larger each cipher-structure one can expand to, thus the higher the success probability of the key-recovery attack. However, more guessed key bits also result in higher time and data complexities. Thus, one has to determine the trade-off between success rate and attack complexity through the number of guessed key bits of k_0. For example, in the second 13-round attack $\mathcal{A}_{II}^{\text{Speck}_{13R}}$, the bit-set [4, 27, 29] is used as a NBS without guessing the corresponding key bit to explicitly fulfill the conditions in each guess.

4.2 Key Recovery Attack on 13-round SPECK32/64

Employing two classical differentials that have identical CNB that have been identified using the above method, and combining them with neural distinguishers, we examine how far a practical attack can go on reduced-round SPECK32/64. A 13-round attack, denoted by $\mathcal{A}^{\text{SPECK}_{13R}}$, is devised as follows.

The preliminary components that capture characteristics of SPECK32/64 for devising the attack $\mathcal{A}^{\text{SPECK}_{13R}}$ are as follows.

1. Two 3-round classical differentials sharing the same output difference $(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ and $(\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ (refer to the rounds colored in blue in Fig. 3), and the set of their 12 NBS, *i.e.*, $\mathcal{NB}$: $\{[22], [13], [20], [5, 28], [15, 24], [12, 19], [6, 29], [6, 12, 11, 18], [4, 27, 29], [14, 21], [0, 8, 31], [30]\}$ (refer to the columns framed by blue lines in Table 3);
2. An 8-round neural distinguisher $\mathcal{ND}^{\text{SPECK}_{8R}}$ trained with difference $(\texttt{0x0040}, \texttt{0x0000})$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{8R}}.\mu$ and $\mathcal{ND}^{\text{SPECK}_{8R}}.\sigma$;
3. A 7-round neural distinguisher $\mathcal{ND}^{\text{SPECK}_{7R}}$ trained with difference $(\texttt{0x0040}, \texttt{0x0000})$ and its wrong key response profiles $\mathcal{ND}^{\text{SPECK}_{7R}}.\mu$ and $\mathcal{ND}^{\text{SPECK}_{7R}}.\sigma$.

The parameters for recovering the last two subkeys k_{12} and k_{11} are denoted as follows.

1. n_{kg}: the number of possible values for the bits of k_0, on which the conditions depend.
2. n_{cts}: the number of ciphertext structures by n_{cts}.
3. n_b: the number of ciphertext pairs in each ciphertext structure, *i.e.*, $2^{|\mathcal{NB}|}$.
4. n_{it}: the total number of iterations on the ciphertext structures.
5. c_1 and c_2: the cutoffs with respect to the scores of the recommended last subkey and second last subkey, respectively.
6. n_{byit1}, n_{cand1} and n_{byit2}, n_{cand2}: the number of iterations and number of key candidates within each iteration in the BAYESIANKEYSEARCH procedures for guessing each of the last and the second last subkeys, respectively.

The attack procedure is as follows (refer to Fig. 2 and 3).

1. Initialize variables $Gbest_{\text{key}} \leftarrow (\text{None}, \text{None})$, $Gbest_{\text{score}} \leftarrow -\infty$.
2. For each of the n_{kg} values of the 6 key bits $k_0[7]$, $k_0[15] \oplus k_0[8]$, $k_0[12] \oplus k_0[5]$, $k_0[1]$, $k_0[2] \oplus k_0[11]$, $k_0[11] \oplus k_0[4]$,
   (a) Generate $n_{cts}/2$ random data pairs , *i.e.*, $(\tilde{x}_1||\tilde{y}_1,\ \tilde{x}'_1||\tilde{y}'_1)$'s, with difference $(\texttt{0x8020}, \texttt{0x4101})$, and satisfying the conditions for being conforming pairs, *i.e.*, $\begin{cases}\tilde{x}_1[7] = k_0[7],\\ \tilde{x}_1[15] \oplus \tilde{y}_1[8] = k_0[15] \oplus k_0[8],\end{cases}$ (refer to Eq. 2), and the conditions for increasing the neutrality probability of four bits *i.e.*,
   $$\begin{cases}\tilde{x}_1[12] \oplus \tilde{y}_1[5] \oplus 1 = k_0[12] \oplus k_0[5],\\ \tilde{y}_1[1] \oplus 1 = k_0[1],\\ \tilde{x}_1[2] \oplus \tilde{y}_1[11] = k_0[2] \oplus k_0[11],\\ \tilde{x}_1[11] \oplus \tilde{y}_1[4] \oplus 1 = k_0[11] \oplus k_0[4],\end{cases}$$
   (refer to Table 3).

(b) From the $n_{cts}/2$ random data pairs, generate $n_{cts}/2$ structures using the NBS in $\mathcal{NB}$, marking the correspondence between old pairs and new pairs that are generated using the NB [22].

(c) Decrypt one round using zero as the subkey for all data in the structures obtained above (*i.e.*, the round in green depicted in Fig. 3) and obtain $n_{cts}/2$ plaintext structures;

(d) Query for the ciphertexts under 13-round Speck32/64 of the $n_{cts}/2 \times n_b \times 2$ plaintexts, thus obtain $n_{cts}/2$ ciphertext structures.

(e) For each couple of ciphertext pairs, denoted by (c_1, c_1') and (c_2, c_2'), whose corresponding couple of data pairs are related by flipping the neutral bit [22], that is the couple $(\tilde{x}_1||\tilde{y}_1, \tilde{x}_1||\tilde{y}_1 \oplus (\texttt{0x8020}, \texttt{0x4101}))$ and $(\tilde{x}_1||\tilde{y}_1 \oplus (\texttt{0x0040}, \texttt{0000}), \tilde{x}_1||\tilde{y}_1 \oplus (\texttt{0x8020}, \texttt{0x4101}) \oplus (\texttt{0x0040}, \texttt{0000}))$, obtain a new couple of ciphertext pairs, that is (c_1, c_2') and (c_2, c_1'). As a result, the new couples generated in this way are corresponding to couples of plaintext pairs for the second differential $(\texttt{0x8060}, \texttt{0x4101})$ and its neutral bit [22]. Thus, additional $n_{cts}/2$ ciphertext structures can be obtained without new queries. In total, n_{cts} ciphertext structures, denoted by $\{\mathcal{C}_1, \ldots, \mathcal{C}_{n_{cts}}\}$, are obtained.

(f) Initialize an array $w_{\max}$ and an array n_{visit} to record the highest distinguisher score obtained so far and the number of visits have received in the last subkey search for the ciphertext structures.

(g) Initialize variables $best_{\text{score}} \leftarrow -\infty$, $best_{\text{key}} \leftarrow (\text{None}, \text{None})$, $best_{\text{pos}} \leftarrow$ None to record the best score, the corresponding best recommended values for the two subkeys obtained among all ciphertext structures and the index of this ciphertext structure.

(h) For j from 1 to n_{it}:

   i. Compute the priority of each of the ciphertext structures as follows: $s_i = w_{\max i} + \alpha \cdot \sqrt{\log_2(j)/n_{\text{visit}i}}$, for $i \in \{1, \ldots, n_{cts}\}$, and $\alpha = \sqrt{n_{cts}}$;

   ii. pick the ciphertext structure with the highest priority score for further processing in this j-th iteration, denote it by $\mathcal{C}$, and its index by idx, $n_{\text{visit}idx} \leftarrow n_{\text{visit}idx} + 1$.

   iii. Run BayesianKeySearch Algorithm 3 with $\mathcal{C}$, the neural distinguisher $\mathcal{ND}^{\textsc{Speck}_{8R}}$ and its wrong key response profile $\mathcal{ND}^{\textsc{Speck}_{8R}}.\mu$ and $\mathcal{ND}^{\textsc{Speck}_{8R}}.\sigma$, n_{cand1}, and n_{byit1} as input parameters; obtain the output, that is a list L_1 of $n_{byit1} \times n_{cand1}$ candidate values for the last subkey and their scores, *i.e.*, $L_1 = \{(g_{1i}, v_{1i}) : i \in \{1, \ldots, n_{byit1} \times n_{cand1}\}\}$.

   iv. Find the maximum $v_{1\max}$ among v_{1i} in L_1, if $v_{1\max} > w_{\max idx}$, $w_{\max idx} \leftarrow v_{1\max}$.

   v. For each of the recommended last subkey $g_{1i} \in L_1$, if the score $v_{1i} > c_1$,

      A. Decrypt the ciphertexts in $\mathcal{C}$ using the g_{1i} by one round and obtain the ciphertext structure $\mathcal{C}'$ of 12-round Speck32/64.

      B. Run BayesianKeySearch Algorithm 3 with $\mathcal{C}'$, the neural distinguisher $\mathcal{ND}^{\textsc{Speck}_{7R}}$ and its wrong key response profile $\mathcal{ND}^{\textsc{Speck}_{7R}}.\mu$ and $\mathcal{ND}^{\textsc{Speck}_{7R}}.\sigma$, n_{cand2}, and n_{byit2} as input parameters; obtain the output, that is a list L_2 of $n_{byit2} \times n_{cand2}$ candidate values for the second to last subkey and their scores, *i.e.*, $L_2 = \{(g_{2i}, v_{2i}) : i \in \{1, \ldots, n_{byit2} \times n_{cand2}\}\}$.

C. Find the maximum among v_{2i} and the corresponding g_{2i} in L_2, and denote them by $v_{2\max}$ and $g_{2\max}$.

D. If $v_{2\max} > best_{\text{score}}$, update $best_{\text{score}} \leftarrow v_{2\max}$, $best_{\text{key}} \leftarrow (g_{1i}, g_{2\max})$, $best_{\text{pos}} \leftarrow idx$.

vi. If $best_{\text{score}} > c_2$, go to Step 2i.

(i) Make a final improvement using VerifierSearch [7] on the value of $best_{\text{key}}$ by examining whether the scores of a set of keys obtained by changing at most 2 bits on top of the incrementally updated $best_{\text{key}}$ could be improved recursively until no improvement obtained, update $best_{\text{score}}$ to the best score in the final improvement; If $best_{\text{score}} > Gbest_{\text{score}}$, $Gbest_{\text{score}} \leftarrow best_{\text{score}}$, $Gbest_{\text{key}} \leftarrow best_{\text{key}}$.

3. Return $Gbest_{\text{key}}$, $Gbest_{\text{score}}$.

In the experimental verification of the attack $\mathcal{A}^{\text{Speck}_{13R}}$, the 8-round and 7-round neural distinguishers provided in [7,9] were used. Concrete parameters and the complexity of $\mathcal{A}^{\text{Speck}_{13R}}$ are as follows. The accuracy of $\mathcal{ND}^{\text{Speck}_{8R}}$ is about 0.514, and that of $\mathcal{ND}^{\text{Speck}_{7R}}$ is about 0.616 (note that $(2 \cdot (0.514 - 0.5))^2 \approx 2^{-10.32}$. Thus, by Conjecture 1, $n_b = 2^{12}$ should be enough).

$$\begin{array}{llll} n_{kg} = 2^6, & n_{cts} = 2^{11}, & n_b = 2^{12}, & n_{it} = 2^{12} \\ c_1 = 25, & c_2 = -690, & n_{byit1} = n_{byit2} = 5, & n_{cand1} = n_{cand2} = 32 \end{array}$$

The data complexity is $n_{kg} \times n_{cts} \times n_b$, that is, $2^{6+11+12}$, *i.e.*, 2^{29} plaintexts (because of the using of two matched differentials, data complexity for getting each ciphertext pair is 1 instead of 2.)

To make the experimental verification economic, we only tested the core of the attack under the situation when the 6 conditions are fulfilled. That is, what tested is that whether one particular loop of the 2^6 loops in Step 2 can successfully recover the last two subkeys. In that particular loop, the trialed value of the 6 key bits of k_0 is the real value. The other loops can be expected to obtain worse scores and wrong key guesses than that particular loop. That is because, when any of the first two conditions is not fulfilled, no correct pairs for the prepended 3-round differential can be generated; When any of the other four conditions is not fulfilled, the quality of the ciphertext structures corresponds to the correct pairs becomes worse. That is, many samples expanded from the correct pair using the neutral bits within a ciphertext structure are not correct pairs. Thus, the combined score should be low. Even if they are indeed of good quality by chance, the influence on the success rate is a positive one. That is because in that case, the recommend key guess can be expected to be good, so the success rate can be better than the results obtained under this assumption. Besides, we assumed that when the there is no correct data pairs conforming the prepended 3-round differentials, the recommended values for the last two subkeys are not correct and the scores are low. Thus, to save time during the experiments, we peeped the generated data pairs, when there is no correct pairs, we terminated that test and assumed it is a failed one and the time of this test is that of running the full n_{it} iterations. In these considerations, the success rate reported below is an underestimation of the real success rate.

The core of the attack was examined in 149 trials. Among the 149 trials, 76 trials were considered to be doomed to failure because of no correct data pairs for the prepended differential, and were terminated directly. We count a key guess as successful if the sum of the hamming weights of the differences between the returned last two subkeys and the real two subkeys is at most two. Within the remaining 73 trials in which the neural distinguishers are called, there are 31 succeeded trials. Thus, we count the success rate as 31/149, which is 0.208 and approximately 0.21.

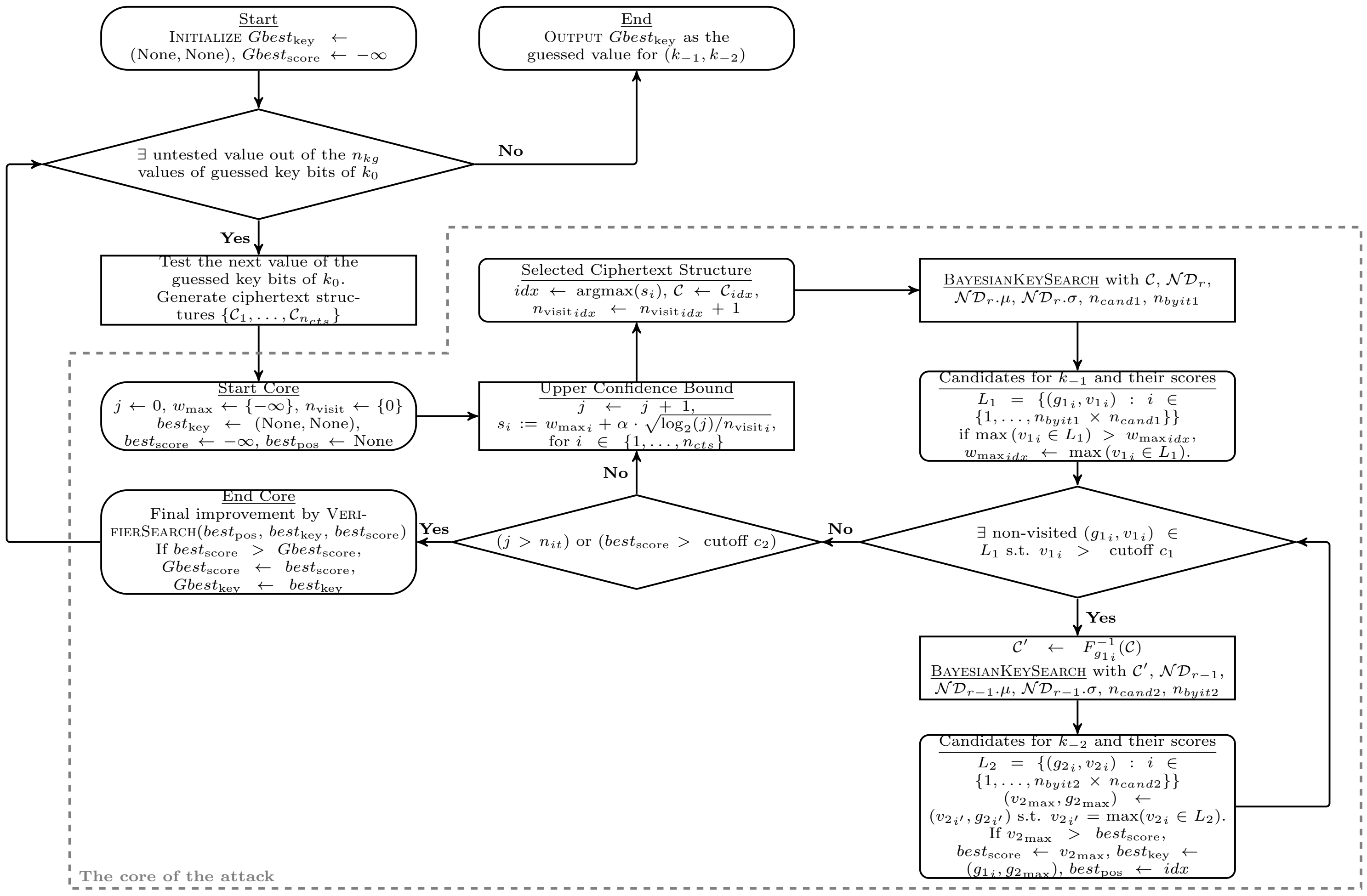

Start
Initialize $Gbest_{\text{key}} \leftarrow$ (None, None), $Gbest_{\text{score}} \leftarrow -\infty$
End
Output $Gbest_{\text{key}}$ as the guessed value for (k_{-1}, k_{-2})
$\exists$ untested value out of the n_{kg} values of guessed key bits of k_0
No
Yes
Test the next value of the guessed key bits of k_0. Generate ciphertext structures $\{\mathcal{C}_1, \ldots, \mathcal{C}_{n_{cts}}\}$
Selected Ciphertext Structure
$idx \leftarrow \text{argmax}(s_i)$, $\mathcal{C} \leftarrow \mathcal{C}_{idx}$, $n_{\text{visit}\,idx} \leftarrow n_{\text{visit}\,idx} + 1$
BayesianKeySearch with $\mathcal{C}$, $\mathcal{ND}_r$, $\mathcal{ND}_r.\mu$, $\mathcal{ND}_r.\sigma$, n_{cand1}, n_{byit1}
Start Core
$j \leftarrow 0$, $w_{\max} \leftarrow \{-\infty\}$, $n_{\text{visit}} \leftarrow \{0\}$ $best_{\text{key}} \leftarrow$ (None, None), $best_{\text{score}} \leftarrow -\infty$, $best_{\text{pos}} \leftarrow$ None
Upper Confidence Bound
$j \leftarrow j + 1$,
$s_i := w_{\max i} + \alpha \cdot \sqrt{\log_2(j)/n_{\text{visit}\,i}}$, for $i \in \{1, \ldots, n_{cts}\}$
Candidates for k_{-1} and their scores
$L_1 = \{(g_{1i}, v_{1i}) : i \in \{1, \ldots, n_{byit1} \times n_{cand1}\}\}$ if $\max(v_{1i} \in L_1) > w_{\max idx}$, $w_{\max idx} \leftarrow \max(v_{1i} \in L_1)$.
No
End Core
Final improvement by VerifierSearch($best_{\text{pos}}$, $best_{\text{key}}$, $best_{\text{score}}$) If $best_{\text{score}} > Gbest_{\text{score}}$, $Gbest_{\text{score}} \leftarrow best_{\text{score}}$, $Gbest_{\text{key}} \leftarrow best_{\text{key}}$
Yes
$(j > n_{it})$ or $(best_{\text{score}} >$ cutoff $c_2)$
No
$\exists$ non-visited $(g_{1i}, v_{1i}) \in L_1$ s.t. $v_{1i} >$ cutoff c_1
Yes
$\mathcal{C}' \leftarrow F^{-1}_{g_{1i}}(\mathcal{C})$
BayesianKeySearch with $\mathcal{C}'$, $\mathcal{ND}_{r-1}$, $\mathcal{ND}_{r-1}.\mu$, $\mathcal{ND}_{r-1}.\sigma$, n_{cand2}, n_{byit2}
Candidates for k_{-2} and their scores
$L_2 = \{(g_{2i}, v_{2i}) : i \in \{1, \ldots, n_{byit2} \times n_{cand2}\}\}$
$(v_{2\max}, g_{2\max}) \leftarrow (v_{2i'}, g_{2i'})$ s.t. $v_{2i'} = \max(v_{2i} \in L_2)$.
If $v_{2\max} > best_{\text{score}}$, $best_{\text{score}} \leftarrow v_{2\max}$, $best_{\text{key}} \leftarrow (g_{1i}, g_{2\max})$, $best_{\text{pos}} \leftarrow idx$
The core of the attack

Fig. 2: Framework of the key-recovery attacks

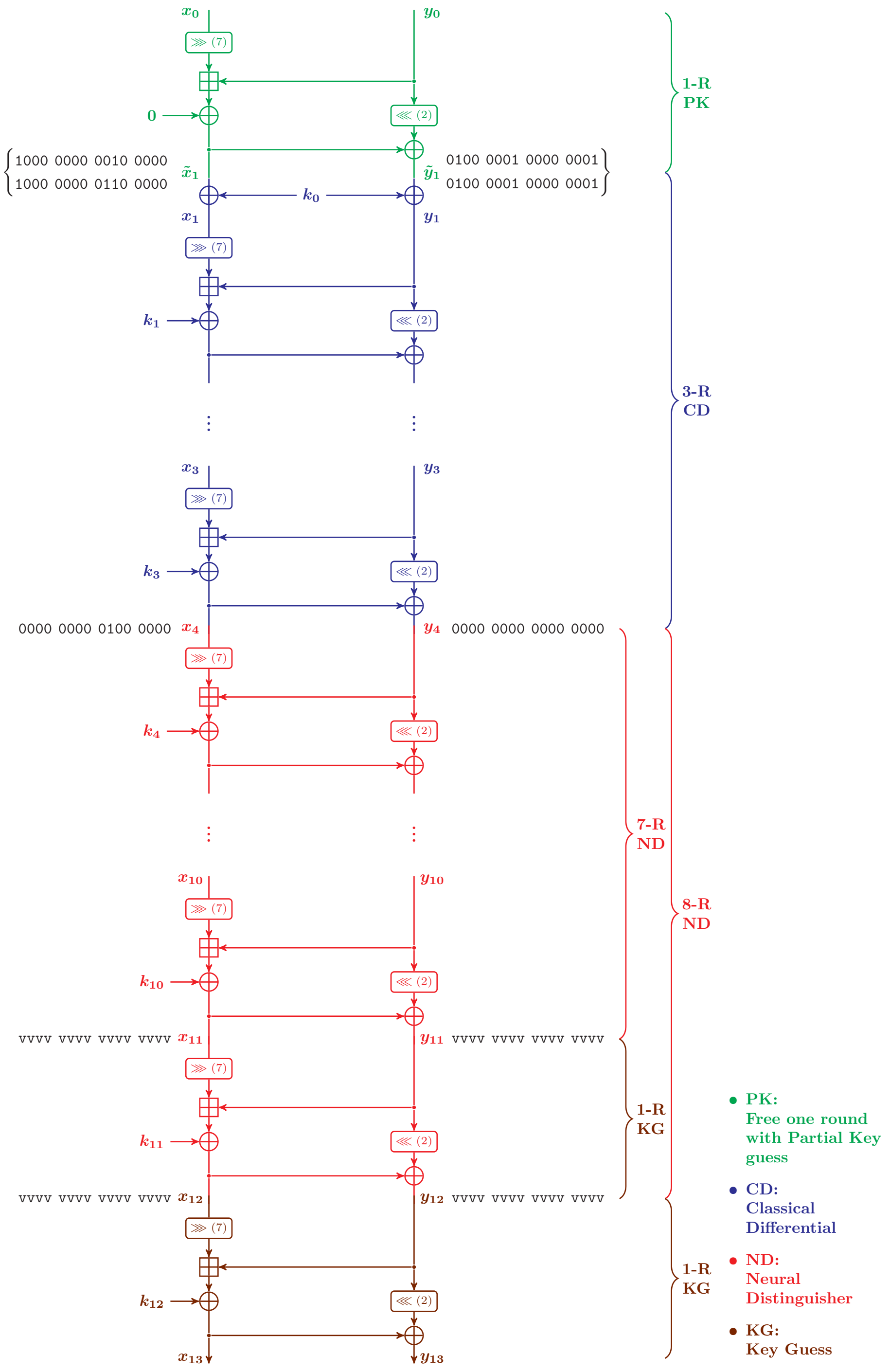


Fig. 3: Components for key-recovery attack on 13-round Speck32/64

The 149 trials were executed using 8 threads of a server with 8 GPUs[3], each of 7 thread ran 20 trials, and 1 thread ran 9 trials[4]. Running all $n_{it} = 4096$ iterations required sightly less than 1 hour (about 55 minutes). Thus, for the 76 trials doomed to be failure, the total executing time is expected to be 76 hours. The full test of 149 trials when directly terminating the 76 trials required 51 hours and 8 minutes (the total core hours for all the 8 threads to terminate). Thus, the full execution of 149 trials is expected to take 76 + 51 hours, that is 130 hours; and thus the average time for each trial is 127 / 149, that is, 0.85 hours. For 2^6 loops in Step 2, the worst situation is that within each loop, all n_{it}, *i.e.*, 4096 iterations are executed. All in all, the full attack requires no more than 2^6, *i.e.*, 64 GPU hours.

Observing that without fulfilling the condition, the bit-set $[4, 27, 29]$ also has a biased probability to be neutral. Thus, we trialed an attack (denoted by $\mathcal{A}_{II}^{\textsc{Speck}_{13R}}$) in which, only 5 bits of k_0 were guessed (*i.e.*, enumerated in Step 2), in which, two bits are for the conforming pair, and 3 bits for the NB except which, all other parameters are the same as the above. Thus, both the time and data complexity can be reduced by 2, that is, 32 GPU hours and 2^{28} chosen plaintexts. However, the success rate also dropped; out of 160 trials, only 19 successful key guesses. That is, the success rate is approximately 0.12.

4.3 Key Recovery Attack on 12-round Speck32/64

To devise key-recovery attack on 12-round Speck32/64, Gohr in [7,9] used the 2-round classical differential $(\texttt{0x0211}, \texttt{0x0a04}) \to (\texttt{0x0040}, \texttt{0x0000})$ combined with the 8-round and 7-round neural distinguishers. For amplifying the weak signal from the 8-round neural distinguisher, 13 single-bit NBs of the prepended 2-round classical differential were exploited. However, many of the 13 NBs are neutral with probabilities that are not high (refer to App. Table 6). Besides, 500 ciphertext structures and 2000 iterations were used to achieve a success rate of 0.40. Thus, the data complexity is $500 \times 2^{13} \times 2$, *i.e.*, $2^{22.97}$ plaintexts. The attack takes roughly 12 hours on a quad-core PC [7] (as listed in Table 1).

From Table 2, it can be seen that there are SNBSs that is completely neutral or is neutral with high probability. Using these SNBSs in Table 2 instead of those used in [7], our experiments show that the success rate of the resulted attack is approximately 1. Note that within 500 ciphertext structures, there are enough correct pairs, which is unlike that in the 11-round attack that uses the same 2-round prepended classical differential but only 100 ciphertext structures in [8]. Thus, when the quality of the ciphertext structures improved by using complete or high-probabilistic neutral bit-sets, the success rate improved considerably.

However, the data complexity is bounded by the weakness of the 8-round distinguisher. Thus, we further considered combining the 3-round classical differential and the stronger 7-round (and 6-round) neural distinguisher to see how

[3] Tesla V100-SXM2-32GB, computeCapability: 7.0; coreClock: 1.53GHz; coreCount: 80; deviceMemorySize: 31.72GB; deviceMemoryBandwidth: 836.37GB/s)

[4] This thread was also designed to run 20 trials; but by accident, it terminated at its ninth trial.

much the data complexity can be reduced. In this case, SNBS are enough for the 7-round neural distinguisher. Thus, those conditional ones can be dismissed in such a 12-round attack. Therefore, all the four 3-round differentials that sharing the many NB can be employed, which makes it possible to obtain one plaintext pair with 3/4 instead of 2 queries (*i.e.*, by obtaining 8 ciphertext pairs with 6 queries as introduced in Sect. 4.1).

Concretely, the components of the 12-round key-recovery attack on Speck32/64, denoted by $\mathcal{A}^{\textsc{Speck}_{12R}}$, are as followed.

1. four 3-round classical differentials $(\texttt{0x8020}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$, $(\texttt{0x8060}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$, $(\texttt{0x8021}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$, $(\texttt{0x8061}, \texttt{0x4101}) \to (\texttt{0x0040}, \texttt{0x0000})$ and the set of their 6 neutral bit(-set)s, *i.e.*, $\mathcal{NB}$: $\{[22], [13], [20], [12, 19], [14, 21], [6, 29]\}$ (refer to the rows framed by green lines in Table 3);
2. a 7-round neural distinguisher $\mathcal{ND}^{\textsc{Speck}_{7R}}$ trained with difference $(\texttt{0x0040}, \texttt{0x0000})$ and its wrong key response profiles $\mathcal{ND}^{\textsc{Speck}_{7R}}.\mu$ and $\mathcal{ND}^{\textsc{Speck}_{7R}}.\sigma$;
3. a 6-round neural distinguisher $\mathcal{ND}^{\textsc{Speck}_{6R}}$ trained with difference $(\texttt{0x0040}, \texttt{0x0000})$ and its wrong key response profiles $\mathcal{ND}^{\textsc{Speck}_{6R}}.\mu$ and $\mathcal{ND}^{\textsc{Speck}_{6R}}.\sigma$.

The framework of the 12-round attack $\mathcal{A}^{\textsc{Speck}_{12R}}$ follows that of $\mathcal{A}^{\textsc{Speck}_{13R}}$ (refer to Fig. 2). The difference is that, at the beginning, we only guess one key bit of k_0, that is $k_0[7]$, because for all four 3-round differentials, the common condition for correct pairs is $x_1[7] = 0$ (refer to Eq. 2). Thus, n_{kg} is 2, and there are only 2 outermost loops.

The concrete parameters of $\mathcal{A}^{\textsc{Speck}_{12R}}$ are as follows. The accuracy of $\mathcal{ND}^{\textsc{Speck}_{7R}}$ is about 0.616, and that of $\mathcal{ND}^{\textsc{Speck}_{6R}}$ is about 0.788 (note that $(2 \cdot (0.616 - 0.5))^2 \approx 2^{-4.22}$. Thus, by Conjecture 1, $n_b = 2^6$ should be enough).

$$\begin{array}{llll} n_{kg} = 2^1, & n_{cts} = 2^{12}, & n_b = 2^6, & n_{it} = 2^{13} \\ c_1 = 10, & c_2 = 20, & n_{byit1} = n_{byit2} = 5, & n_{cand1} = n_{cand2} = 32 \end{array}$$

The data complexity is $n_{kg} \times n_{cts} \times n_b \times 3/4$, that is, $2^{18.58}$ plaintexts. To compare with previous attacks, the experiment were done using CPU. Concretely, 320 trials were done with 16 threads in a CPU server[5]. Each thread ran 20 trials. Within the 320 trials, 173 trials have no correct ciphertext pairs before being expanded by neutral bit-sets thus were terminated directly and taken as failed trials. In the remaining 147 trials, there are 103 success trials (the returned last two subkeys have hamming distance to the real subkeys at most two). Thus, the success rate is computed as 103/320, *i.e.*, 0.32.

The 320 trials, among which 173 trials terminated directly and 147 trials calls the neural distinguishers, took 162 CPU core hours in total. For a trial, running full 8192 iterations requires about 1 hour and 50 minutes. Thus, the worse case to run two outermost loops for a full attack takes less than 4 CPU hours.

[5] Equipped with a 16-core Intel(R) Xeon(R) CPU E5-2680 0 2.70GHz, and 128GB RAM, on CentOS 7.6.

5 Neural Distinguishers on Round-Reduced Simon32/64

This section presents the neural distinguishers on Simon32/64 obtained in this work, using which, key-recovery attacks covering 16 rounds are devised and presented in the next section. The advantage in terms of data complexity further convince that machine learning can produce powerful cryptographic distinguishers that can be used to devise efficient key-recovery attacks competitive to the published results obtained using orthodox cryptanalysis methods.

5.1 The Choice of the Network Architecture

Considering that several state-of-the-art neural network structures have been developed in the field of machine learning, a preliminary search for a better network other than the Residual Network (ResNet) [10] used in [8] were conducted.

Considering that both the Dense Network (DenseNet) [12] and the Squeeze-and-Excitation Network (SENet) [11] show advantages in specific tasks than ResNet, these two networks together with ResNet were investigated. The results on the performance of distinguishers covers 7 to 9 rounds Simon32/64 under the three different network structures are presented in Table 4. From comparison, it can be seen that SENet yields distinguishers that are superior to that of the other two. In the following, we only report essential details of the distinguishers trained using the SENet.

5.2 The Training of Neural Distinguishers

The training schemes were followed from that in [8]. It was reported that there was a gap between attacks that only use the information contained in observed *differences of ciphertext pairs* and the full information contained in output *values of ciphertext pairs*. That indicates that neural distinguishers successfully use features of the ciphertext pairs invisible to all differential distinguishers. Our training on reduced-round Simon32/64 with pure differences of ciphertext pairs confirm this observation, they are inferior to those trained with values of ciphertext pairs.

However, we found that distinguishers fed with partial values combined with partial differences between ciphertext pairs, instead of full values of ciphertext pairs, could still perform good and be more useful than their counterparts to do key-recovery attacks, due to the specific round structure of Simon compared with Speck.

Training using the basic scheme. Using the basic training scheme and adopting SENet, neural distinguishers to recognize output pairs of 7-, 8-, 9-round Simon32/64 with the input difference $(\texttt{0x0000}, \texttt{0x0040})$ are obtained. That is, given an output pair (x, y) and (x', y') and represented in the form of (x, y, x', y'), they can predict whether the data corresponds to input pairs with difference $(\texttt{0x0000}, \texttt{0x0040})$ of the 7-, 8-, 9-round Simon32/64. To make a distinction from their counterparts accepting transformed data, *i.e.*, $(x, x', y \oplus y')$, the 7-, 8-, 9-round neural distinguishers presented here are named as $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{7R}}$, $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{8R}}$, and $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$, respectively.

The 7-round $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{7R}}$ achieves an accuracy as high as 0.9825, which drops by 0.17 per round to 0.8151 and 0.6325 for $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{8R}}$ and $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$, respectively. Summaries are presented in Table 4 for detailed accuracy and in Fig. 4 for the wrong key response profile.

Thus, using this basic scheme, if accuracy drops constantly with the increasing of number of round, the best can be achieved is expected to be a 9-round one, which is indeed the case confirmed by our attempt to train 10 rounds resulting in an accuracy of 0.5011 only.

Training to simulate KeyAverageing algorithm. A successful training of the 10-round distinguisher is achieved by adopting the training scheme of simulating a KeyAverageing Algorithm [8] used with the 9-round $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$. Concretely, a size 2^{20} sample set $\mathcal{S}$ of ciphertext pairs for 10-round Simon32/64 is generated, one half corresponds to plaintext pairs with difference $(\texttt{0x0000}, \texttt{0x0040})$ and the other half corresponds to random plaintext pairs. The labels of these samples are not assigned directly, but using the KeyAverageing Algorithm calling the 9-round $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$. That is, each ciphertext pair c_i in the set $\mathcal{S}$ is decrypted by one-round using all possible values of the 10-th round subkey; thus 2^{16} intermediate values $c'_{i,j}$'s for $j \in \{0,1\}^{16}$ are generated; grading the $c'_{i,j}$'s using the 9-round $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$, and combining the 2^{16} scores into a score for the ciphertext pair c_i by transforming the scores into real-vs-random likelihood ratios and averaging. This combined score is then taken as the label of c_i in $\mathcal{S}$.

Using the sample set $\mathcal{S}$ with the labels so obtained, a training, which follows the training of the best 7-round neural distinguisher in [8], is performed from a randomly initialized network state for 300 epochs at batch size 5000 with a single learning rate drops from 0.001 to 0.0001 at epoch 200. This training procedure results in a 10-round distinguisher, named as $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{10R}}$ with accuracy 0.5551, as summarized in Table 4 for detailed accuracy and Fig. 4 for the wrong key response profile.

Training using the Staged Training Method. The best 11-round distinguisher that can be successfully used in a practical key-recovery attack, is trained using the staged training method, which was the same method used to train the 8-round distinguisher of Speck32/64 in [8]. Concretely, in the first stage, the best 9-round distinguisher $\mathcal{ND}_{\mathbf{VV}}^{\textsc{Simon}_{9R}}$ is retained to recognize 8-round Simon32/64 with the input difference $(\texttt{0x0440}, \texttt{0x0100})$. Note that, the most likely difference to appear three rounds after the input difference $(\texttt{0x0000}, \texttt{0x0040})$ is $(\texttt{0x0440}, \texttt{0x0100})$, and the probability is about 2^{-4}. In this first stage, the number of examples for training and for testing are 2^{28} and 2^{26}, respectively. The number of epochs is 10 and the learning rate is 10^{-4}. In the second stage, the resulted network of the first stage is retained to recognize 11-round Simon32/64

Table 4: Summary of neural distinguishers on Simon32/64

#R	Name	Network	Accuracy	True Positive Rate	True Negative Rate
7		ResNet	0.9197	0.8929	0.9504
		DenseNet	0.9240	0.8888	0.9661
	$\mathcal{ND}^{\textsc{Simon}7R}_{\mathbf{VV}}$	SENet	0.9802	0.9631	0.9986
8		ResNet	0.7248	0.7488	0.7051
		DenseNet	0.7421	0.7643	0.7233
	$\mathcal{ND}^{\textsc{Simon}8R}_{\mathbf{VV}}$	SENet	0.8148	0.7987	0.7987
	$\mathcal{ND}^{\textsc{Simon}8R}_{\mathbf{VD}}$	SENet	0.6587	0.6979	0.6316
9		ResNet	0.6259	0.6681	0.6007
		DenseNet	0.6442	0.6850	0.6182
	$\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VV}}$	SENet	0.6532	0.6982	0.6226
	$\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VD}}$	SENet	0.5629	0.5739	0.5548
10	$\mathcal{ND}^{\textsc{Simon}10R}_{\mathbf{VV}}$ *	SENet	0.5551	0.5679	0.5463
	$\mathcal{ND}^{\textsc{Simon}10R}_{\mathbf{VV}}$ +	SENet	0.5608	0.5732	0.5520
11	$\mathcal{ND}^{\textsc{Simon}11R}_{\mathbf{VV}}$	SENet	0.5173	0.5178	0.5168

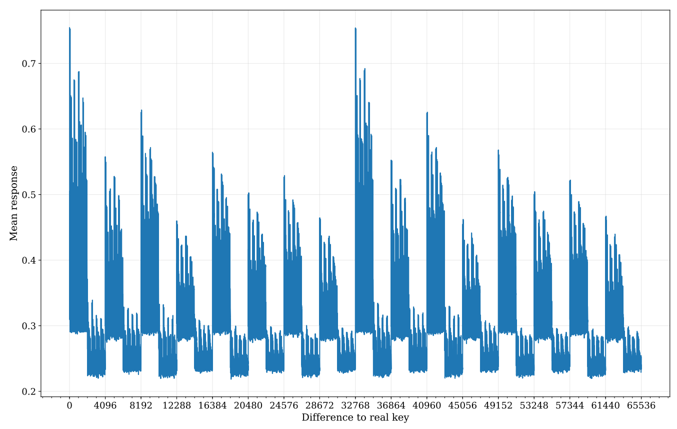
* This neural distinguisher is trained using the KeyAveraging algorithm.
+ This neural distinguisher is trained using the staged training method.

with the input difference (`0x0000`, `0x0040`). For this training, 2^{30} examples are freshly generated and fed, and 2^{28} examples are for verification. One epoch with learning rate 10^{-4} is done. In the last stage, the resulted network of the second stage is retained in two epochs with 2^{30} freshly generated data for training and 2^{28} data for verification. The learning rate is 10^{-5}.

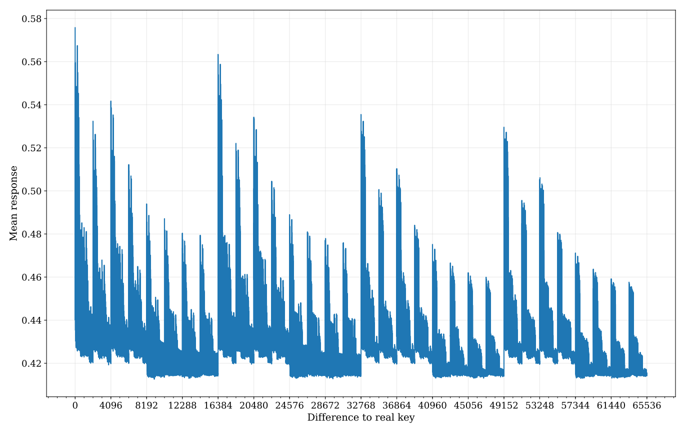
The resulted distinguisher $\mathcal{ND}^{\textsc{Simon}11R}_{\mathbf{VV}}$ achieves an accuracy 0.5173 (refer to Table 4 for detailed accuracy and Fig. 4 for the wrong key response profile.)

Training directly using data of form $(x, x', y \oplus y')$. Notice that, once the output of the r-th round (x_r, x'_r, y_r, y'_r) is known, one can directly compute $(x_{r-1}, x'_{r-1}, y_{r-1} \oplus y'_{r-1})$ without knowing the $(r-1)$-th subkey. Thus, an $(r-1)$-round distinguisher accepting data of the form $(x, x', y \oplus y')$ can be used as an r-round distinguisher in the key-recovery attack. With this consideration, $(r-1)$-round distinguishers accepting data of the form $(x, x', y \oplus y')$ are trained to see whether they are superior to r-round distinguishers accepting data of the form (x, x', y, y'). To make a distinction, let us denote the former by $\mathcal{ND}^{\textsc{Simon}(r-1)R}_{\mathbf{VD}}$ and the latter by $\mathcal{ND}^{\textsc{Simon}rR}_{\mathbf{VV}}$.

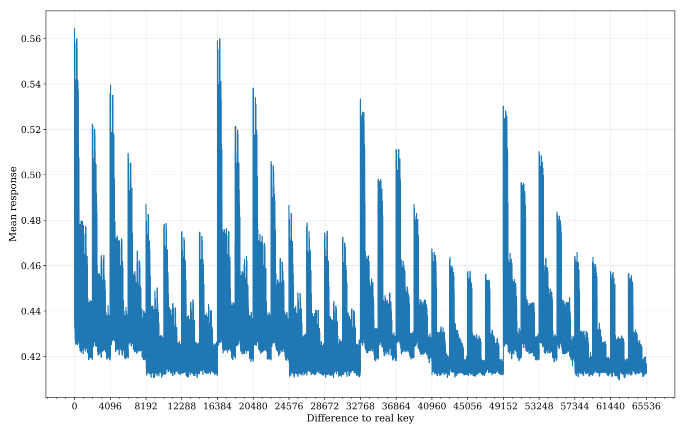
The results show that $\mathcal{ND}^{\textsc{Simon}(r-1)R}_{\mathbf{VD}}$ indeed could achieve slightly better accuracy than $\mathcal{ND}^{\textsc{Simon}rR}_{\mathbf{VV}}$ (refer to Table 4 for detailed accuracy and Fig. 4 for their wrong key response profiles for more comparisons). Thus, they were used in the key-recovery attacks presented in the next section.



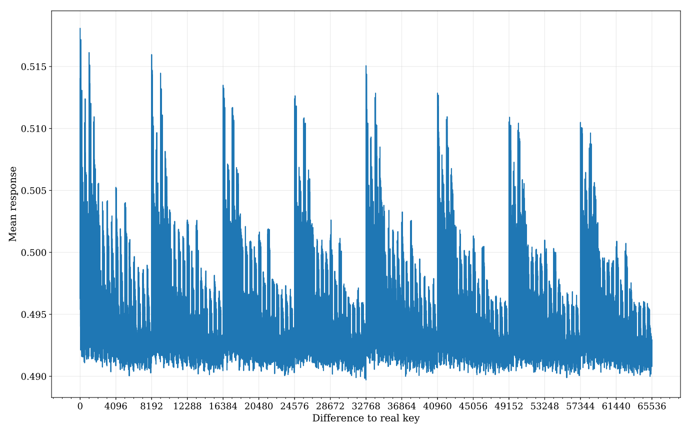
(a) $\mathcal{ND}^{\textsc{Simon}8R}_{\mathbf{VV}}$: directly trained with data of the form $((x, y, x', y'))$



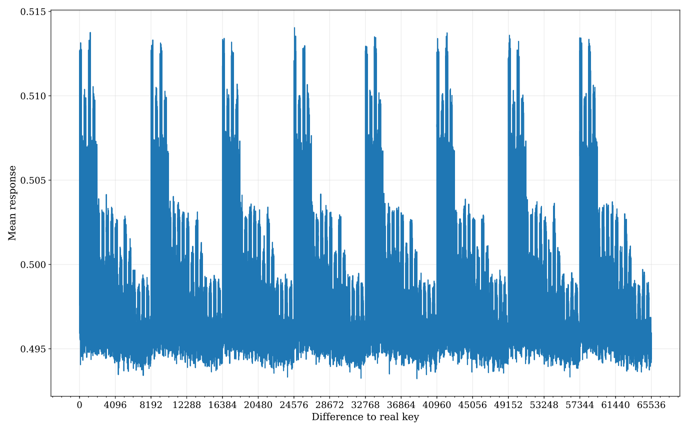
(b) $\mathcal{ND}^{\textsc{Simon}8R}_{\mathbf{VD}}$: directly trained with data of the form $((x, x', y \oplus y'))$



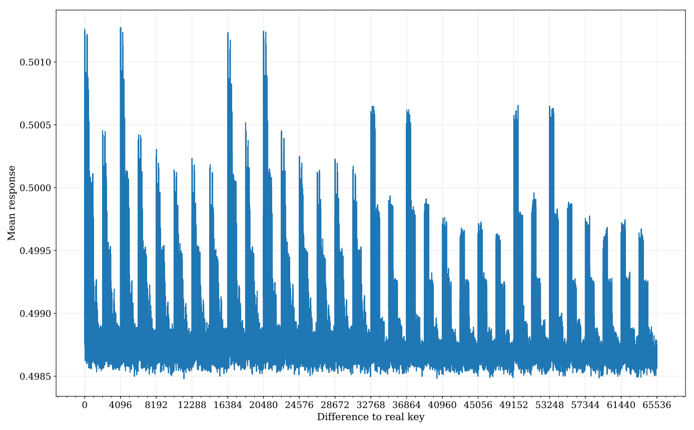
(c) $\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VV}}$: directly trained with data of the form $((x, y, x', y'))$



(d) $\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VD}}$: directly trained with data of the form $((x, x', y \oplus y'))$



(e) $\mathcal{ND}^{\textsc{Simon}10R}_{\mathbf{VV}}$: trained using $\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VV}}$ and KeyAveraging algorithm



(f) $\mathcal{ND}^{\textsc{Simon}11R}_{\mathbf{VV}}$: trained using $\mathcal{ND}^{\textsc{Simon}9R}_{\mathbf{VV}}$ and $\mathcal{DD}^{\textsc{Simon}8R}_{(\texttt{0440,0100})}$ in staged training method

Fig. 4: Wrong key response profile (only μ_δ shown) for neural distinguishers on Simon32/64 (used 2^{14} ciphertexts for (a-e) and 2^{18} for (f))

6 Key-recovery Attacks on Round-Reduced Simon32/64

Under a similar framework to the key-recovery attacks on Speck32/64, the trained neural distinguishers can be prepended with a classical differential to perform key-recovery attacks.

The attack presented in this section, named as $\mathcal{A}_I^{\text{Simon}16R}$, combines the longest but weak neural distinguisher with a differential that has many SNBS.

In Appendix A.1, another attack, named as $\mathcal{A}_{II}^{\text{Simon}16R}$, is presented. It combines relatively strong neural distinguishers with a differential that is one round longer but has fewer neutral bits. $\mathcal{A}_I^{\text{Simon}16R}$ is superior to $\mathcal{A}_{II}^{\text{Simon}16R}$, but $\mathcal{A}_{II}^{\text{Simon}16R}$ achieves better data complexity than previous attacks in literature. It succeeded by using a few of the identified CSNBS of the longer classical differential. .

The classical component in the attack $\mathcal{A}_I^{\text{Simon}16R}$ presented in the sequel is a 3-round differential $(\texttt{0x0440}, \texttt{0x1000}) \rightarrow (\texttt{0x0000}, \texttt{0x0040})$ (prob. $\approx 2^{-8}$).

Similar to attacks on Speck32/64, to obtain decent scores from the responses of the neural distinguishers, combined response from the neural distinguisher over a number of samples from the same distribution are to be used. Thus, to obtain enough samples from the same distribution, neutral bits of the prepended classical differential are exploited.

6.1 Finding Neutral Bits for the Classical Differentials

Finding SNBS for 3-round Differential. For the 3-round differential to be prepended to the neural distinguishers, one can obtain all neutral bits and SNBS (simultaneously complementing up to 4 bits) using the following algebraic method.

Given the input and output differences $(\texttt{0x0440}, \texttt{0x1000})$ and $(\texttt{0x0000}, \texttt{0x0040})$, one can build the non-linear equations on the derivative functions. Because the degrees of the derivative functions corresponding to this 3-round differential is low (*i.e.*, 4), this system of non-linear equations can be solved by computing the Gröbner basis, which can be done using the PolyBoRi library integrated in SageMath [4]. In the obtained Gröbner basis, those disappeared variables correspond to the single-bit neutral bits of the differential. To find SNBS, the following method is used. For each of the 41448 sets (*i.e.*, 32 + 496 + 4960 + 35960 sets) of at most four bits, in the resulted Gröbner basis, replace this set of variables with their complements simultaneously; if the Gröbner basis does not change, the set of variables corresponds to a SNBS.

Using the above algebraic method and experimental double-verification, all the neutral bits and SNBS are obtained. There are 9 single neutral bits [2], [3], [4], [6], [8], [9], [10], [18], [22]}, 2 2-SNBS $\{[0, 24], [12, 26]\}$ (actually, there are 38 2-SNBS; but 36 out 38 are formed by combinations of the 9 single neutral bits); all 3-SNBS and 4-SNBS are formed by combinations of the 9 single neutral bits and 2 2-SNBS. Thus, there are 11 independent neutral bits and SNBS in total.

From the resulted Gröbner basis (also observed by experiments), for an input pair $((x, y), (x', y'))$ to conform the 3-round differential $(\texttt{0x0440}, \texttt{0x1000}) \to (\texttt{0x0000}, \texttt{0x0040})$, one has $\begin{cases} x[1] = x'[1] = 0, \\ x[3] = x'[3] = 0. \end{cases}$ (3)

6.2 Key Recovery Attack on 16-round Simon32/64

The components of $\mathcal{A}_I^{\text{SIMON}_{16R}}$ are as follows (refer to Fig. 6).

1. a 3-round classical differential $(\texttt{0x0440}, \texttt{0x1000}) \to (\texttt{0x0000}, \texttt{0x0040})$ (refer to the rounds colored in blue in Fig. 6), and a set of its 11 NBS {[2], [3], [4], [6], [8], [9], [10], [18], [22], [0, 24],[12, 26] };
2. a 11-round neural distinguisher $\mathcal{ND}_{\mathbf{VV}}^{\text{SIMON}_{11R}}$ trained using the staged approach under difference $(\texttt{0x0000}, \texttt{0x0040})$, and its wrong key response profiles $\mathcal{ND}_{\mathbf{VV}}^{\text{SIMON}_{11R}}.\mu$ and $\mathcal{ND}_{\mathbf{VV}}^{\text{SIMON}_{11R}}.\sigma$.
3. a 9-round neural distinguisher $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}$ trained under difference $(\texttt{0x0000}, \texttt{0x0040})$ and fed with data of type $(x, x', y \oplus y')$, and its wrong key response profiles $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}.\mu$ and $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}.\sigma$.

The goal is to recover the last two subkeys k_{15} and k_{14}. A difference with the attack $\mathcal{A}^{\text{SPECK}_{13R}}$ is that, as one of the neural distinguishers $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}$ accepts data of type $(x, x', y \oplus y')$, after guessing k_{15} and k_{14} and decrypting a ciphertext pair to $(x_{14}, y_{14}), (x'_{14}, y'_{14})$, one can compute $(x_{13}, x'_{13}, y_{13} \oplus y'_{13})$ by inverting one round with 0 as the subkey, and thus can be feed to $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}$.

At the beginning, we guess two key bits of k_0, that is $k_0[1]$ and $k_0[3]$, because for the 3-round differential, the conditions for correct pairs are $x_1[1] = x'_1[1] = 0$ and $x_1[3] = x'_1[3] = 0$ (refer to Eq. 3); no more key bits need to be guessed because the number of non-conditional neutral bits is enough). Thus, n_{kg} is 2, and there are 2^2 outermost loops.

The framework of attack $\mathcal{A}_I^{\text{SIMON}_{16R}}$ is the same as that of $\mathcal{A}^{\text{SPECK}_{13R}}$ on Speck32/64 (refer to Fig. 2). The concrete parameters of the attack are as follows. The accuracy of $\mathcal{ND}_{\mathbf{VV}}^{\text{SIMON}_{11R}}$ is 0.5173, and that of $\mathcal{ND}_{\mathbf{VD}}^{\text{SIMON}_{9R}}$ is 0.5629 (note that $(2 \cdot (0.5173 - 0.5))^2 \approx 2^{-9.71}$. Thus, by Conjecture 1, $n_b = 2^{11}$ should be enough).

$$\begin{array}{llll} n_{kg} = 2^2, & n_{cts} = 2^7, & n_b = 2^{11}, & n_{it} = 2^9 \\ c_1 = 25, & c_2 = 100, & n_{byit1} = n_{byit2} = 5, & n_{cand1} = n_{cand2} = 32 \end{array}$$

The data complexity is $n_{kg} \times n_{cts} \times n_b \times 2$, that is, 2^{21} plaintexts. To examine the performance of the attack, experiments are done using 8 threads on the same GPU server testing $\mathcal{A}^{\text{SPECK}_{13R}}$. In total 99 trials are run[6]. Within the 99 trials, all trials have correct ciphertext pairs and all called the neural distinguishers. There are 49 success trials, for which the returned last two subkeys have a Hamming

[6] They are designed to run 20 trials each; but because of the walltime exceeded the required, they were killed; In total, 99 trials had completed, 7 trials terminated without finish (two almost finished and succeed).

distance to the real subkeys of at most two. Thus, the success rate is computed as 49/99, *i.e.*, 0.49.

The 99 (+7) trials took 78 core hours in total. For a trial, it shows that running full 512 iterations requires less than 1 hour. Thus, the worst case to run 2^2 outermost loops (on guessed values of $k_0[0]$ and $k_0[3]$) for a full attack takes less than 4 GPU hours.

7 Conclusions and Future Work

In this work, potentials of neural distinguishers shown in [8] were further confirmed by the following results.

1. The first practical neural distinguishers-based 13-round key-recovery attack and an improved 12-round key-recovery attack on Speck32/64 were devised, which have considerable advantage in terms of time complexity than attacks devised using orthodox cryptanalaysis;
2. Various neural distinguishers covers up to 11-round Simon32/64 were presented, some of which accept data including partial value and partial difference and turn out to be useful;
3. The first practical neural distinguishers-based 16-round key-recovery attack on Simon32/64 was devised, which have considerable advantage in terms of data complexity than the attack devised using orthodox cryptanalaysis.

These results were achieved by enhancing the classical components in the attack scheme proposed in [8] (for Speck32/64 and Simon32/64) and training new neural distinguishers using various of training schemes (for Simon32/64). The classical components is mainly the neutral bits of the differentials prepended to the neural distinguishers. Generalized neutral bits, including simultaneous-neutral bit-sets and conditional neutral bits, were deeply explored. In doing so, samples are well-structured and the density of positive samples at local points is improved. Using the combinations of responses on the structured samples, weak neural distinguishers could be used to successfully perform key-recovery.

We conjectured (refer to Conj. 1) that, for a neural distinguisher with accuracy of bias ϵ being successfully used in the improved key-recovery attack proposed in [8], it is required to have about $c \times 1/(2\epsilon)^2$ samples in each ciphertext structure (thus requires $\log_2(c \times 1/(2\epsilon)^2)$ neutral bits). This conjecture might provide a guide for future works. Experiments of our attacks support this conjecture. However, a theoretical treatment remains as an interesting open problem.

The key-recovery attack with Upper Confidence Bounds and BayesianKeySearch has shown its effectiveness on guessing keys. However, the parameters, especially the cutoffs, which determine the execution time and success rate, were empirically decided in the situation of lacking theoretical guidance. Thus, the attacks presented might have their better counterparts with a better cutoff settings. It is an interesting future work to use other reinforcement learning methods to design attacks in which the parameters can be adjusted to the best automatically.

References

1. H. A. Alkhzaimi and M. M. Lauridsen. Cryptanalysis of the SIMON Family of Block Ciphers. Cryptology ePrint Archive, Report 2013/543, 2013. http://eprint.iacr.org/2013/543.
2. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. https://eprint.iacr.org/2013/404.
3. E. Biham and R. Chen. Near-Collisions of SHA-0. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 290–305. Springer, Heidelberg, Aug. 2004.
4. M. Brickenstein, A. Dreyer, B. Erocal, M. Albrecht, S. King, and C. Bouillaguet. Sage 9.3 Reference Manual: Polynomials: Boolean Polynomials. https://doc.sagemath.org/html/en/reference/polynomial_rings/sage/rings/polynomial/pbori/pbori.html. Accessed: 2021-5.
5. A. R. Choudhuri and S. Maitra. Differential Cryptanalysis of Salsa and ChaCha – An Evaluation with a Hybrid Model. Cryptology ePrint Archive, Report 2016/377, 2016. https://eprint.iacr.org/2016/377.
6. I. Dinur. Improved Differential Cryptanalysis of Round-Reduced Speck. In A. Joux and A. M. Youssef, editors, *SAC 2014*, volume 8781 of *LNCS*, pages 147–164. Springer, Heidelberg, Aug. 2014.
7. A. Gohr. Implementation of the Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning. GitHub Repository. https://github.com/agohr/deep_speck, 2019.
8. A. Gohr. Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 150–179. Springer, Heidelberg, Aug. 2019.
9. A. Gohr. Improving Attacks on Round-Reduced Speck32/64 using Deep Learning. Cryptology ePrint Archive, Report 2019/037, 2019. https://eprint.iacr.org/2019/037.
10. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.
11. J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu. Squeeze-and-excitation networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(8):2011–2023, 2020.
12. G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017.
13. A. Joux and T. Peyrin. Hash Functions and the (Amplified) Boomerang Attack. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 244–263. Springer, Heidelberg, Aug. 2007.
14. S. Knellwolf, W. Meier, and M. Naya-Plasencia. Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 130–145. Springer, Heidelberg, Dec. 2010.
15. S. K. Langford and M. E. Hellman. Differential-Linear Cryptanalysis. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 17–25. Springer, Heidelberg, Aug. 1994.
16. Stefan Kölbl. CryptoSMT: An easy to use tool for cryptanalysis of symmetric primitives. https://github.com/kste/cryptosmt.

Supplementary Material

A A Second Attack and the Visualization of the Attacks on 16-round Simon32/64

A.1 The Second Attack on 16-round Simon32/64

Details of Find CSNBS for 4-round Differential. For the 4-round differential to be prepended to the neural distinguisher, *i.e.*, $(\texttt{0x1000}, \texttt{0x4440}) \to (\texttt{0x0000}, \texttt{0x0040})$, NB and SNBS are scarce; there are only 2 single NB and 2 2-SNBS.

However, when fixing values of some input bits, more bits become neutral. Given the 9 single NB and the 2 2-SNBS of the 3-round differential $(\texttt{0x0440}, \texttt{0x1000}) \to (\texttt{0x0000}, \texttt{0x0040})$, CSNBS for the 4-round differential can be constructed as follows. Denote the input bits to the 4-round Simon32/64 by $\texttt{x}_{15}$, $\texttt{x}_{14}$, …, $\texttt{x}_0$, $\texttt{y}_{15}$, $\texttt{y}_{14}$, …, $\texttt{y}_0$. To deduce the conditions for these input bits to be neutral for the 4-round differential, one considers one round transformation as depicted in Fig. 5.

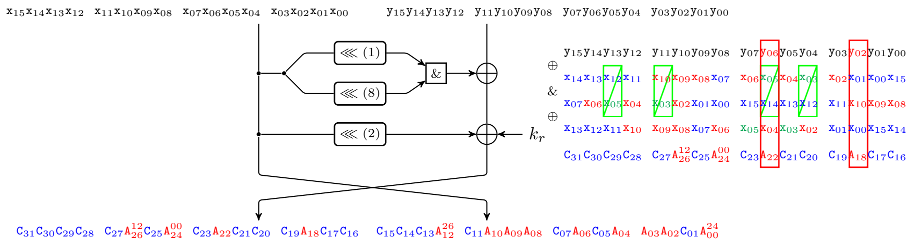


Fig. 5: Deduce conditional simultaneous-neutral bits/bit-sets for 4-round differential of Simon32/64

Indicate the neutrality of bit i by $\texttt{A}_i$ as neutral and $\texttt{C}_i$ as non-neutral for the 3-round differential. Besides, indicate the neutrality of bit i under the condition that simultaneously changing bits j by $\texttt{A}_i^j$ as neutral. Corresponding to the 9 single NB [2], [3], [4], [6], [8], [9], [10], [18], [22] and two 2-SNBS [0, 24], [12, 26] of the 3-round differential, we indicate the neutrality (with respect to the 3-round differential) of the output of the first round of the 4-round as follows

$$\texttt{C}_{31}\texttt{C}_{30}\texttt{C}_{29}\texttt{C}_{28}\ \texttt{C}_{27}\texttt{A}_{26}^{12}\texttt{C}_{25}\texttt{A}_{24}^{00}\ \texttt{C}_{23}\texttt{A}_{22}\texttt{C}_{21}\texttt{C}_{20}\ \texttt{C}_{19}\texttt{A}_{18}\texttt{C}_{17}\texttt{C}_{16}$$
$$\texttt{C}_{15}\texttt{C}_{14}\texttt{C}_{13}\texttt{A}_{12}^{26}\ \texttt{C}_{11}\texttt{A}_{10}\texttt{A}_{09}\texttt{A}_{08}\ \texttt{C}_{07}\texttt{A}_{06}\texttt{C}_{05}\texttt{A}_{04}\ \texttt{A}_{03}\texttt{A}_{02}\texttt{C}_{01}\texttt{A}_{00}^{24}$$

Note that, for an input pair $((x, y), (x', y'))$ to conform the 4-round differential $(\texttt{0x1000}, \texttt{0x4440}) \to (\texttt{0x0000}, \texttt{0x0040})$, one have that $\begin{cases} \texttt{x}_5 = \texttt{x}'_5 = 0, \\ \texttt{x}_3 = \texttt{x}'_3 = 0. \end{cases}$

Firstly, one can directly deduce the 2 single NB and the 2 2-SNBS for the 4-round differential from that of the 3-round differential as follows.

- From the neutrality of $\mathtt{A}_{22}$ and $\mathtt{A}_{18}$ for the 3-round differential, and because
$$\begin{cases} \boxed{\mathtt{y}_{06}} \oplus \mathtt{x}_{05}\mathtt{x}_{14} \oplus \mathtt{x}_{14} = \mathtt{A}_{22}, \\ \boxed{\mathtt{y}_{02}} \oplus \mathtt{x}_{01}\mathtt{x}_{10} \oplus \mathtt{x}_{00} = \mathtt{A}_{18}, \end{cases}$$
we have that $\boxed{\mathtt{y}_{06}}$ and $\boxed{\mathtt{y}_{02}}$ (*i.e.*, [2] and [6]) are neutral for the 4-round differential.
- From the neutrality of $\mathtt{A}_{10}$ and $\mathtt{A}_{18}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{10}$. The variable $\mathtt{x}_{10}$ is involved in the following equations in the one round transformation
$$\begin{cases} \boxed{\mathtt{x}_{10}} = \mathtt{A}_{10}, \\ \mathtt{y}_{02} \oplus \mathtt{x}_{01}\boxed{\mathtt{x}_{10}} \oplus \mathtt{x}_{00} = \mathtt{A}_{18}, \\ \boxed{\mathtt{y}_{12}} \oplus \mathtt{x}_{11}\mathtt{x}_{04} \oplus \boxed{\mathtt{x}_{10}} = \mathtt{C}_{28}, \\ \mathtt{y}_{11} \oplus \boxed{\mathtt{x}_{10}}\mathtt{x}_{03} \oplus \mathtt{x}_{09} = \mathtt{C}_{27}. \end{cases}$$
Because $\mathtt{A}_{10}$ and $\mathtt{A}_{18}$ are neutral for the appended 3-round differential, the first two equations do not impose conditions for $\boxed{\mathtt{x}_{10}}$ to be neutral for the 4-round differential. In the third equation, although $\mathtt{C}_{28}$ is non-neutral for the appended 3-round differential, there is a free variable $\boxed{\mathtt{y}_{12}}$; thus, complementing $\boxed{\mathtt{x}_{10}}$ and $\boxed{\mathtt{y}_{12}}$ simultaneously does not violate the non-neutrality of $\mathtt{C}_{28}$. In the fourth equation, because $\mathtt{x}_{03}$ must be zero for all conforming pairs of the 4-round differential, complementing $\boxed{\mathtt{x}_{10}}$ does not violate the non-neutrality of $\mathtt{C}_{27}$. Thus, we have that $\boxed{\mathtt{x}_{10}}$ and $\boxed{\mathtt{y}_{12}}$ (*i.e.*, [12, 26]) form a SNBS for the 4-round differential.
- From the simultaneous-neutrality of $\mathtt{A}_{12}^{26}$ and $\mathtt{A}_{26}^{12}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{12}$ and $\mathtt{y}_{10}$. Because $\mathtt{x}_{12}$ and $\mathtt{y}_{10}$ are involved in the following equations in the one round transformation
$$\begin{cases} \boxed{\mathtt{x}_{12}} = \mathtt{A}_{12}^{26}, \\ \boxed{\mathtt{y}_{10}} \oplus \mathtt{x}_{09}\mathtt{x}_{02} \oplus \mathtt{x}_{08} = \mathtt{A}_{26}^{12}, \\ \boxed{\mathtt{y}_{14}} \oplus \mathtt{x}_{13}\mathtt{x}_{06} \oplus \boxed{\mathtt{x}_{12}} = \mathtt{C}_{30}, \\ \mathtt{y}_{04} \oplus \mathtt{x}_{03}\boxed{\mathtt{x}_{12}} \oplus \mathtt{x}_{02} = \mathtt{C}_{20}, \text{ where } \mathtt{x}_{03} = 0, \end{cases}$$
similar to the above analysis, we have that $\boxed{\mathtt{x}_{12}}$, $\boxed{\mathtt{y}_{10}}$, and $\boxed{\mathtt{y}_{14}}$ (*i.e.*, [10, 14, 28]) form a SNBS for the 4-round differential.

Next, let consider the conditional neutrality of input bits for the 4-round differential.

- From the neutrality of $\mathtt{A}_{08}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{08}$. The variable $\mathtt{x}_{08}$ is involved in the following equations in the one round transformation
$$\begin{cases} \boxed{\mathtt{x}_{08}} = \mathtt{A}_{08}, \\ \boxed{\mathtt{y}_{10}} \oplus \mathtt{x}_{09}\mathtt{x}_{02} \oplus \boxed{\mathtt{x}_{08}} = \mathtt{C}_{26}, \\ \boxed{\mathtt{y}_{09}} \oplus \boxed{\mathtt{x}_{08}}\mathtt{x}_{01} \oplus \mathtt{x}_{07} = \mathtt{C}_{25}, \\ \boxed{\mathtt{y}_{00}} \oplus \mathtt{x}_{15}\boxed{\mathtt{x}_{08}} \oplus \mathtt{x}_{14} = \mathtt{C}_{16}. \end{cases}$$
In the second equation, to not violate the non-neutrality of $\mathtt{C}_{26}$, one should complement

$\boxed{\mathtt{x}_{08}}$ and $\boxed{\mathtt{y}_{10}}$ simultaneously. In the third equation, to not violate the non-neutrality of $\mathtt{C}_{25}$, when $\mathtt{x}_{01} = 0$, one can freely complement $\boxed{\mathtt{x}_{08}}$; when $\mathtt{x}_{01} = 1$, one should complement $\boxed{\mathtt{x}_{08}}$ and $\boxed{\mathtt{y}_{09}}$ simultaneously. In the fourth equation, to not violate the non-neutrality of $\mathtt{C}_{16}$, when $\mathtt{x}_{15} = 0$, one can freely complement $\boxed{\mathtt{x}_{08}}$; when $\mathtt{x}_{15} = 1$, one should complement $\boxed{\mathtt{x}_{08}}$ and $\boxed{\mathtt{y}_{00}}$ simultaneously. Therefore, we have a CSNBS for the 4-round differential as follows

$$\begin{cases} [\mathtt{x}_{08}, \mathtt{y}_{10}],\ \textit{i.e.},\ [24, 10] & (\mathtt{x}_{01}, \mathtt{x}_{15}) = (0, 0); \\ [\mathtt{x}_{08}, \mathtt{y}_{10}, \mathtt{y}_{09}],\ \textit{i.e.},\ [24, 10, 9] & (\mathtt{x}_{01}, \mathtt{x}_{15}) = (1, 0); \\ [\mathtt{x}_{08}, \mathtt{y}_{10}, \mathtt{y}_{00}],\ \textit{i.e.},\ [24, 10, 0] & (\mathtt{x}_{01}, \mathtt{x}_{15}) = (0, 1); \\ [\mathtt{x}_{08}, \mathtt{y}_{10}, \mathtt{y}_{09}, \mathtt{y}_{00}],\ \textit{i.e.},\ [24, 10, 9, 0] & (\mathtt{x}_{01}, \mathtt{x}_{15}) = (1, 1). \end{cases}$$

- From the neutrality of $\mathtt{A}_{06}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{06}$. The variable $\mathtt{x}_{06}$ is involved in the following equations in the one round transformation $\begin{cases} \boxed{\mathtt{x}_{06}} = \mathtt{A}_{06}, \\ \boxed{\mathtt{y}_{08}} \oplus \mathtt{x}_{07}\mathtt{x}_{00} \oplus \boxed{\mathtt{x}_{06}} = \mathtt{C}_{24}, \\ \boxed{\mathtt{y}_{07}} \oplus \boxed{\mathtt{x}_{06}}\mathtt{x}_{15} \oplus \mathtt{x}_{05} = \mathtt{C}_{23}, \\ \boxed{\mathtt{y}_{14}} \oplus \mathtt{x}_{13}\boxed{\mathtt{x}_{06}} \oplus \mathtt{x}_{12} = \mathtt{C}_{30}. \end{cases}$ Similar to the above analysis, we have a CSNBS for the 4-round differential as follows

$$\begin{cases} [\mathtt{x}_{06}, \mathtt{y}_{08}],\ \textit{i.e.},\ [22, 8] & (\mathtt{x}_{15}, \mathtt{x}_{13}) = (0, 0); \\ [\mathtt{x}_{06}, \mathtt{y}_{08}, \mathtt{y}_{07}],\ \textit{i.e.},\ [22, 8, 7] & (\mathtt{x}_{15}, \mathtt{x}_{13}) = (1, 0); \\ [\mathtt{x}_{06}, \mathtt{y}_{08}, \mathtt{y}_{14}],\ \textit{i.e.},\ [22, 8, 14] & (\mathtt{x}_{15}, \mathtt{x}_{13}) = (0, 1); \\ [\mathtt{x}_{06}, \mathtt{y}_{08}, \mathtt{y}_{07}, \mathtt{y}_{14}],\ \textit{i.e.},\ [22, 8, 7, 14] & (\mathtt{x}_{15}, \mathtt{x}_{13}) = (1, 1). \end{cases}$$

- From the neutrality of $\mathtt{A}_{04}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{04}$. The variable $\mathtt{x}_{04}$ is involved in the following equations in the one round transformation $\begin{cases} \boxed{\mathtt{x}_{04}} = \mathtt{A}_{04}, \\ \mathtt{y}_{06} \oplus \mathtt{x}_{05}\mathtt{x}_{14} \oplus \boxed{\mathtt{x}_{04}} = \mathtt{A}_{22}, \\ \boxed{\mathtt{y}_{05}} \oplus \boxed{\mathtt{x}_{04}}\mathtt{x}_{13} \oplus \mathtt{x}_{03} = \mathtt{C}_{21}, \\ \boxed{\mathtt{y}_{12}} \oplus \mathtt{x}_{11}\boxed{\mathtt{x}_{04}} \oplus \mathtt{x}_{10} = \mathtt{C}_{28}. \end{cases}$ Similar to the above analysis, we have a CSNBS for the 4-round differential as follows

$$\begin{cases} [\mathtt{x}_{04}],\ \textit{i.e.},\ [20] & (\mathtt{x}_{13}, \mathtt{x}_{11}) = (0, 0); \\ [\mathtt{x}_{04}, \mathtt{y}_{05}],\ \textit{i.e.},\ [20, 5] & (\mathtt{x}_{13}, \mathtt{x}_{11}) = (1, 0); \\ [\mathtt{x}_{04}, \mathtt{y}_{12}],\ \textit{i.e.},\ [20, 12] & (\mathtt{x}_{13}, \mathtt{x}_{11}) = (0, 1); \\ [\mathtt{x}_{04}, \mathtt{y}_{05}, \mathtt{y}_{12}],\ \textit{i.e.},\ [20, 5, 12] & (\mathtt{x}_{13}, \mathtt{x}_{11}) = (1, 1). \end{cases}$$

- From the neutrality of $\mathtt{A}_{02}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{02}$. The variable $\mathtt{x}_{02}$ is involved in the following equations in the one round transformation $\begin{cases} \boxed{\mathtt{x}_{02}} = \mathtt{A}_{02}, \\ \boxed{\mathtt{y}_{04}} \oplus \mathtt{x}_{03}\mathtt{x}_{12} \oplus \boxed{\mathtt{x}_{02}} = \mathtt{C}_{20}, \\ \boxed{\mathtt{y}_{03}} \oplus \boxed{\mathtt{x}_{02}}\mathtt{x}_{11} \oplus \mathtt{x}_{01} = \mathtt{C}_{19}, \\ \boxed{\mathtt{y}_{10}} \oplus \mathtt{x}_{09}\boxed{\mathtt{x}_{02}} \oplus \mathtt{x}_{08} = \mathtt{C}_{26}. \end{cases}$ Similar to the above analysis, we have a CSNBS for the 4-round differential as follows

$$\begin{cases} [\mathtt{x}_{02}, \mathtt{y}_{04}], \;\; i.e., \; [18, 4] & (\mathtt{x}_{11}, \mathtt{x}_{09}) = (0, 0); \\ [\mathtt{x}_{02}, \mathtt{y}_{04}, \mathtt{y}_{03}], \;\; i.e., \; [18, 4, 3] & (\mathtt{x}_{11}, \mathtt{x}_{09}) = (1, 0); \\ [\mathtt{x}_{02}, \mathtt{y}_{04}, \mathtt{y}_{10}], \;\; i.e., \; [18, 4, 10] & (\mathtt{x}_{11}, \mathtt{x}_{09}) = (0, 1); \\ [\mathtt{x}_{02}, \mathtt{y}_{04}, \mathtt{y}_{03}, \mathtt{y}_{10}], \;\; i.e., \; [18, 4, 3, 10] & (\mathtt{x}_{11}, \mathtt{x}_{09}) = (1, 1). \end{cases}$$

– From the simultaneous neutrality of $\mathtt{A}_{00}^{24}$ and $\mathtt{A}_{24}^{00}$ for the 3-round differential, let us consider the neutrality of $\mathtt{x}_{00}$ and $\mathtt{y}_{08}$. The variable $\mathtt{x}_{00}$ and $\mathtt{y}_{08}$ are involved in the following equations in the one round transformation

$$\begin{cases} \boxed{\mathtt{x}_{00}} = \mathtt{A}_{00}^{24}, \\ \mathtt{y}_{02} \oplus \mathtt{x}_{01}\mathtt{x}_{10} \oplus \boxed{\mathtt{x}_{00}} = \mathtt{A}_{18}, \\ \boxed{\mathtt{y}_{01}} \oplus \boxed{\mathtt{x}_{00}}\mathtt{x}_{09} \oplus \mathtt{x}_{15} = \mathtt{C}_{17}, \\ \boxed{\mathtt{y}_{08}} \oplus \mathtt{x}_{07}\boxed{\mathtt{x}_{00}} \oplus \mathtt{x}_{06} = \mathtt{A}_{24}^{00}. \end{cases}$$

Similar to the above analysis, we have a CSNBS for the 4-round differential as follows

$$\begin{cases} [\mathtt{x}_{00}, \mathtt{y}_{08}], \;\; i.e., \; [16, 8] & (\mathtt{x}_{09}, \mathtt{x}_{07}) = (0, 0); \\ [\mathtt{x}_{00}, \mathtt{y}_{08}, \mathtt{y}_{01}], \;\; i.e., \; [16, 8, 1] & (\mathtt{x}_{09}, \mathtt{x}_{07}) = (1, 0); \\ [\mathtt{x}_{00}], \;\; i.e., \; [16] & (\mathtt{x}_{09}, \mathtt{x}_{07}) = (0, 1); \\ [\mathtt{x}_{00}, \mathtt{y}_{01}], \;\; i.e., \; [16, 1] & (\mathtt{x}_{09}, \mathtt{x}_{07}) = (1, 1). \end{cases}$$

In summary, for the 4-round differential $(0x1000, 0x4440) \to (0x0000, 0x0040)$ of SIMON32/64, there are 9 complete NB/SNBS/CSNBS, that is

1. 2 single NB: [2], [6]
2. 2 SNBS: [12, 26], [10, 14, 28]
3. 5 CSNBS in Table 5.

Table 5: CSNBS for the 4-round differential (0x1000, 0x4440) → (0x0000, 0x0040) of SIMON32/64

Bit-set	C.	Bit-set	C.	Bit-set	C.	Bit-set	C.	Bit-set	C.
$x[1, 15]$		$x[15, 13]$		$x[13, 11]$		$x[11, 9]$		$x[9, 7]$	
[24, 10],	00	[22, 8],	00	[20],	00	[18, 4],	00	[16, 8],	00
[24, 10, 9],	10	[22, 8, 7],	10	[20, 5],	10	[18, 4, 3],	10	[16, 8, 1]	10
[24, 10, 0],	01	[22, 8, 14],	01	[20, 12],	01	[18, 4, 10],	01	[16],	01
[24, 10, 9, 0]	11	[22, 8, 7, 14]	11	[20, 12, 5]	11	[18, 4, 3, 10]	11	[16, 1]	11

C.: Conditions on $x[i, j]$, *e.g.*, $x[i, j] = 10$ means $x[i] = 1$ and $x[j] = 0$.

□: CSNBS that are used in the 16-round attack $\mathcal{A}_{II}^{\text{SIMON}16R}$ on SIMON32/64.

Note that, for an input pair $((x, y), (x', y'))$ to conform the 4-round differential (0x1000, 0x4440) → (0x0000, 0x0040), one have that

$$\begin{cases} x[5] = x'[5] = 0, \\ x[3] = x'[3] = 0. \end{cases} \quad (4)$$

The attack $\mathcal{A}_{II}^{\text{Simon}_{16R}}$. The components of $\mathcal{A}_{II}^{\textsc{Simon}_{16R}}$ are as follows (refer to Fig. 7).

1. a 4-round classical differential $(\texttt{0x1000}, \texttt{0x4440}) \to (\texttt{0x0000}, \texttt{0x0040})$ (refer to the rounds colored in blue in Fig. 7), and a set of its $4 + 3$ neutral bit(-set)s (*i.e.*, four non-conditional ones $[2]$, $[6]$, $[12, 26]$, $[10, 14, 28]$ and the three ones conditioned on $x[15, 13]$, $x[13, 11]$, $x[11, 9]$ (refer to the columns framed by green lines in Table 5));
2. a 9-round neural distinguisher $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{9R}}$ trained under difference $(\texttt{0x0000}, \texttt{0x0040})$ and fed with data of type $(x, x', y \oplus y')$, and its wrong key response profiles $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{9R}}.\mu$ and $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{9R}}.\sigma$;
3. a 8-round neural distinguisher $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{8R}}$ trained under difference $(\texttt{0x0000}, \texttt{0x0040})$ and fed with data of type $(x, x', y \oplus y')$. and its wrong key response profiles $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{8R}}.\mu$ and $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{8R}}.\sigma$.

The goal is to recover the last two subkeys k_{15} and k_{14}. A difference with the attack $\mathcal{A}^{\textsc{Speck}_{13R}}$ and $\mathcal{A}_{I}^{\textsc{Simon}_{16R}}$ is that, as the neural distinguishers accept data of type $(x, x', y \oplus y')$, after guessing k_{15} and decrypting a ciphertext pair to $(x_{15}, y_{15}), (x'_{15}, y'_{15})$, one can compute $(x_{14}, x'_{14}, y_{14} \oplus y'_{14})$ by inverse one round with 0 as the subkey, and thus can be fed to $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{9R}}$. After guessing k_{14} and decrypting a pair of $(x_{15}, y_{15}), (x'_{15}, y'_{15})$ to $(x_{14}, y_{14}), (x'_{14}, y'_{14})$, one can compute $(x_{13}, x'_{13}, y_{13} \oplus y'_{13})$ by inverse one round with 0 as the subkey, and thus can be fed to $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{8R}}$.

Another difference is that, at the beginning, we guessed two key bits of k_0, that is $k_0[3]$ and $k_0[5]$, because for the 4-round differentials, the conditions for correct pairs is $x_1[5] = x'_1[5] = 0$ and $x_1[3] = x'_1[3] = 0$ (refer to Eq. 4); and four key bits $k_0[15]$, $k_0[13]$, $k_0[11]$, $k_0[9]$ for employing three conditional neutral bits (refer to Table 5). For different values of the chosen data pairs $(\tilde{x}_1, \tilde{y}_1)$, with guessed values of the four key bits, we choose different neutral bit-sets to generate the structures.

Thus, n_{kg} is $2 + 4$, and there are 2^6 outermost loops.

Except these differences, other part of the framework of attack $\mathcal{A}_{II}^{\textsc{Simon}_{16R}}$ is the same as that of the 13-round attack $\mathcal{A}^{\textsc{Speck}_{13R}}$ on Speck32/64. The concrete parameters of the attack are as follows. The accuracy of $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{9R}}$ is 0.5629, and that of $\mathcal{ND}_{\mathbf{VD}}^{\textsc{Simon}_{8R}}$ is 0.6587 (note that $(2 \cdot (0.5629 - 0.5))^2 \approx 2^{-5.98}$. Thus, by Conjecture 1, $n_b = 2^7$ should be enough).

$$\begin{array}{llll} n_{kg} = 2^{2+4}, & n_{cts} = 2^{10}, & n_b = 2^{4+3}, & n_{it} = 2^{11} \\ c_1 = 20, & c_2 = 70, & n_{byit1} = n_{byit2} = 5, & n_{cand1} = n_{cand2} = 32 \end{array}$$

The data complexity is $n_{kg} \times n_{cts} \times n_b \times 2$, that is, 2^{24} plaintexts. To examine the performance of the attack, experiments were done using 8 threads on the same GPU server testing $\mathcal{A}^{\textsc{Speck}_{13R}}$. Each thread ran 20 trails, thus, 160 trails were run in total. Within the 160 trails, all trails have correct ciphertext pairs and all called the neural distinguishers. There are 52 success trails (the returned last two subkeys have hamming distance to the real subkeys at most two). Thus, the success rate is computed as 52/160, *i.e.*, 0.325.

The 160 trails took 120.25 core hours in total. For a trail, running full 2048 iterations requires less than 1 hour (about 50 minutes). Thus, the worst case to run 2^6 outermost loops for a full attack should take roughly 64 GPU hours.

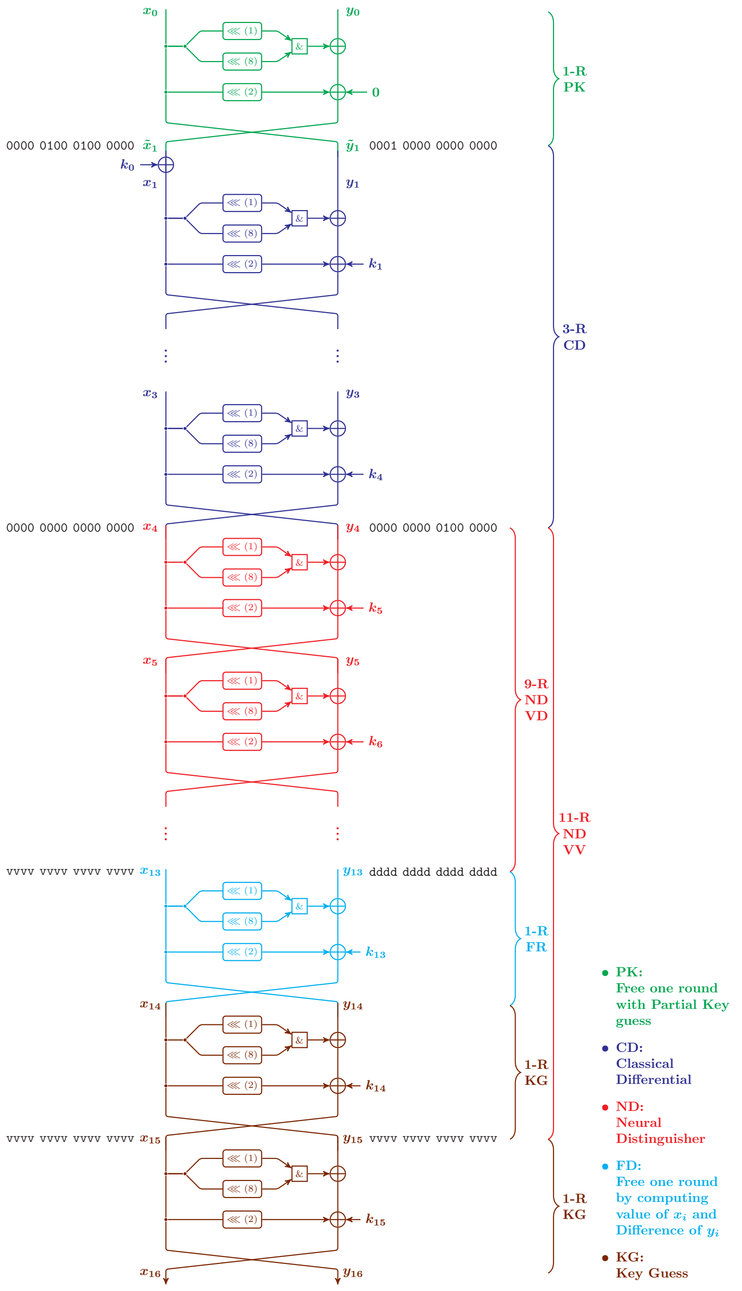
A.2 Visualization of the Attacks

B Details of the Key-recovery Attack in [9]

B.1 Neutral bits Used in [9]

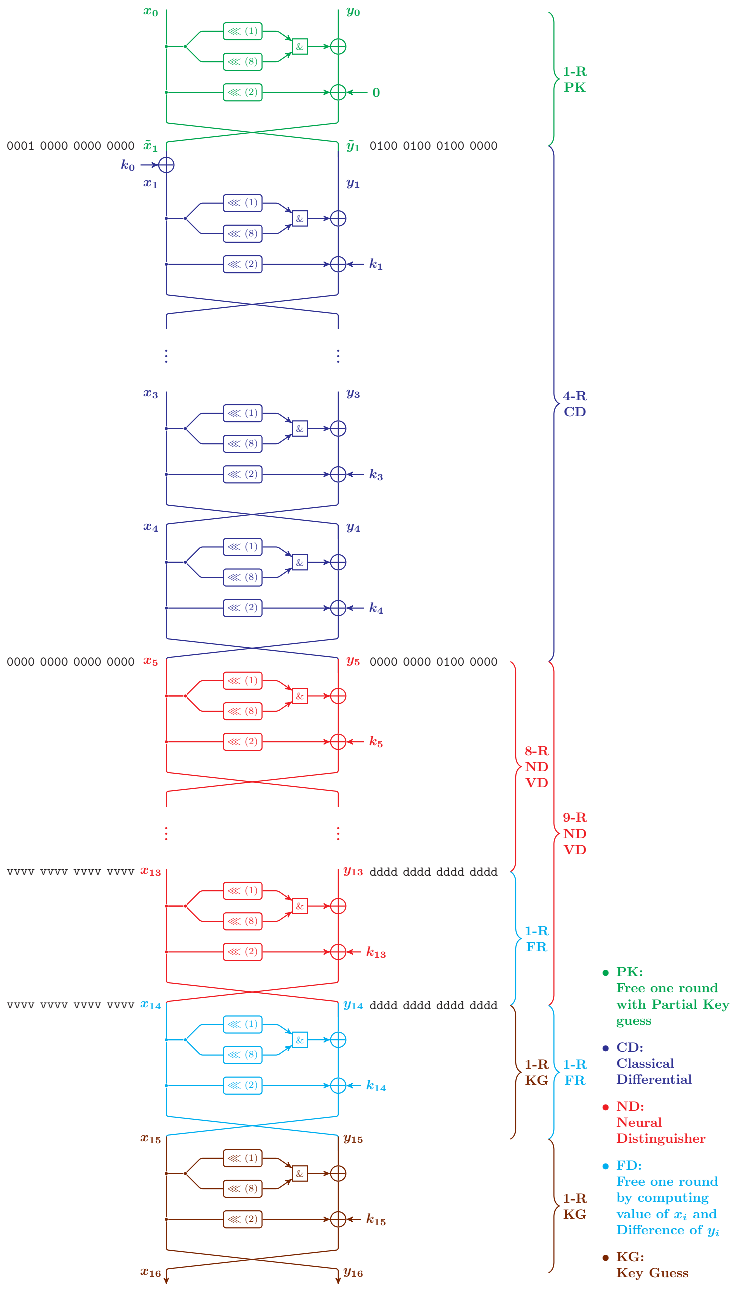
Table 6: (Probabilistic) single-bit neutral bit for 2-round differential $(\texttt{0x0211}, \texttt{0x0a04}) \to (\texttt{0x0040}, \texttt{0x0000})$ of Speck32/64

NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.	NBs	Pr.
[20]	1	[21]	1	[22]	1	[14]	0.965	[15]	0.938	[23]	0.812	[7]	0.806
[30]	0.809	[0]	0.763	[8]	0.664	[24]	0.649	[31]	0.644	[1]	0.574		



x_0
y_0
$\lll (1)$
$\lll (8)$
$\lll (2)$
&
0
1-R PK
0000 0100 0100 0000 $\tilde{x}_1$
$\tilde{y}_1$ 0001 0000 0000 0000
k_0
x_1
y_1
k_1
3-R CD
x_3
y_3
k_4
0000 0000 0000 0000 x_4
y_4 0000 0000 0100 0000
k_5
x_5
y_5
k_6
9-R ND VD
11-R ND VV
vvvv vvvv vvvv vvvv x_{13}
y_{13} dddd dddd dddd dddd
k_{13}
1-R FR
x_{14}
y_{14}
k_{14}
1-R KG
vvvv vvvv vvvv vvvv x_{15}
y_{15} vvvv vvvv vvvv vvvv
k_{15}
1-R KG
x_{16}
y_{16}
● PK: Free one round with Partial Key guess
● CD: Classical Differential
● ND: Neural Distinguisher
● FD: Free one round by computing value of x_i and Difference of y_i
● KG: Key Guess

Fig. 6: Components for key-recovery attack $\mathcal{A}_I^{\text{SIMON}16R}$ on 16-round SIMON32/64



x_0
y_0
$\lll (1)$
$\lll (8)$
$\lll (2)$
&
0
1-R PK
0001 0000 0000 0000 $\tilde{x}_1$
$\tilde{y}_1$ 0100 0100 0100 0000
k_0
x_1
y_1
k_1
x_3
y_3
k_3
x_4
y_4
k_4
4-R CD
0000 0000 0000 0000 x_5
y_5 0000 0000 0100 0000
k_5
8-R ND VD
9-R ND VD
vvvv vvvv vvvv vvvv x_{13}
y_{13} dddd dddd dddd dddd
k_{13}
1-R FR
vvvv vvvv vvvv vvvv x_{14}
y_{14} dddd dddd dddd dddd
k_{14}
1-R KG
1-R FR
x_{15}
y_{15}
k_{15}
1-R KG
x_{16}
y_{16}
● PK: Free one round with Partial Key guess
● CD: Classical Differential
● ND: Neural Distinguisher
● FD: Free one round by computing value of x_i and Difference of y_i
● KG: Key Guess

Fig. 7: Components for key-recovery attack $\mathcal{A}_{II}^{\text{SIMON}16R}$ on 16-round SIMON32/64