electronics

MDPI

Article

Truncated Differential-Neural Key Recovery Attacks on Round-Reduced HIGHT

Byoungjin Seok

School of Cybersecurity, Korea University, Seoul 02841, Republic of Korea; bjseok@korea.ac.kr

Abstract: Recently, differential-neural cryptanalysis, which integrates deep learning with differential cryptanalysis, has emerged as a powerful and practical cryptanalysis method. It has been particularly applied to lightweight block ciphers, which are characterized by simple structures and operations, and relatively small block and key sizes. In resource-constrained environments, such as Internet of Things (IoT), it is essential to verify the resistance of existing lightweight block ciphers against differential-neural cryptanalysis to ensure security. In differential-neural cryptanalysis, a deep learning model, known as a neural distinguisher, is trained to differentiate a target cipher from others, facilitating key recovery through statistical analysis. For successful differential-neural cryptanalysis, it is crucial to develop a highly accurate neural distinguisher and to optimize the key recovery attack algorithm. In this paper, we introduce a novel neural distinguisher and key recovery attack against the 15-round reduced HIGHT cipher. Our proposed neural distinguisher is capable of distinguishing HIGHT ciphertext by analyzing only a portion of the ciphertext, which we refer to as a truncated neural distinguisher. Notably, our experiments demonstrate that the truncated neural distinguisher achieves performance comparable to existing distinguishers trained on entire ciphertext blocks, while enabling a more efficient key recovery attack through a divide-and-conquer strategy. Furthermore, we observe a significant improvement in key recovery efficiency compared to traditional cryptanalysis methods.

Keywords: differential-neural cryptanalysis; neural distinguisher; neural key recovery attack; lightweight block ciphers; HIGHT

Citation: Seok, B. Truncated Differential-Neural Key Recovery Attacks on Round-Reduced HIGHT. *Electronics* **2024**, *13*, 4053. https://doi.org/10.3390/electronics13204053

Academic Editor: Andreas Mauthe

Received: 5 September 2024
Revised: 1 October 2024
Accepted: 14 October 2024
Published: 15 October 2024

1. Introduction

Deep learning has shown excellent outcomes in various fields such as object detection [1,2], natural language processing [3], and the decisions of trained models, which are quite correct (with accuracies of greater than 90%). In the field of cryptography, machine learning has been applied to cryptanalysis for decades, with most of the focus on side-channel attacks [4–6]. Recently, there have been attempts to apply deep learning techniques to the field of cryptanalysis, which focuses on analyzing the security of cryptographic algorithms. Notably, at CRYPTO '19, A. Gohr [7] was the first to introduce differential-neural cryptanalysis, a novel approach that combines deep learning with the traditional cryptanalysis technique of differential cryptanalysis [8]. In differential-neural cryptanalysis, a dataset of ciphertext pairs, generated based on specific differential characteristics, is used to train a deep learning model. This trained model, known as a neural distinguisher, is capable of distinguishing whether the input ciphertext satisfies the learned differential characteristics. Through statistical analysis of the neural distinguisher's results, A. Gohr successfully conducted a key recovery attack on 11-round SPECK-32/64, demonstrating that this approach was more efficient than traditional differential cryptanalysis. Following this development, differential-neural cryptanalysis has garnered significant attention in the field of cryptanalysis, leading to continuous research efforts, including improvements in neural distinguishers [9–14], key recovery attacks [15,16], and extensions to other block ciphers [17–19].

Meanwhile, lightweight block ciphers are being actively developed and standardized (including PRESENT [20], LEA [21], HIGHT [22], CLEFIA [23], and SPECK/SIMON [24]) as a key security technology for providing confidentiality in resource-constrained environments, such as IoT. Lightweight block ciphers are designed with simpler structures and operations, as well as smaller block and key sizes compared to general block ciphers, to ensure efficiency. While these characteristics offer advantages in terms of implementation and processing, they also make lightweight block ciphers susceptible to differential-neural cryptanalysis due to their simplicity. Since most lightweight block ciphers were not designed with resistance to deep learning-based attacks in mind, it is essential to conduct security evaluations against such threats to ensure their safe operation. In this regard, we conduct differential-neural cryptanalysis on the standard lightweight block cipher HIGHT, which was proposed at CHES 2005.

HIGHT, our target cipher, is one of the most representative Addition-Rotation-XOR(ARX)-based lightweight block ciphers. It is included to the open source cryptographic library CRYPTO++ [25], which is used world-widely. The security of HIGHT has been analyzed theoretically through various cryptanalytic methods [22,26–28]. However, these research results are all about theoretical security, and it is difficult to apply from a practical perspective. Additionally, although research on differential-neural cryptanalysis exists, including the 14-round neural distinguisher study by D. Pal et al. [29], its practical application to key recovery attacks has not yet been conducted (a more detailed explanation of this work can be found in Section 2.3). Therefore, in this paper, we develop an improved neural distinguisher and, based on this, propose a new neural key recovery attack algorithm against the 15-round reduced HIGHT. The contributions of this paper can be summarized as follows:

- We present a truncated neural distinguisher that learns only a part of the HIGHT block to distinguish real ciphertexts from others. The truncated neural distinguisher showed the same accuracy as the neural distinguisher, which learned the full ciphertext block of HIGHT. This means that full block information is not necessary to train neural distinguisher. To the best of our knowledge, this type of neural distinguisher was presented for the first time in this paper. The approach for making a truncated neural distinguisher is not dedicated to HIGHT, so it can be extended to various block ciphers (especially, we expect that it is effective against generalized Feistel-based ciphers).
- The results of our key recovery attack are the first differential-neural cryptanalysis-based key recovery attack on HIGHT, and show improved efficiency compared to the traditional differential key recovery attack.

The remainder of this paper is organized as follows. Section 2 provides the preliminaries for understanding our proposed attack, including a brief description of HIGHT and A. Gohr's differential-neural cryptanalysis. Section 3 presents our approaches to make neural distinguishers of HIGHT. Section 4 presents a new key recovery attack on 15-round HIGHT based on our truncated neural distinguisher and experiment results. In Section 5, we discuss the resistance of HIGHT to differential-neural cryptanalysis based on the results of this paper, as well as future directions. Finally, we conclude this paper and draw the future directions in Section 6.

2. Preliminaries

2.1. A Brief Description of Lightweight Block Cipher HIGHT

HIGHT is an ARX-based lightweight block cipher designed to use only modular addition, XOR, and rotation operations, making it efficiently implementable in resource-constrained computing environments such as Radio-Frequency Identification (RFID) and Ubiquitous Sensor Network (USN). The overall structure is based on a generalized Feistel network [30]. The block size of HIGHT is 64 bits, and the key size is 128 bits. The plaintext P and ciphertext C is treated as a concatenation of bytes such as $P = P_7||\ldots||P_1||P_0$ and $C = C_7||C_6||\ldots||C_1||C_0$. Similarly, the master key is denoted as $MK = MK_{16}||MK_{15}||\ldots||MK_1||MK_0$. In addition, the operations are performed on a byte-level, specifically utilizing a

modular addition in 2^8 (denoted as $\boxplus$), XOR (denoted as $\oplus$), and 8-bit left rotation (denoted as $\lll$).

The encryption process consists of three procedures including Initial Transformation, Round Function, and Final Transformation. In here, the whitening key WK and subkey SK, which are derived from MK using the key schedule (Algorithm 1). Let the i-th round internal state in encryption process be $X_i = X_{i,7}||X_{i,6}||\ldots||X_{i,1}||X_{i,0}$. The initial and final transformations are performed once at the beginning and finish, respectively. The round function, which is shown in Figure 1, is performed 32 times. Here, F_0 and F_1 are defined as follows:

$$\begin{aligned} F_0(x) &= x^{\lll 1} \oplus x^{\lll 2} \oplus x^{\lll 7}, \\ F_1(x) &= x^{\lll 3} \oplus x^{\lll 4} \oplus x^{\lll 6}. \end{aligned} \tag{1}$$

Algorithm 2 specifies the overall process of HIGHT encryption. In addition, the decryption process uses a modular subtraction in 2^8 $\boxminus$ instead of $\boxplus$, and performs the encryption process in the opposite direction using the reverse order of WK and SK.

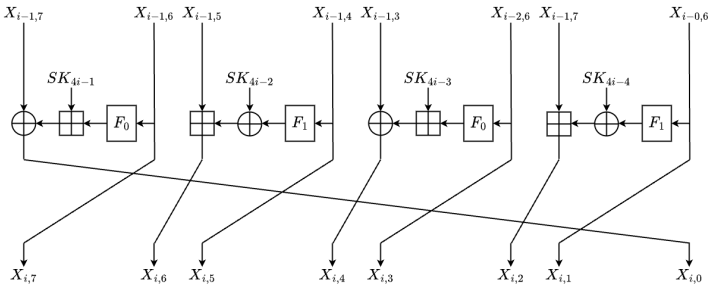

Figure 1. The i-th round function of HIGHT

Algorithm 1 Key Schedule of HIGHT

KeySchedule(MK) {
  // Whitening Key Generation
  For $i = 0$ to 7 {
    If $0 \leq i \leq 3$, then $WK_i \leftarrow MK_{i+12}$;
    Else, $WK_i \leftarrow MK_{i-4}$;
  }
  // Constant Generation (δ)
  $s_0 \leftarrow 0_2$; $s_1 \leftarrow 1_2$; $s_2 \leftarrow 0_2$; $s_3 \leftarrow 1_2$; $s_4 \leftarrow 1_2$;
  $s_5 \leftarrow 0_2$; $s_6 \leftarrow 1_2$;
  $\delta_0 \leftarrow s_6||s_5||s_4||s_3||s_2||s_1||s_0$;
  For $i = 0$ to 127 {
    $s_{i+6} \leftarrow s_{i+2} \oplus s_{i-1}$;
    $\delta_i \leftarrow s_6||s_5||s_4||s_3||s_2||s_1||s_0$;
  }
  // Subkey Generation
  For $i = 0$ to 7 {
    For $j = 0$ to 7 {
      $SK_{16i+j} \leftarrow MK_{j-i \mod 8} \boxplus \delta_{16i+j}$;
    }
    For $j = 0$ to 7 {
      $SK_{16i+j+8} \leftarrow MK_{(j-i \mod 8)+8} \boxplus \delta_{16i+j+8}$;
    }
  }
  return WK, SK
}

Algorithm 2 Encryption of HIGHT

HIGHTENCRYPTION(P, WK, SK) {
// Initial Transformation
$X_{0,0} \leftarrow P_0 \boxplus WK_0$; $X_{0,1} \leftarrow P_1$;
$X_{0,2} \leftarrow P_2 \oplus WK_1$; $X_{0,3} \leftarrow P_3$;
$X_{0,4} \leftarrow P_4 \boxplus WK_2$; $X_{0,5} \leftarrow P_5$;
$X_{0,6} \leftarrow P_6 \oplus WK_3$; $X_{0,7} \leftarrow P_7$;
// Round Function
For $i = 0$ to 31 {
$X_{i+1,1} \leftarrow X_{i,0}$; $X_{i+1,3} \leftarrow X_{i,2}$;
$X_{i+1,5} \leftarrow X_{i,4}$; $X_{i+1,7} \leftarrow X_{i,6}$;
$X_{i+1,0} \leftarrow X_{i,7} \oplus (F_0(X_{i,6}) \boxplus SK_{4i+3})$;
$X_{i+1,2} \leftarrow X_{i,1} \boxplus (F_1(X_{i,0}) \oplus SK_{4i+2})$;
$X_{i+1,4} \leftarrow X_{i,3} \oplus (F_0(X_{i,2}) \boxplus SK_{4i+2})$;
$X_{i+1,6} \leftarrow X_{i,5} \boxplus (F_1(X_{i,4}) \oplus SK_{4i+1})$;
}
// Final Transformation
$C_0 \leftarrow X_{32,1} \boxplus WK_4$; $C_1 \leftarrow X_{32,2}$;
$C_2 \leftarrow X_{32,3} \oplus WK_5$; $C_3 \leftarrow X_{32,4}$;
$C_4 \leftarrow X_{32,5} \boxplus WK_6$; $C_5 \leftarrow X_{32,6}$;
$C_6 \leftarrow X_{32,7} \oplus WK_7$; $C_7 \leftarrow X_{32,0}$;
return C
}

2.2. Neural Distinguisher and Its Utilization for Key Recovery Attack

A neural distinguisher [7] is a trained deep learning model for distinguishing the real ciphertexts from the random ciphertexts. Let the plaintext pair be represented as (P, P'), the ciphertext pair as (C, C'), and the block cipher encryption function with a secret key be represented as E_K, where K is a secret key. The real ciphertext pair refers to a ciphertext pair $(C = E_K(P), C' = E_K(P'))$, corresponding to the plaintext pair $P \oplus P' = \alpha$. In the case of real ciphertext pairs, it is labeled as $label = 1$, whereas it is labeled as $label = 0$ in the case of random ciphertext pairs. The dataset composed of real and random ciphertext pairs can be expressed as follows:

$$\begin{cases} (C, C', label = 1), \text{if } P \oplus P' = \alpha \\ (C, C', label = 0), \text{if } P \oplus P' \neq \alpha \end{cases} \tag{2}$$

Then, the problem that the neural distinguisher has to solve from the ciphertext dataset is a binary classification problem of classifying whether the input ciphertext (C, C') has a label of 1 or 0. This can be thought of as learning multiple differentials of the output differences propagated from the input difference α, in the perspective of cryptanalysis, to distinguish between real ciphertext pairs and random ciphertext pairs. In A. Gohr's work, the strategy for choosing the input difference is that the input difference has a small hamming weight and has a differential trail with a high probability. In this regard, α was set 0x0040/0000, which is a three-round characteristic of nine-round differential characteristic for SPECK-32/64 [31].

The neural distinguisher's structure was composed of a residual network preceded by a single bit-sliced convolution and followed by a densely connected prediction head. The size of input channels was set to maintain the word size of target cipher (note that the word size of SPECK-32/64 is 16 bits). The dataset was composed 10^7 samples, and the last 10^6 samples were used for validation. The epochs for training was set 200, and optimization was performed against mean square error loss with an L2 weight regularization using the Adam optimizer. In addition, a cyclic learning rate scheduler was used. The detailed descriptions of the structure and training pipeline of the neural distinguisher are shown in Sections 4.2 and 4.3 in [7]. Following this training pipeline, A. Gohr could obtain five-, six-, seven-, and eight-

round neural distinguishers for SPECK-32/64. Table 1 shows the accuracy of the trained neural distinguishers. Here, the accuracy is calculated by comparing the classification results and labels. The classification is performed through the response of the neural distinguisher Z, and the input ciphertext pair is classified as the real case if $Z > 0.5$.

Table 1. Accuracy of A. Gohr's SPECK-32/64 neural distinguisher.

Rounds	Distinguisher	Accuracy	TPR [1]	TNR [2]
5	Neural Distinguisher	0.929	0.904	0.954
5	Differential Distinguisher	0.911	0.877	0.947
6	Neural Distinguisher	0.788	0.724	0.853
6	Differential Distinguisher	0.758	0.680	0.837
7	Neural Distinguisher	0.616	0.533	0.699
7	Differential Distinguisher	0.591	0.543	0.640
8	Neural Distinguisher	0.514	0.519	0.508
8	Differential Distinguisher	0.512	0.496	0.527

[1] True positive rate. [2] True negative rate.

Then, the response of the neural distinguisher trained in this manner is utilized to recover the subkey. As shown in Figure 2, the idea of key recovery is that when the neural distinguisher was trained for n-round, we prepare $(n+1)$-round ciphertexts, partially decrypt one round, denoted as D_K using the key K, and then check the response of the neural distinguisher against the partially decrypted ciphertexts. Let the $(n+1)$-round ciphertext pair be denoted as (C_i, C_i'), where $i = 0, \ldots, t-1$ and t is total number of ciphertext pairs. The response of the neural distinguisher for the i-th ciphertext pair is represented as Z_i^k, where k is the key which is used for partial decryption. It represents the likelihood $Pr(real|(D_k(C_i), D_k(C_i')))$ of each input data following the real ciphertext pair distribution. The input data are classified as the real ciphertext if $Z_i^k > 0.5$. This likelihood Z_i^k can be transformed the key-dependent score v_k to utilize from the recovery perspective through the following formula:

$$v_k := \sum_{i=1}^{t} \log_2(Z_i^k/(1-Z_i^k)) \tag{3}$$

where v_k is a summed result of logarithms of real vs. random likelihood ratio. Therefore, this will be the larger value when the larger response of the neural distinguisher is classified as real cases for each partial decryption. As a result, if the guessed key is the right key, more real cases will appear than the wrong key, so we can identify the right key by performing key ranking according to v_k.

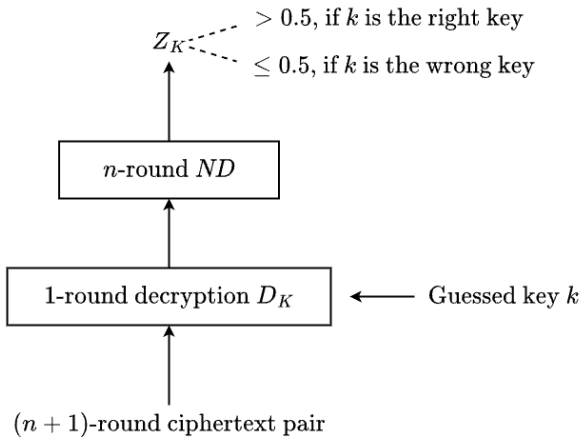

Figure 2. A $(n+1)$-round key guess using n-round neural distinguisher.

2.3. Existing Works Related to Neural Distinguisher

After A. Gohr's differential neural cryptanalysis, A. Benamira et al. [10] presented research results on the interpretation of the features learned by the neural distinguisher and the accuracy of the neural distinguisher at EUROCRYPT 2021. Here, it was experimentally confirmed that the features learned by A. Gohr's neural distinguisher could be interpreted as intermediate round truncated differential characteristics from a cryptanalysis perspective. Additionally, it was confirmed that the initial convolution operation in the neural distinguisher's internal layer converts the input ciphertext data into a form that can represent the difference. Specifically, when a 64-bit ciphertext pair (C, C') is expressed as word units $(C_l, C_r, C'l, C'r)$, through the initial convolution, the input ciphertext pair data are transformed into $(\Delta L, \Delta V, V_0, V_1)$, where $\Delta L = C_l \oplus C_r$, $\Delta V = V_0 \oplus V_1$, $V_0 = C_l \oplus C_r$, $V_1 = C'_l \oplus C'_r$.

In D. Pal et al. [29], a neural distinguisher study was conducted on the HIGHT cipher targeted in this paper. In this study, the seven-round characteristic 0x0080000000000000 of HIGHT differential characteristic, proposed by J. Yin et al. [26], was set as the input difference to generate the ciphertext pairs (C, C'). Additionally, the difference $C \oplus C'$ of the ciphertext pairs was concatenated to form a 192-bit dataset (i.e., $(C, C', C \oplus C')$) and used for training. As a result, an eight-round neural distinguisher with 98% accuracy was obtained, and a fourteen-round neural distinguisher was constructed by prepending a six-round differential characteristic (0x80008AC28A01A0 $\rightarrow$ 0x0080000000000000) with a probability of 2^{30}. The HIGHT neural distinguisher constructed in our study selects the input difference from the 13-round characteristic of [32], similar to the D. Pal's HIGHT neural distinguisher, but there are differences in the input difference setting value and the dataset format. In addition, our neural distinguisher works for 15 rounds (not 14 rounds) using the partial difference of 15-round ciphertext pairs.

3. HIGHT Neural Distinguishers

3.1. Approach for Making HIGHT Neural Distinguisher

In this study, we focused on designing efficiently to create and utilize the neural distinguisher. The main idea for constructing a neural distinguisher for this purpose is to create a neural distinguisher that depends on a partial key $k' \subset k$. As explained in Section 2.2, when utilizing a neural distinguisher from a key recovery perspective, the response Z_i^k of an n-round neural distinguisher is transformed into v_k, making it dependent on the $(n+1)$-round encryption key k. While finding round keys is more efficient than directly recovering the master key, the search space for keys increases exponentially as the size of the round keys grows, making this approach a significant burden. However, by using a divide-and-conquer method to first guess k' instead of k, and then guess the remaining key bits, the exponential complexity of key guessing can be improved.

To construct a neural distinguisher that is dependent on a partial key k', instead of using the entire ciphertext as input data for the neural distinguisher, we developed a new method to distinguish between real and random inputs by training on information based on partial ciphertexts. In detail, we constructed the input of the neural distinguisher using the difference for $(X_{i+1,6}, X_{i+1,5}, X_{i+1,2}, X_{i+1,1})$ in the output word of the i-th round HIGHT round function. Since the neural distinguisher trained on this dataset uses only partial information of the ciphertext, it does not care about any information on $(X_{i+1,7}, X_{i+1,4}, X_{i+1,3}, X_{i+1,0})$. In addition, it means that the neural distinguisher focuses only on the difference rather than the output distribution. In other words, our intention is to train the neural distinguisher to propagate the multiple truncated differential characteristic $(\Delta X_{i+1,6}, \Delta X_{i+1,5}, \Delta X_{i+1,2}, \Delta X_{i+1,1})$ from the input difference Δ_{in} in terms of the differential characteristic. Accordingly, we named the neural distinguisher constructed based on this approach as a truncated neural distinguisher. Figure 3 shows an overview of the proposed truncated neural distinguisher. The detailed descriptions and rationale of the proposed neural distinguisher are in the following subsection.

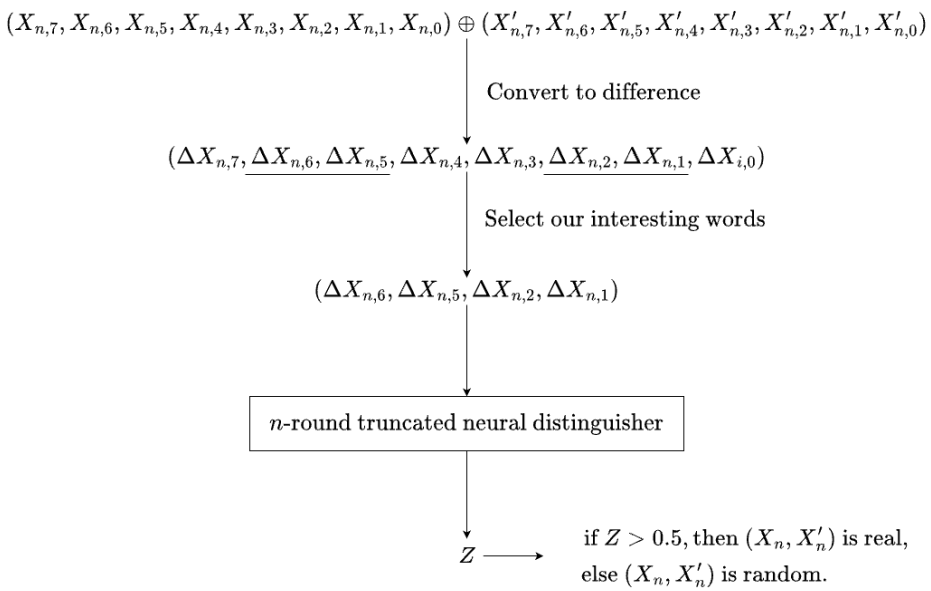

Figure 3. Truncated neural distinguisher for HIGHT.

3.2. Rationale of our Neural Distinguisher

The basic design rationales of the proposed neural distinguisher, including the input difference setting and neural network structure, are similar to A. Gohr's neural distinguisher design principles. The input difference was set to the 6-round characteristic 0x0000010000000000 of the 13-round differential characteristic (Table 2) proposed by E. Bagherzadeeh and Z. Ahmadian [32]. The input difference is propagated with a probability of 1/2, and has a Hamming weight of 1, which is consistent with the HIGHT neural distinguisher proposed by D. Pal et al. The neural network structure is composed of a residual network with a depth of 1, and it is designed to maintain the word unit (8-bit) of HIGHT internally. However, through the following analysis, the proposed neural distinguisher is designed to use only partial information of the ciphertext.

Table 2. Differential characteristics for 13-round HIGHT [32].

Rounds	Characteristic	$log_2 p$
0	0x01004483E20084F2	-
1	0x000083E20080F201	−3
2	0x0000E2C1804A0100	−9
3	0x0000C1184A010000	−6
4	0x000018C801000000	−6
5	0x0000C80100000000	−4
6	0x0000010000000000	−3
7	0x0001000000000000	−1
8	0x0100000000000082	−2
9	0x00000000009C8201	−3
10	0x000000039C7A0100	−8
11	0x00E803BC7A010000	−5
12	0xE800BCF801000002	−6
13	0x00B6F80100B002E8	−5
-	$\Sigma \log_2 p$	−61

The design of the proposed neural distinguisher to operate in a truncated manner was based on the observation of input conversion in A. Benamira et al.'s neural distinguisher construction method, which was described in Section 2.3. A. Benamira et al. transformed input data to $(\Delta L, \Delta V, V_0, V_1)$ from the ciphertext pairs (C_l, C_r, C'_l, C'_r) to improve the accuracy of the neural distinguisher. When each element of the transformed form corresponds

to the internal state of SPECK-32/64, it is related to the pre- and post-information of the non-linear operation, namely modular addition. Based on this, we hypothesized that the neural distinguisher explores the pre- and post-difference characteristics of modular addition to learn the characteristics of non-linear operations during training. We performed a property analysis of modular addition in HIGHT based on this conjecture.

Considering the i-round HIGHT round function, we can see that values prior to word rotation are two types of operation results. These two types have different characteristics from the XOR difference perspective of a ciphertext pair. To explain this, let us denote the left and right inputs of the internal operation as a and b, respectively, and the round key as sk. Also, let the Type-1 operation be denoted as $G : \{a, b, sk\} \rightarrow \{c, d\}$ and Type-2 operation as $H : \{a, b, sk\} \rightarrow \{c, d\}$. If we denote the function G as H, then they can be defined as follows:

$$\begin{aligned} G(a, b, sk) &:= (c = (F_0(b) \boxplus sk) \oplus a, d = a), \\ H(a, b, sk) &:= (c = (F_1(b) \oplus sk) \boxplus a, d = a). \end{aligned} \tag{4}$$

If the XOR differences between the left and right words are α and β, respectively (i.e., $a \oplus a' = \alpha$ and $b \oplus b' = \beta$), since all operations except modular addition are linear operations with respect to XOR differences, then the differences are propagated as shown in Figure 4. The output XOR difference of the right word of G and H maintains the input difference as β. However, the left difference has a different property. First, in the case of G, the modular addition is performed with respect to the output of F_0 and sk, so the difference is probabilistically propagated. Here, if we denote the output difference of modular addition as γ, the output difference of the left word of G is $\alpha \oplus \gamma$. On the other hand, in the case of H, the round key addition is performed through XOR, so $F_0(\beta)$ is maintained with a probability of 1 (note that both F_0 and F_1 are linear operations from the XOR difference perspective because they are composed of rotation and XOR operations). Subsequently, the difference is probabilistically propagated through modular addition. Here, if we denote the output difference of modular addition as θ, the output difference of the left word of H is θ. The output difference of the left word for both G and H ($\alpha \oplus \gamma$ and θ, respectively) is probabilistically propagated, but there is a different property on these output differences in whether they depend on the round key sk. While γ in $\alpha \oplus \gamma$ varies depending on the value of sk, θ is independent of the value of sk. In other words, the output difference of G is affected by the round key, but that of H is not.

From the perspective of the neural distinguisher learning the characteristics of G and H, consider given data of (Δc, Δd) for neural distinguisher training. Based on the conjecture we built earlier, that the neural distinguisher learns the pre- and post-relationship of modular addition using (Δc, Δd), it is easier to learn H than G. For G, it consists of ($\Delta c = \alpha \oplus \gamma$, $\Delta\beta$). Here, the right word difference $\Delta\beta$ can be easily derived, as it is linearly related to one of the inputs of the modular addition, $F_0(\beta)$. However, the left word difference $\alpha \oplus \gamma$ that is masked from the value of γ, which itself varies probabilistically with the value of sk, is known only to the neural distinguisher. On the other hand, for H, it consists of ($\Delta c = \theta$, $\Delta d = \beta$). Here, β can be easily derived from the input of the modular addition, $F_1(\beta)$, regardless of the value of sk. Furthermore, the left word difference θ is the output difference of the modular addition, allowing the neural distinguisher to focus on learning the non-linear operation. Based on these analysis results, we constructed two types of neural distinguishers: a Type-1 neural distinguisher that learns only the output of G, and a Type-2 neural distinguisher that learns only the output of H, denoted as ND_{T1} and ND_{T2}, respectively. For ND_{T1} and ND_{T2}, the dataset was sorted to the state before the word rotation of the round function to represent the output form of G and H, respectively. Accordingly, the dataset for ND_{T1} consisted of $(C_{n,0}, C_{n,7}, C_{n,4}, C_{n,3})$, and the dataset for ND_{T2} consisted of $(C_{n,6}, C_{n,5}, C_{n,2}, C_{n,1})$. We present a comparison of the two types of neural distinguishers in the following section.

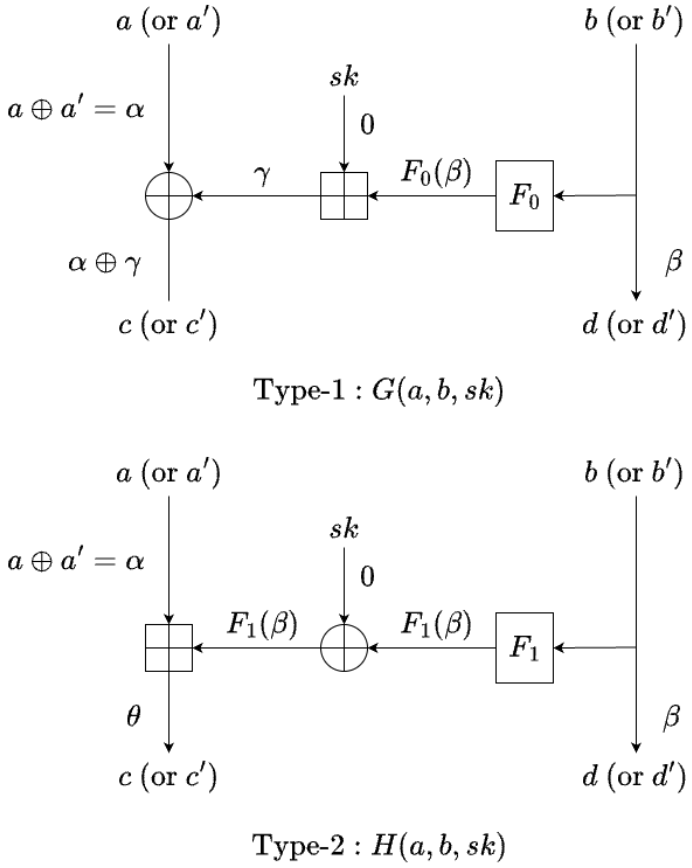

Figure 4. Type-1 and Type-2 operation in HIGHT round function.

3.3. Training Results

The training pipeline for ND_{T1} and ND_{T2} was constructed as follows. We generated 10^7 pairs of HIGHT ciphertexts (C, C'), satisfying the plaintext pair (P, P') with input difference Δ_{in} =0x0000010000000000. The generated ciphertexts were preprocessed to transform them into the difference form corresponding to the input of each neural distinguisher. From the generated dataset, 10^6 samples were used for validation, while the remaining samples were allocated for training. The batch size and optimization were configured the same as A. Gohr's training pipeline (batch size: 5000, loss: MSE with L2 weights regularization, optimization: Adam, learning rate: cyclic learning rate scheduler). The training was performed for a total of 200 epochs.

The model was implemented using Tensorflow and the training environment was Intel Xeon Gold 6230R (26 Core/2.1 GHz) with a Geforce RTX 3090 and 64 GB RAM. Each neural distinguisher training took an average of 11,634.5 s (approximately 3.2 h) in our environment. Table 3 shows the comparison of the training performance of the neural distinguishers. As a result of the training, while the neural distinguisher was able to distinguish the input ciphertext with an accuracy of over 90% from the sixth to the eighth rounds, a significant drop in accuracy was observed in the ninth round, with ND_{T2} showing approximately 82% accuracy and ND_{T1} showing about 59%. Furthermore, from the 10th round onwards, except for ND_{T1}, which had an accuracy of about 57%, the results dropped to around 50%, which is essentially meaningless when considering the binary classification problem that the neural distinguisher is designed to solve. When comparing the accuracy of ND_{T1} and ND_{T2} for rounds 6 to 9, where both types of neural distinguishers showed meaningful performance, ND_{T2} consistently exhibited higher accuracy, except for the seventh round. In particular, in the ninth round, the accuracy difference between ND_{T1}

and ND_{T2} was significant, with a gap of 23%. This supports our earlier conjecture that H is easier to learn than G.

Based on this analysis of the training results, we chose the nine-round ND_{T2} as the most optimal option for designing a key recovery attack. The reason is that in key recovery attacks, which require repeated queries to the neural distinguisher, the closer the accuracy is to 50%, the less reliable the query results become. Therefore, selecting a neural distinguisher with significantly higher accuracy allows for a more efficient key recovery attack. Detailed explanations of the key recovery attack will be covered in the next section.

Table 3. Accuracy of various types of HIGHT neural distinguisher.

Round	Neural Distinguishers	Accuracy
6	ND_{T2}	100.00%
6	ND_{T1}	93.47%
7	ND_{T2}	93.00%
7	ND_{T1}	99.90%
8	ND_{T2}	93.43%
8	ND_{T1}	92.95%
9	ND_{T2}	82.18%
9	ND_{T1}	58.93%
10	ND_{T2}	50.07%
10	ND_{T1}	57.07%
11	ND_{T2}	50.05%
11	ND_{T1}	50.08%

4. Key Recovery Attack Based on Truncated Neural Distinguisher

4.1. Overview of the Proposed Attack

In this section, we propose a HIGHT key recovery attack using the previously constructed nine-round ND_{T2}. In the attack described in this paper, HIGHT is reduced to 15 rounds, assuming the application of initial and final transformations. The goal is to recover $WK_4(MK_0), WK_6(MK_2)$ used in the final transformation. Figure 5 shows an overview of the proposed attack. We expanded the nine-round truncated neural distinguisher to a fifteen-round truncated neural distinguisher by prepending a six-round differential characteristic ($\Delta_{initial} \rightarrow \Delta_{in}$) with a probability of 2^{-31} (here, Δ_{inital} = 0x01004483E20084F2 and Δ_{in} = 0x0000010000000000). The process of the key recovery attack based on the expanded truncated neural distinguisher is as follows:

1. Prepare a set of chosen plaintext pairs $P_i = (p, p')$ satisfying $p \oplus p' = \Delta_{initial}$, where $i = 0, 1, \ldots, c-1$ (this maintains $\Delta_{initial}$ with a probability of 2^{-8} after the initial transformation is performed).
2. Obtain a set of corresponding ciphertext pairs $C_i = (c, c')$ for the plaintext pairs P_i based on the chosen plaintext attack scenario, where $i = 0, 1, \ldots, c-1$ (with a probability of 2^{-31} for satisfying the differential characteristic $\Delta_{initial} \rightarrow \Delta_{in}$ up to six rounds).
3. For candidate keys WK_4 and WK_6, perform partial decryption ($\boxminus$) on (C_5^i, C_4^i, C_1^i, C_0^i) of ciphertext pairs C to obtain the differences of partial plaintext pairs (ΔX_6^i, ΔX_5^i, ΔX_2^i, ΔX_1^i) (note that we increase the index by 1 to account for the word rotation in the final transformation).
4. Obtain the response of the 15-round truncated neural distinguisher $Z^i_{WK_4,WK_6}$ for the differences of partial plaintext pairs $(\Delta X_6^i, \Delta X_5^i, \Delta X_2^i, \Delta X_1^i)$.
5. Convert the responses of the neural distinguisher to key scores by calculating $V_{WK_4,WK_6} = \sum_{i=0}^{c-1} Z^i_{WK_4,WK_6} / (1 - Z^i_{WK_4,WK_6})$.
6. Return the WK_4 and WK_6 corresponding to the highest key score.

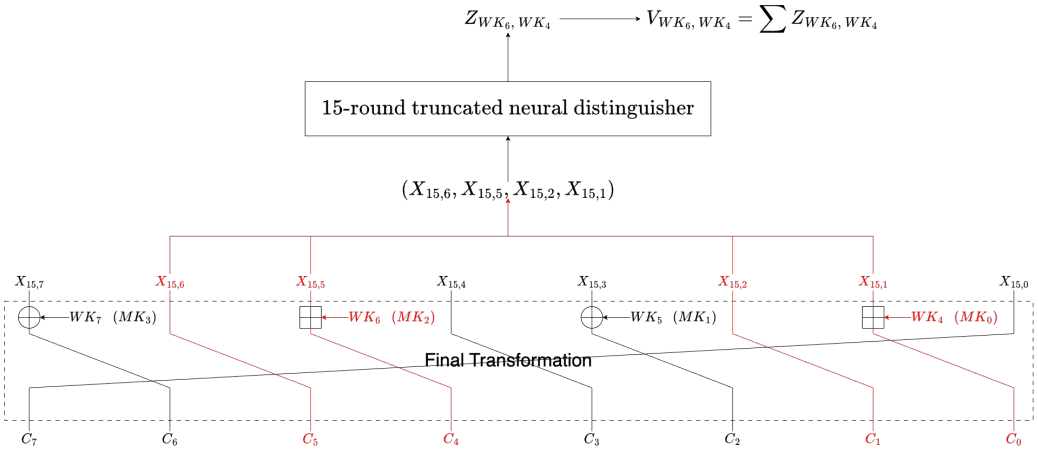

Figure 5. Partial whitening key guess of final transformation on 15-round HIGHT.

4.2. Experiments and Evaluation

The experiments for the proposed attack were performed by assuming that $\Delta_{inital} \rightarrow \Delta_{in}$ holds with probability $2^{-8} \times 2^{-31} = 2^{-39}$ for six rounds after the initial transformation, and recovering WK_4 and WK_6 of nine-round HIGHT with the final transformation applied. The experiments were conducted on the same Intel Xeon Gold 6230R (26 Core/2.1 GHz) with Geforce RTX 3090 environment as the training environment of the neural distinguisher, and were performed approximately 100 times. The number of selected plaintext pairs used in the attack was 8 (i.e., $c = 2^3$). As a result, each attack took an average of 9 s (911.5 s for 100 trials). The success of the attack was measured by how similar the returned keys were to the real value of WK_4 and WK_6. The results showed that WK_4 and WK_6 had different recovery performances. Specifically, WK_6 was accurately recovered in about 39 out of 100 trials, and for the remaining cases, it was recovered with a Hamming distance of 2 from the real key. However, WK_4 was not accurately recovered in all of the trials, and a result with a Hamming distance of approximately 4 from the actual key was obtained. These results indicate that, although it is difficult to accurately recover against WK_4, it can recover WK_6 and effective key candidate reduction can be achieved with a small amount of data.

4.3. Complexity of the Proposed Attack

As the partial whitening keys could be recovered using eight chosen plaintext pairs in our experiment, the data complexity of the proposed attack for 15-round HIGHT is $2 \times 2^3 \times 2^{39} = 2^{43}$, considering the prepending initial transformation and six-round differential characteristic. From a time complexity perspective, the proposed attack needs to perform key guessing for a total of 2^{16} cases of WK_4 and WK_6, each with 2^8 cases. Additionally, partial decryption ($\boxminus$) and neural distinguisher queries must be performed for each ciphertext pair. Therefore, the time complexity is $2^{16} \times 2^{42} = 2^{58}$ for partial decryption and neural distinguisher prediction times.

To compare the complexity of the proposed attack with that of traditional cryptanalysis methodologies, a brief explanation of previously known HIGHT cryptanalysis results is provided. The most representative attacks on full-round HIGHT are O. Özen et al. [33]'s related-key impossible differential attack and D. Hong et al. [28]'s biclique attack, both of which have theoretical time complexities exceeding 2^{125}, making them impractical. Since the assumptions underlying the previously explained attacks differ from the proposed attack in this paper, direct comparisons are difficult. Therefore, we performed a comparison based on the complexity analysis of a traditional differential attack using the best differential characteristic presented in [32]. Following the traditional differential attack methodology in [34], constructing an attack by applying the 13-round differential characteristic with a probability of 2^{-61} (Table 2), the data complexity is $2 \times 2^3 \times 2^{61} = 2^{65}$, and the time complexity is the sum of the partial decryption time for WK_6 and WK_4 guessing,

which is $2^8 \times 2^{64} + 2^8 \times 2^{64} = 2^{73}$ partial decryption times. Compared to the traditional differential attack, the proposed attack is capable of targeting more rounds and demonstrates significantly better performance in terms of both data complexity and time complexity.

5. Discussion and Future Directions

In the previous section, we demonstrated that partial key recovery is possible based on the nine-round ND_{T2} trained by targeting the H of HIGHT. In particular, the proposed attack was able to analyze more rounds more efficiently compared to traditional differential attacks. This suggests that by learning the partial output when constructing the neural distinguisher, more efficient differential-neural cryptanalysis can be achieved in the context of key recovery attacks. Furthermore, it overcomes the limitation where traditional differential cryptanalysis struggled to analyze beyond 13 rounds of HIGHT. However, the results of the proposed attack in this paper do not imply that the overall security of HIGHT is compromised. The version of HIGHT targeted in this paper can recover partial keys when reduced to 15 rounds, but the standardized HIGHT, which is used in resource-constrained environments such as IoT devices, operates with 32 rounds. To extend the key recovery attack to more rounds, constructing a truncated neural distinguisher with higher accuracy is essential. However, the truncated neural distinguisher with meaningful accuracy could only be constructed up to 10 rounds, and it did not show meaningful accuracy beyond 11 rounds (see Table 3). In the proposed attack we performed, the neural distinguisher was used by prepending a six-round differential characteristic, so the resistance of HIGHT against the proposed attack can be considered up to 16 rounds. However, if a key recovery attack is extended with a 16-round distinguisher, it is unlikely to be more efficient than a traditional differential attack, due to its low accuracy. Therefore, the practical resistance starts from 16 rounds. Consequently, full-round HIGHT currently holds sufficient security against differential-neural cryptanalysis, and the devices using HIGHT are considered secure. However, the results presented in this paper suggest that differential-neural cryptanalysis based on truncated neural distinguishers could provide improved analysis for HIGHT and similar ciphers that have been analyzed through traditional differential cryptanalysis, especially generalized Feistel-based block ciphers. For this reason, further research is needed to extend truncated neural distinguisher studies to other block ciphers with similar structures to HIGHT.

6. Conclusions

In this paper, we conducted differential-neural cryptanalysis on HIGHT, an international standard lightweight block cipher. Through analysis of the internal operation properties of the HIGHT round function, we proposed a new concept of neural distinguisher, named the truncated neural distinguisher, that can distinguish ciphertext using only partial words of the plaintext. From the perspective of utilization in key recovery attacks, the truncated neural distinguisher can perform partial decryption for specific words and allow for efficient key recovery by dividing and conquering the key search. In the experiment, the truncated neural distinguisher was built for a nine-round HIGHT. Using this truncated neural distinguisher, we performed partial whitening key recovery for a 15-round HIGHT by prepending a 6-round differential characteristic. The proposed key recovery attack showed significantly improved data and time complexity compared to traditional differential key recovery attacks. We expect that the proposed truncated neural distinguisher can be extended to various generalized Feistel-based block ciphers and plan to conduct future research in this regard.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by Ministry of Education (No. RS-2024-00462291).

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis. (IJCV)* **2015**, *115*, 211–252. [CrossRef]
2. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; IEEE Computer Society: Piscataway, NJ, USA, 2018; pp. 7132–7141. [CrossRef]
3. Young, T.; Hazarika, D.; Poria, S.; Cambria, E. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Comput. Intell. Mag.* **2018**, *13*, 55–75. [CrossRef]
4. Lerman, L.; Bontempi, G.; Markowitch, O. *Side Channel Attack: An Approach Based on Machine Learning*; Center for Advanced Security Research Darmstadt: Darmstadt, Germany, 2011; pp. 29–41.
5. Mushtaq, M.; Akram, A.; Bhatti, M.K.; Chaudhry, M.; Yousaf, M.; Farooq, U.; Lapotre, V.; Gogniat, G. Machine Learning For Security: The Case of Side-Channel Attack Detection at Run-time. In Proceedings of the 25th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2018, Bordeaux, France, 9–12 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 485–488. [CrossRef]
6. Wei, L.; Luo, B.; Li, Y.; Liu, Y.; Xu, Q. I Know What You See: Power Side-Channel Attack on Convolutional Neural Network Accelerators. In Proceedings of the 34th Annual Computer Security Applications Conference, ACSAC 2018, San Juan, PR, USA, 3–7 December 2018; ACM: New York, NY, USA, 2018; pp. 393–406. [CrossRef]
7. Gohr, A. Improving attacks on round-reduced speck32/64 using deep learning. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019*; Proceedings, Part II 39; Springer: Berlin/Heidelberg, Germany, 2019; pp. 150–179.
8. Biham, E.; Shamir, A. Differential cryptanalysis of DES-like cryptosystems. *J. Cryptol.* **1991**, *4*, 3–72. [CrossRef]
9. Hou, Z.; Ren, J.; Chen, S. Improve neural distinguishers of simon and speck. *Secur. Commun. Netw.* **2021**, *2021*, 1–11. [CrossRef]
10. Benamira, A.; Gerault, D.; Peyrin, T.; Tan, Q.Q. A deeper look at machine learning-based cryptanalysis. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, 17–21 October 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 805–835.
11. Gohr, A.; Leander, G.; Neumann, P. An Assessment of Differential-Neural Distinguishers. *Cryptol. ePrint Arch.* **2022**, 1–42. Available online: https://eprint.iacr.org/2022/1521 (accessed on 19 August 2024).
12. Liu, J.; Ren, J.; Chen, S.; Li, M. Improved neural distinguishers with multi-round and multi-splicing construction. *J. Inf. Secur. Appl.* **2023**, *74*, 103461. [CrossRef]
13. Lu, J.; Liu, G.; Liu, Y.; Sun, B.; Li, C.; Liu, L. Improved Neural Distinguishers with (Related-key) Differentials: Applications in SIMON and SIMECK. *arXiv* **2022**, arXiv:2201.03767.
14. Seok, B.; Chang, D.; Lee, C. A novel approach to construct a good dataset for differential-neural cryptanalysis. *IEEE Trans. Dependable Secur. Comput.* 2024, *early access*. [CrossRef]
15. Bao, Z.; Guo, J.; Liu, M.; Ma, L.; Tu, Y. Enhancing differential-neural cryptanalysis. In *Proceedings of the Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, 5–9 December 2022*; Proceedings, Part I; Springer: Berlin/Heidelberg, Germany, 2023; pp. 318–347.
16. Zhang, L.; Wang, Z.; Wang, B. Improving Differential-Neural Cryptanalysis. *IACR Commun. Cryptol.* **2024**, *1*, 1–28. [CrossRef]
17. Baksi, A.; Baksi, A. Machine learning-assisted differential distinguishers for lightweight ciphers. In *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*; Springer: Singapore, 2022; pp. 141–162.
18. Zahednejad, B.; Lyu, L. An improved integral distinguisher scheme based on neural networks. *Int. J. Intell. Syst.* **2022**, *37*, 7584–7613. [CrossRef]
19. Sun, T.; Shen, D.; Long, S.; Deng, Q.; Wang, S. Neural Distinguishers on TinyJAMBU-128 and GIFT-64. In *Proceedings of the Neural Information Processing: 29th International Conference, ICONIP 2022, Virtual Event, 22–26 November 2022*; Proceedings, Part V; Springer: Berlin/Heidelberg, Germany, 2023; pp. 419–431.
20. Bogdanov, A.; Knudsen, L.R.; Leander, G.; Paar, C.; Poschmann, A.; Robshaw, M.J.; Seurin, Y.; Vikkelsoe, C. PRESENT: An ultra-lightweight block cipher. In *Proceedings of the Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, 10–13 September 2007*; Proceedings 9; Springer: Berlin/Heidelberg, Germany, 2007; pp. 450–466.
21. Hong, D.; Lee, J.K.; Kim, D.C.; Kwon, D.; Ryu, K.H.; Lee, D.G. LEA: A 128-bit block cipher for fast encryption on common processors. In *Proceedings of the Information Security Applications: 14th International Workshop, WISA 2013, Jeju Island, Republic of Korea, 19–21 August 2013*; Revised Selected Papers 14; Springer: Berlin/Heidelberg, Germany, 2014; pp. 3–27.
22. Hong, D.; Sung, J.; Hong, S.; Lim, J.; Lee, S.; Koo, B.S.; Lee, C.; Chang, D.; Lee, J.; Jeong, K.; et al. HIGHT: A new block cipher suitable for low-resource device. In *Proceedings of the Cryptographic Hardware and Embedded Systems-CHES 2006: 8th International Workshop, Yokohama, Japan, 10–13 October 2006*; Proceedings 8; Springer: Berlin/Heidelberg, Germany, 2006; pp. 46–59.
23. Shirai, T.; Shibutani, K.; Akishita, T.; Moriai, S.; Iwata, T. *The 128-Bit Blockcipher CLEFIA (Extended Abstract). FSE 2007. LNCS, Vol. 4593*; Springer: Heidelberg, Germany, 2007.
24. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. The SIMON and SPECK lightweight block ciphers. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, 8–12 June 2015; pp. 1–6.
25. Dai, W. Crypto++ Library. Version 8.7.0. Available online: https://www.cryptopp.com (accessed on 19 August 2024).

26. Yin, J.; Ma, C.; Lyu, L.; Song, J.; Zeng, G.; Ma, C.; Wei, F. Improved cryptanalysis of an ISO standard lightweight block cipher with refined MILP modelling. In *Proceedings of the Information Security and Cryptology: 13th International Conference, Inscrypt 2017, Xi'an, China, 3–5 November 2017*; Revised Selected Papers 13; Springer: Berlin/Heidelberg, Germany, 2018; pp. 404–426.
27. Koo, B.; Hong, D.; Kwon, D. Related-key attack on the full HIGHT. In *Proceedings of the Information Security and Cryptology-ICISC 2010: 13th International Conference, Seoul, Republic of Korea, 1–3 December 2010*; Revised Selected Papers 13; Springer: Berlin/Heidelberg, Germany, 2011; pp. 49–67.
28. Hong, D.; Koo, B.; Kwon, D. Biclique attack on the full HIGHT. In *Proceedings of the Information Security and Cryptology-ICISC 2011: 14th International Conference, Seoul, Republic of Korea, 30 November–2 December 2011*; Revised Selected Papers 14; Springer: Berlin/Heidelberg, Germany, 2012; pp. 365–374.
29. Pal, D.; Mandal, U.; Chaudhury, M.; Das, A.; Chowdhury, D.R. A Deep Neural Differential Distinguisher for ARX based Block Cipher. *Cryptol. ePrint Arch.* **2022**, 1–26. Available online: https://eprint.iacr.org/2022/1195 (accessed on 19 August 2024).
30. Bogdanov, A.; Shibutani, K. Generalized Feistel networks revisited. *Des. Codes Cryptogr.* **2013**, *66*, 75–97. [CrossRef]
31. Abed, F.; List, E.; Lucks, S.; Wenzel, J. Differential cryptanalysis of round-reduced Simon and Speck. In *Proceedings of the Fast Software Encryption: 21st International Workshop, FSE 2014, London, UK, 3–5 March 2014*; Revised Selected Papers 21; Springer: Berlin/Heidelberg, Germany, 2015; pp. 525–545.
32. Bagherzadeh, E.; Ahmadian, Z. MILP-based automatic differential search for LEA and HIGHT block ciphers. *IET Inf. Secur.* **2020**, *14*, 595–603. [CrossRef]
33. Özen, O.; Varıcı, K.; Tezcan, C.; Kocair, Ç. Lightweight block ciphers revisited: Cryptanalysis of reduced round PRESENT and HIGHT. In *Proceedings of the Information Security and Privacy: 14th Australasian Conference, ACISP 2009, Brisbane, Australia, 1–3 July 2009*; Proceedings 14; Springer: Berlin/Heidelberg, Germany, 2009; pp. 90–107.
34. Lai, X.; Massey, J.L.; Murphy, S. Markov ciphers and differential cryptanalysis. In *Proceedings of the Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques, Brighton, UK, 8–11 April 1991*; Proceedings 10; Springer: Berlin/Heidelberg, Germany, 1991; pp. 17–38.