Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Cryptanalysis

Tarun Yadav(✉) and Manoj Kumar

Scientific Analysis Group, DRDO, Metcalfe House Complex, Delhi 110 054, India
{tarunyadav,manojkumar}@sag.drdo.in

Abstract. The differential attack is a basic cryptanalytic technique for block ciphers. Application of machine learning shows promising results for the differential cryptanalysis. In this paper, we present a new technique to extend the classical differential distinguisher using machine learning (ML). We use r-round classical differential distinguisher to build an s-round ML based differential distinguisher. This s-round ML distinguisher is used to construct an $(r+s)$-round differential-ML distinguisher with the reduced data complexity. We demonstrate this technique on the lightweight block ciphers SPECK32, SIMON32, and GIFT64 by constructing the differential-ML distinguishers. The data complexities of distinguishers for 9-round SPECK32, 12-round SIMON32, and 8-round GIFT64 are reduced from 2^{30} to 2^{20}, 2^{34} to 2^{22}, and 2^{38} to 2^{20} respectively. Moreover, the differential-ML distinguisher for SIMON32 is the first 12-round distinguisher with the data complexity less than 2^{32}.

Keywords: Block cipher · Differential cryptanalysis · Machine learning

1 Introduction

Cryptanalysis of block ciphers witnessed the remarkable progress after the proposal of differential attack on DES by Biham and Shamir [8] in 1990. The differential attack is a basic and widely used cryptanalytic approach against the block ciphers. This attack is generalised and combined with other cryptanalytic techniques to reduce the attack complexity. High probability differential characteristics are the first and foremost requirement for the attack to succeed. In 1994, M. Matsui proposed a method based on the branch-and-bound technique [17] to search the high probability differential characteristics. In 2011, Mouha et al. proposed a new technique using mixed integer linear programming (MILP) to search the differential characteristics [18]. The method based on MILP uses optimization problem solvers to construct high probability differential characteristics. Most of the block ciphers follow the Shannon's principles [14] and wide trail design strategy [11] to thwart the differential attack. In practice, we need

P. Longa and C. Ràfols (Eds.): LATINCRYPT 2021, LNCS 12912, pp. 191–212, 2021.
https://doi.org/10.1007/978-3-030-88238-9_10

a differential with the probability greater than 2^{-n} to distinguish r rounds of an n-bit block cipher from the random data. Any r-round characteristic with a probability less than 2^{-n} cannot be used to mount the differential attack on r or more rounds of a block cipher. A differential characteristic is useful till it requires less data than the available limit i.e. 2^n pairs. The motivation of this paper is to find a technique which can be used to extend the classical differential characteristics without (much) increasing the data complexity. Machine learning based differential cryptanalysis approach works well to solve this problem.

Machine learning techniques are used to determine the meticulous relations in the data. Since such relations define the security strength of the cipher, identification of these relations plays an important role. In cryptanalysis domain, the machine learning techniques are explored very recently to mount the key recovery attack using differential cryptanalysis [12].

In this paper, we combine the classical and machine learning techniques to design an ML based generic extension for any classical differential distinguisher. This approach provides the better results with (much) lower data complexity. We extend an r-round high probability classical differential distinguisher with an s-round ML based differential distinguisher. The extended distinguisher is used to distinguish the $(r + s)$ rounds of a block cipher using less data. With this extension, the hybrid distinguisher outperforms both the classical and ML based distinguisher. We call this hybrid distinguisher a *differential-ML distinguisher*. This technique is experimented on three different types of lightweight block ciphers SPECK32 [4], SIMON32 [4], and GIFT64 [3] and better results are obtained with very high accuracy.

The remaining part of the paper is organised as follows. In Sect. 2, we compare our technique with the previous work. In Sect. 3, we provide a brief description of the lightweight block ciphers SIMON32, SPECK32 and GIFT64. We discuss the classical differential distinguisher and machine learning based differential distinguisher in Sect. 4 and describe the existing work on differential distinguishers using machine learning. In Sect. 5, we propose a novel technique to construct the differential-ML distinguisher. We demonstrate our technique on SPECK32, SIMON32, and GIFT64 block ciphers and present the results in Sect. 6. The paper is concluded in Sect. 7.

Notations: We have used the following notations in this paper:

- Δ_r :Output difference after r rounds
- 2^{-p_r} :Probability of r-round differential characteristic
- $D_{x\cdots y}$:Distinguisher for $(y - x + 1)$ rounds; x and y are round indices
- $D^{CD}_{x\cdots y}$:Classical differential distinguisher
- $D^{ML}_{x\cdots y}$:Machine learning based differential distinguisher
- $D^{CD\rightarrow ML}_{x\cdots y}$:Differential-ML distinguisher

Conventions: Throughout the paper, we refer an r-round differential distinguisher with the single input and single output difference as a classical differential distinguisher $D^{CD}_{1\cdots r}$.

2 Comparison with the Previous Work

A. Gohr [12] used machine learning techniques and proposed the idea of learning the differences to mount a key recovery attack. He presented a technique to construct the ML based differential distinguisher and used it for the key recovery attack on SPECK32. Gohr compared this technique with the classical differential attack and showed that complexity of the key recovery attack is reduced by using the ML distinguisher. Baksi et al. [2] also used the same approach to design the ML distinguisher for GIMLI cipher and GIMLI hash [5]. Various ML architectures are compared in [2] and it is claimed that ML distinguisher for GIMLI outperforms the classical differential distinguisher.

A. Gohr presented the 11-round key recovery attack on SPECK32. In this attack, 7-round ML based distinguisher is used and it is extended to 9 rounds by pre-pending a 2-round high probability differential distinguisher. In Gohr's approach, the accuracy and the data complexity of the 9-round extended distinguisher is not discussed explicitly. Although, the accuracy of extended distinguisher is quite low, yet it is used in the key recovery with various cipher specific optimizations. In this paper, we present a new technique to extend r-round classical differential distinguisher using an s-round ML distinguisher. Now, the extended distinguisher works as the $(r + s)$ rounds differential-ML distinguisher. The proposed technique ensures that the accuracy of differential-ML distinguisher is high and comparable to the classical differential distinguisher. We experimentally show that there is an exponential reduction in the data complexity of the $(r + s)$-round distinguisher by using the proposed differential-ML distinguisher.

3 Block Ciphers: SPECK32, SIMON32, and GIFT64

SPECK and SIMON are two families of the block ciphers proposed by Beaulieu et al. [4] in 2013. These block ciphers are designed to provide the high performance across a range of devices. There are 10 versions of each cipher based on the block and key size combinations which makes them suitable for a wide range of applications. We discuss the encryption algorithm for 32-bit block size and 64-bit key variants of each block cipher. We omit the key expansion algorithm and original paper [4] can be referred for more details.

GIFT is designed by improving the bit permutation of the lightweight block cipher PRESENT. Based on the input plaintext block size, there are two versions of GIFT namely GIFT64 and GIFT128. In each version, the 128-bit key is used to encrypt the input plaintext. A brief description of SPECK32, SIMON32, and GIFT64 block ciphers is provided in the following subsections.

3.1 Description of SPECK32

SPECK32 is a block cipher with 32-bit block size and 64-bit key size. There are total 22 rounds in SPECK32. It is based on the Feistel network and can

be represented by the composition of two Feistel maps. Its encryption algorithm divides the 32-bit input into the two 16-bit words (L_r, R_r) and the key expansion algorithm extracts the 16-bit round subkeys (RK_r) for each round. The round function comprises of addition modulo 2^{16}, bitwise XOR, left and right circular shift operations as described in Algorithm 1.

Algorithm 1: Encryption Algorithm of SPECK32

Input: $P = (L_0||R_0)$ and $RK_r (0 \leq r \leq 21)$
Output: $C = (L_{22}||R_{22})$
for *r=0 to 21* **do**
  $L_{r+1} = ((L_r \ggg 7) + R_r) \oplus RK_r$
  $R_{r+1} = L_{r+1} \oplus (R_r \lll 2)$
end

3.2 Description of SIMON32

SIMON32 is a block cipher with 32-bit block size and 64-bit key size. There are total 32 rounds in SIMON32 and it is also based on the Feistel network. Its encryption algorithm divides the 32-bit input into two 16-bit words (L_r, R_r). The key expansion algorithm expands the 64-bit master key to provide the 16-bit round subkeys (RK_r) for each round. It applies a round function consisting the bitwise XOR, bitwise AND, and left circular shift operations on the left 16-bit words in each round. The encryption algorithm of SIMON32 is described in Algorithm 2.

Algorithm 2: Encryption Algorithm of SIMON32

Input: $P = (L_0||R_0)$ and $RK_r (0 \leq r \leq 31)$
Output: $C = (L_{32}||R_{32})$
for *r=0 to 31* **do**
  $L_{r+1} = (L_r \lll 1 \ \& \ L_r \lll 8) \oplus (L_r \lll 2) \oplus R_r \oplus RK_r$
  $R_{r+1} = L_r$
end

3.3 Description of GIFT64

GIFT64 encrypts a 64-bit plaintext block using the 128-bit key and generates a 64-bit ciphertext block [3]. There are total 28 rounds in GIFT64. In each round, S-box, bit permutation, round subkeys and constant additions are applied through the round function. The key expansion algorithm extracts the 32-bit subkeys (RK_r) from the 128-bit key. Its encryption algorithm uses a 4-bit S-box S (Table 1), bit permutation P_{64} (Table 2), 6-bit round constants C_r (Table 3) and 32-bit round subkeys RK_r as described in Algorithm 3.

Algorithm 3: Encryption Algorithm of GIFT64

Input: $P(= X_0) = (x_{63}, x_{62}, \cdots, x_0)$ and $RK_r = (U, V)(0 \leq r \leq 27)$
Output: $C = X_{28}$
for *r=0 to 27* **do**
  for *j=0 to 15* **do**
    $(y'_{3+4*j}, y'_{2+4*j}, y'_{1+4*j}, y'_{0+4*j}) = S(x_{3+4*j}, x_{2+4*j}, x_{1+4*j}, x_{0+4*j})$
  end
  $(y_{63}, y_{62}, \cdots, y_0) = P_{64}(y'_{63}, y'_{62}, \cdots, y'_0)$
  for *k=0 to 5* **do**
    $y_{3*(k+1)+k} = c_r \oplus y_{3*(k+1)+k}$
  end
  for *l=0 to 15* **do**
    $y_{4l+1} = y_{4l+1} \oplus u_l$
    $y_{4l} = y_{4l} \oplus v_l$
  end
  $X_{r+1} = (y_{63}, y_{62}, \cdots, y_0) \oplus (1 \ll 63)$
end

S-box: The 4-bit S-box (Table 1) is applied 16 times in parallel in each round.

Table 1. S-Box

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	1	a	4	c	6	f	3	9	2	d	b	7	5	0	8	e

Bit Permutation: The diffusion layer uses a permutation P_{64} (Table 2) on 64 bits in each round.

Table 2. Bit permutation

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P_{64}(i)$	0	17	34	51	48	1	18	35	32	49	2	19	16	33	50	3
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P_{64}(i)$	4	21	38	55	52	5	22	39	36	53	6	23	20	37	54	7
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P_{64}(i)$	8	25	42	59	56	9	26	43	40	57	10	27	24	41	58	11
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P_{64}(i)$	12	29	46	63	60	13	30	47	44	61	14	31	28	45	62	15

Round Constants: In each round, the 6-bit round constant C_r given in the Table 3 is used, where c_0 refers to the least significant bit. For subsequent rounds, it is updated as follows:

$$(c_5, c_4, c_3, c_2, c_1, c_0) \leftarrow (c_4, c_3, c_2, c_1, c_0, c_5 \oplus c_4 \oplus 1)$$

Table 3. Round constants

Rounds (r)	Constants (C_r)
1–14	01 03 07 0F 1F 3E 3D 3B 37 2F 1E 3C 39 33
15–28	27 0E 1D 3A 35 2B 16 2C 18 30 21 02 05 0B

4 Differential Cryptanalysis

Differential cryptanalysis was applied on DES [19] and its exhaustive attack complexity was reduced. This created a path for other cryptanalytic techniques e.g. linear [16], impossible differential [6], algebraic [10], etc. [9]. While designing a block cipher, its output is tested for indistinguishability from the random permutations. However, there may not exist any relationship between the single input and output occurrences but there may exist the non-random relations between the input and output differences. The basic approach of differential attack is to study the propagation of input differences and exploitation of non-random relations between the input and output differences. This attack works with differential characteristics providing the high probability relation between the input and output differences. The high probability differential characteristics are used in the key recovery attack by adding some rounds on the top and bottom of the differential characteristic.

4.1 Classical Differential Distinguisher

There exists several automated techniques to search the optimal differential distinguishers for block ciphers [13]. In this paper, we use the available differential distinguishers for SPECK32 [1] and SIMON32 [7]. We extend the 6-round distinguisher for SPECK32 and 7-round distinguisher for SIMON32 using the ML distinguisher. For GIFT64, we construct the high probability differential characteristics for 4 rounds using the branch-and-bound based search technique [15] and extend this distinguisher with the ML distinguisher.

4.1.1 Differential Characteristic for SPECK32

Abed et al. [1] presented the 9-round differential characteristics for SPECK32 with the probability of 2^{-31}. We choose 8-round differential characteristic presented in Table 4 [1] and use the 6-round differential characteristic ($\Delta_0 \rightarrow \Delta_6$) with the probability of 2^{-13} in our experiments.

Table 4. Differential characteristic of SPECK32 [1]

Round (r)	Input difference ($\triangle_r$)	Probability (2^{-p_r})
0	0211 0A04	1
1	2800 0010	2^{-4}
2	0040 0000	2^{-6}
3	8000 8000	2^{-6}
4	8100 8102	2^{-7}
5	8000 840A	2^{-9}
6	850A 9520	2^{-13}
7	802A D4A8	2^{-19}
8	81A8 D30B	2^{-26}

4.1.2 Differential Characteristic for SIMON32

Biryukov et al. [7] presented the 12-round differential characteristics for SIMON32 with the probability of 2^{-34}. From the 12-round characteristic presented in Table 5 [7], we use the 7-round differential characteristic ($\Delta_0 \rightarrow \Delta_7$) with the probability of 2^{-16} in our experiments.

Table 5. Differential characteristic of SIMON32 [7]

Round (r)	Input difference ($\triangle_r$)	Probability (2^{-p_r})
0	0400 1900	1
1	0100 0400	2^{-2}
2	0000 0100	2^{-4}
3	0100 0000	2^{-4}
4	0400 0100	2^{-6}
5	1100 0400	2^{-8}
6	4200 1100	2^{-12}
7	1D01 4200	2^{-16}
8	0500 1D01	2^{-24}
9	0100 0500	2^{-27}
10	0100 0100	2^{-29}
11	0500 0100	2^{-31}
12	1500 0500	2^{-34}

4.1.3 Differential Characteristic for GIFT64

We construct the 4-round optimal differential characteristic with high probability using branch-and-bound based search algorithm [15]. We use this 4-round differential characteristic with the probability of 2^{-12} to construct the differential-ML distinguisher for GIFT64 (Table 6).

Table 6. Differential characteristic of GIFT64

Round (r)	Input difference ($\triangle_r$)	Probability (2^{-p_r})
0	0000 0000 0000 000A	1
1	0000 0000 0000 0001	2^{-2}
2	0008 0000 0000 0000	2^{-5}
3	0000 0000 2000 1000	2^{-7}
4	0044 0000 0011 0000	2^{-12}

4.2 Differential Distinguisher Using Machine Learning

For a chosen input difference, we use the neural distinguisher design proposed by A. Gohr [12]. We also consider the improvements in this design suggested by Baksi et al. [2] and use dense layers of MLPs (Multi Layers Perceptrons) instead of the convolution networks. We use two hidden layers with 1024 neurons in each layer and train the model on ciphertext differences rather than ciphertext pairs. These improvements increase the learning efficiency of the model.

The model is trained on the data with chosen and random differences. This approach works well because the model learns sensitivity as well as specificity in the data. The sensitivity corresponds to the true positive predictions while the specificity corresponds to the true negative predictions. Initially, we generate a set of random plaintexts $(P_1, P_2, \cdots, P_N)$ and assign a label 0 or 1 to each plaintext randomly. If label of the plaintext P_i is 1, then we generate another plaintext $P_i^{'}$ having a difference Δ_r with P_i otherwise $P_i^{'}$ is generated randomly. The difference Δ_r corresponds to the output difference of the classical differential distinguisher. We encrypt the plaintexts P_i and $P_i^{'}$ using the s-round CIPHER$_s$ to get the ciphertexts C_i and $C_i^{'}$. The set of ciphertext differences $(C_i \oplus C_i^{'})$ along with the labels is used as training data (TD) for the training phase. Other than the training data, we also generate the validation data (VD) which is used by the trained model M to determine the validation accuracy. Size of TD and VD is subjected to the available computing resources. We train the model M on training data till the validation accuracy is saturated. The saturation implies that there is a negligible improvement in the validation accuracy of i^{th} training epoch (α_{s_i}) in comparison to the validation accuracies ($\alpha_{s_{i-1}}$ and $\alpha_{s_{i-2}}$) of the last two training epoches. We consider the model M as a valid distinguisher $(D^{ML}_{r+1\cdots r+s})$ if the validation accuracy (α_s) is at least 0.51 (Algorithm 4).

Once a valid ML based distinguisher$(D^{ML}_{r+1\cdots r+s})$ is obtained, we generate a pair of plaintexts with chosen difference (Δ_r). ORACLE is queried for the corresponding ciphertexts and $D^{ML}_{r+1\cdots r+s}$ is used to make the prediction on ciphertexts difference. If the prediction probability is at least 0.51 then we consider that the ciphertext pair belongs to the CIPHER$_s$ otherwise not. The accuracy of such prediction is expected to be α_s.

Algorithm 4: ML based differential distinguisher $D^{ML}_{r+1\cdots r+s}$

Function `DataGeneration`(N,Δ_r,*s=no. of rounds*)**:**
  Data Set (D) $\leftarrow$ (.)
  $K \leftarrow$ Choose a random key
  $(P_1, P_2, \cdots, P_N) \leftarrow$ Generate a set of random plaintexts
  $(b_1, b_2, \cdots, b_N) \leftarrow$ Initialize a set of labels
  for $i \leftarrow$ *1 to* N **do**
    $b_i \leftarrow random(0, 1)$ ▷ $random(0, 1)$ return either 0 or 1 randomly
    if $b_i = 0$ **then**
      $P_i^{'} \leftarrow$ Choose a random plaintext
    end
    else
      $P_i^{'} = P_i \oplus \Delta_r$
    end
    $C_i \leftarrow \text{CIPHER}_s(P_i, \text{K})$ ▷ s-round encryption
    $C_i^{'} \leftarrow \text{CIPHER}_s(P_i^{'}, \text{K})$ ▷ s-round encryption
    Append D by $(C_i \oplus C_i^{'}, b_i)$
  end
  return D
End Function
Procedure `Trainig Phase`($D^{CD}_{1\cdots r}(\Delta_0 \rightarrow \Delta_r)$,*s=no of rounds*)**:**
  Training Data (TD) $\leftarrow$ DataGeneration(2^{25}, Δ_r,s)
  Validation Data (VD) $\leftarrow$ DataGeneration(2^{22}, Δ_r,s)
  for $i \leftarrow$ *1 to* 10 **do**
    Train ML Model (M) on TD
    Validate M on VD
    $\alpha_{s_i} \leftarrow$ Validation Accuracy of M
    if *(*$i \geq 3$ *and* $\alpha_{s_i} \approx \alpha_{s_{i-1}}$ *and* $\alpha_{s_{i-1}} \approx \alpha_{s_{i-2}}$*)* **then**
      $\alpha_s = \alpha_{s_i}$
      goto Line 32
    end
  end
  if $\alpha_s \geq 0.51$ **then**
    $D^{ML}_{r+1\cdots r+s} \leftarrow M$
  end
  else
    M is not a valid distinguisher
  end
End Procedure
Procedure `Prediction Phase`($D^{CD}_{1\cdots r}(\Delta_0 \rightarrow \Delta_r)$, $D^{ML}_{r+1\cdots r+s}$)**:**
  $P \leftarrow$ Choose a random plaintext
  $P^{'} = P \oplus \Delta_r$
  $C \leftarrow \text{ORACLE}(P)$
  $C^{'} \leftarrow \text{ORACLE}(P^{'})$
  $p \leftarrow$ prediction probability for $(C \oplus C^{'})$ using $D^{ML}_{r+1\cdots r+s}$
  if *(*$p \geq 0.51$*)* **then**
    ORACLE = CIPHER_s
  end
  else
    ORACLE $\neq \text{CIPHER}_s$
  end
End Procedure

5 Differential-ML Distinguisher: Extending Classical Differential Distinguisher Using Machine Learning

The accuracy plays an important role to design the machine learning based differential distinguisher. There is a trade-off between the accuracy and the number of rounds covered. If we increase the number of rounds then accuracy of the ML distinguisher may decrease. The data complexity of a low accuracy distinguisher cannot be compared with the classical distinguisher due to high amount of false positive and false negative in the ML distinguisher. Therefore, we propose a new technique which uses the ML distinguisher to extend the existing classical distinguisher. Since, the accuracy of the proposed extended distinguisher is high, we can compare its data complexity with the classical distinguisher.

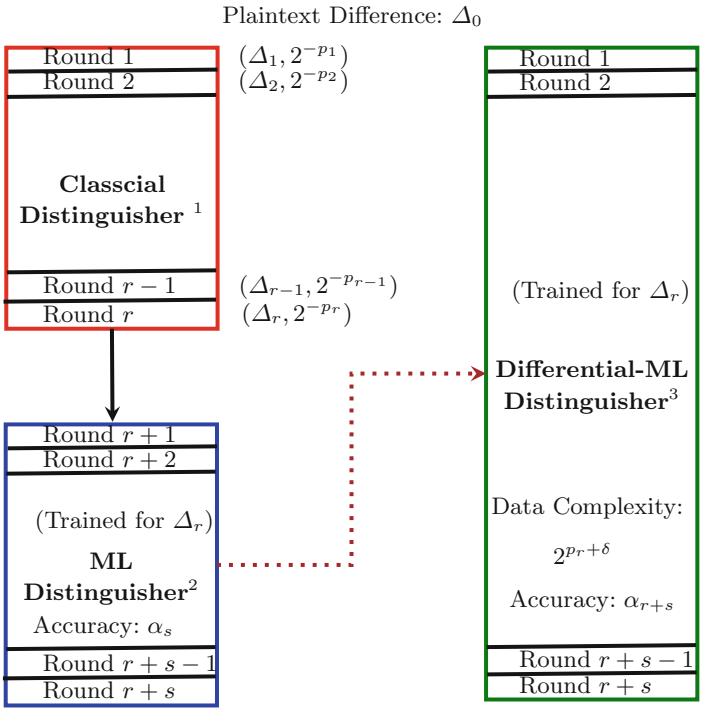


Fig. 1. Extending the classical distinguisher using ML distinguisher (1. Classical distinguisher: $D^{CD}_{1\cdots r}$ 2. ML distinguisher: $D^{ML}_{r+1\cdots r+s}$ 3. Differential-ML distinguisher: $D^{CD\to ML}_{1\cdots r+s}$)

Algorithm 5: Differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$: ($D^{CD}_{1\cdots r}$, $D^{ML}_{r+1\cdots r+s}$, $T = \alpha_s$, C_T, β)

Function `Construction Phase`($D^{CD}_{1\cdots r}$*(Data:* 2^{p_r}*)*, $D^{ML}_{r+1\cdots r+s}$, $T = \alpha_s$)**:**
  $\delta \leftarrow 0$
  repeat
    for $k \leftarrow$ *1 to 10* **do**
      $K \leftarrow$ Choose a random key
      $(P_{\Delta_0}, P^{'}_{\Delta_0}) \leftarrow 2^{\delta} * 2^{p_r}$ plaintext pairs with difference Δ_0
      $(P_R, P^{'}_R) \leftarrow 2^{\delta} * 2^{p_r}$ plaintext pairs with random difference
      $(C_{\Delta_0}, C^{'}_{\Delta_0}) \leftarrow$ (CIPHER$_{r+s}(P_{\Delta_0}, K)$,CIPHER$_{r+s}(P^{'}_{\Delta_0}, K)$)
      $(C_R, C^{'}_R) \leftarrow$ (CIPHER$_{r+s}(P_R, K)$,CIPHER$_{r+s}(P^{'}_R, K)$)
      $p_{\Delta_0} \leftarrow$ prediction probabilities for $(C_{\Delta_0} \oplus C^{'}_{\Delta_0})$ using $D^{ML}_{r+1\cdots r+s}$
      $p_R \leftarrow$ prediction probabilities for $(C_R \oplus C^{'}_R)$ using $D^{ML}_{r+1\cdots r+s}$
      TP $\leftarrow$ number of elements with $p_{\Delta_0} \geq T$
      TN $\leftarrow$ number of elements with $p_R \geq T$
      Plot the curve for TP and TN values
    end
    $\delta \leftarrow \delta + 1$
  until *(TP and TN curves do not intersect)*;
  $C_T \approx$ average of ordinates of closest points on TP and TN curves
  Data Complexity(β) $\leftarrow 2^{\delta} * 2^{p_r}$
  return C_T, β
End Function
Procedure `Prediction Phase`($D^{CD\rightarrow ML}_{1\cdots r+s}$)**:**
  Test Data (TD) $\leftarrow$ (.)
  for $i \leftarrow$ *1 to* β **do**
    $P_i \leftarrow$Choose a random plaintext
    $P^{'}_i = P_i \oplus \Delta_0$
    $C_i \leftarrow$ ORACLE(P_i)
    $C^{'}_i \leftarrow$ ORACLE($P^{'}_i$)
    Append TD by $C_i \oplus C^{'}_i$
  end
  $p \leftarrow$ prediction probabilities for elements in TD using $D^{ML}_{r+1\cdots r+s}$
  if *((number of pairs with* $p \geq T$*)* $\geq C_T$*)* **then**
    ORACLE = CIPHER$_{r+s}$
  end
  else
    ORACLE $\neq$ CIPHER$_{r+s}$
  end
end Procedure

To extend the r-round classical differential distinguisher $D^{CD}_{1\cdots r}$ ($\Delta_0 \rightarrow \Delta_r$), we use the difference Δ_r to model s-round distinguisher ($D^{ML}_{r+1\cdots r+s}$) with an accuracy α_s. The accuracy α_s defines the distinguishing ability of the distinguisher and better accuracy gives better predictions. Now, the distinguisher $D^{ML}_{r+1\cdots r+s}$ can be used to distinguish the output of CIPHER$_s$ with an accuracy α_s.

The data complexity of the r-round classical differential distinguisher ($\Delta_0 \rightarrow \Delta_r$, probability: 2^{-p_r}) is 2^{p_r} chosen plaintext pairs. It is expected to get at least one occurrence of Δ_r in the output difference. If we provide 2^{p_r} ciphertext pairs after the $(r+s)$ rounds of encryption to the distinguisher $D^{ML}_{r+1\cdots r+s}$ then we expect that the ML distinguisher $D^{ML}_{r+1\cdots r+s}$ will correctly predict one occurrence corresponding to the difference Δ_r. Since ML distinguisher learns the multiple output differences, we expect that it will learn the pattern of differences which are suggested by the classical differential distinguisher. Therefore, we require at least 2^{p_r} data to model the $(r+s)$-round differential-ML distinguisher ($D^{CD\rightarrow ML}_{1\cdots r+s}$). Now, the accuracy α_s of the s-round ML distinguisher plays a significant role. If accuracy α_s is low then the accuracy α_{r+s} of the distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ for 2^{p_r} data will also be low. The accuracy of the differential-ML distinguisher must be high to compare it with the $(r+s)$-round classical differential distinguisher. To increase the accuracy α_{r+s}, we propose a novel technique which requires additional data (2^{δ}). Therefore, data complexity of the differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ becomes $2^{p_r+\delta}$, where δ defines the additional data required to increase the accuracy of predictions (Fig. 1).

In our technique, we define the differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ with five parameters ($D^{CD}_{1\cdots r}$, $D^{ML}_{r+1\cdots r+s}$, $T = \alpha_s$, C_T, β). Where, T is the threshold probability, C_T is the cutoff on the number of pairs with the prediction probability $\geq$ T and β is data complexity of the differential-ML distinguisher. These parameters are required to construct the differential-ML distinguisher. We set α_s as the threshold probability (T) and propose an experimental approach to calculate C_T and β (Algorithm 5). We start with the minimum data (2^{p_r}) and set δ as 0. We generate a set of 2^{p_r} plaintext pairs with the difference Δ_0 and another set of 2^{p_r} plaintext pairs with the random differences. These pairs are encrypted using the CIPHER$_{r+s}$. The distinguisher $D^{ML}_{r+1\cdots r+s}$ is used to get the prediction probabilities p_{Δ_0} and p_R as explained in Algorithm 5.

Using these probabilities, we get the True Positive (TP) and the True Negative (TP) values. We repeat this process 10 times and plot the curve for TP and TN values. If the TP and TN curves intersect, then we increase the data requirement and repeat the process with the increased data. We repeat the process until we get the non intersecting curves. Once such curves are obtained, data complexity (β) of the differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ becomes $2^{p_r+\delta}$. To calculate C_T, we take average of the closest points on the TP and TN curves. Closest points correspond to the minimum number of predictions on TP curve and maximum number of predictions on TN curve. The value of C_T is taken as the separation cutoff for TP and TN curves and it is used by the distinguisher ($D^{CD\rightarrow ML}_{1\cdots r+s}$) to distinguish the data sample correctly. The complete procedure to construct the differential-ML distinguisher is described in Algorithm 5.

The differential-ML distinguisher $D_{1\cdots r+s}^{CD\rightarrow ML}$ act as the $(r+s)$-round distinguisher. We choose a set of β plaintext pairs with the difference Δ_0 and get the ciphertext pairs after $(r+s)$ rounds. The distinguisher $D_{1\cdots r+s}^{CD\rightarrow ML}$ makes the prediction for each ciphertext pair. If the number of pairs with the prediction probability greater than T is above the cutoff threshold C_T then the distinguisher $D_{1\cdots r+s}^{CD\rightarrow ML}$ classifies whether the given data is an output from the target CIPHER$_{r+s}$ or not. The prediction procedure is described in the prediction phase of the Algorithm 5.

With the proposed distinguisher $D_{1\cdots r+s}^{CD\rightarrow ML}$, we can achieve very high accuracy to distinguish the output of the CIPHER$_{r+s}$ and it can be used to mount a key recovery attack. The experiments to construct the differential-ML distinguishers are presented in the next section.

6 Experimental Results

We construct the differential-ML distinguisher for the 32-bit variants of the lightweight block ciphers SPECK and SIMON and 64-bit variant of GIFT. We experimented on 32-bit and 64-bit block ciphers due to constraints on available resources. With more computing power, ciphers with larger block size can be explored to construct differential-ML distinguisher. We extend the classical differential distinguisher discussed in Sect. 4 with the ML distinguisher. Using this novel technique, we construct the differential-ML distinguishers for 9-round SPECK32, 12-round SIMON32, and 8-round GIFT64 with (much) less data complexity than the classical distinguishers.

We used Keras-GPU[1] library in Google colab[2] for the model training and predictions. In each experiment, ADAM optimizer is used for the adaptive learning rates and the Mean-Square-Error is used as the loss function. The validation batch accuracy is considered as the accuracy (α_s) of the trained model.

6.1 Differential-ML Distinguisher: SPECK32

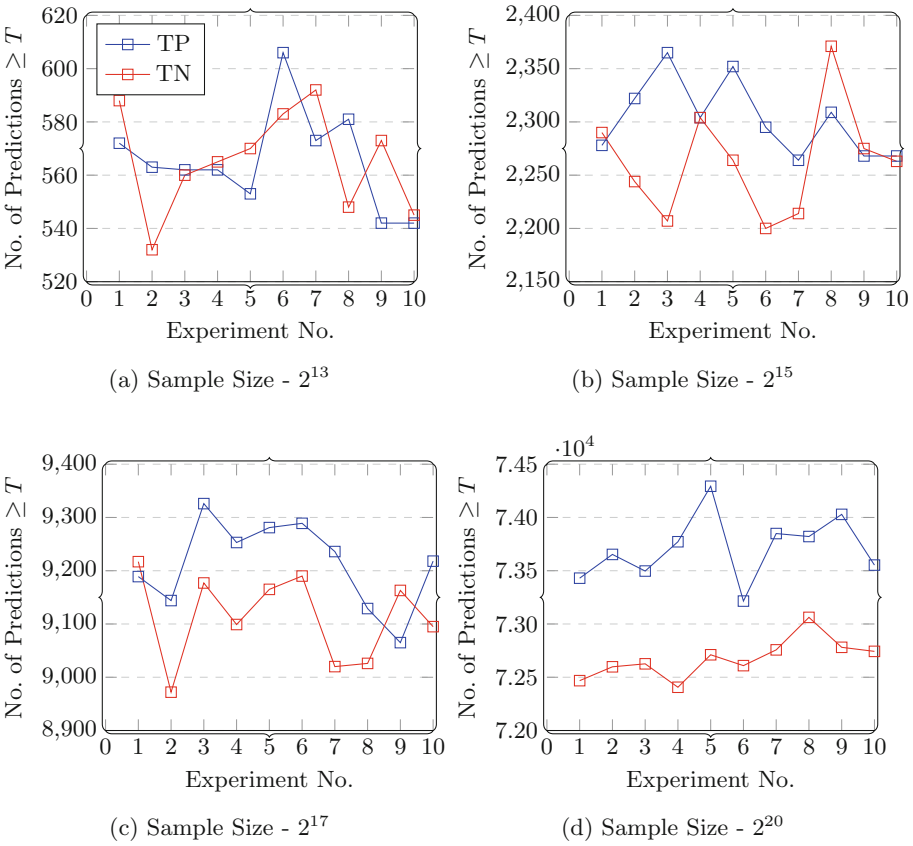
For SPECK32, we use the classical differential characteristic for 6 rounds ($\Delta_0 \rightarrow \Delta_6$) as described in the Table 4. We have an output difference 0x850A9520 after 6 rounds with the probability of 2^{-13}. We train the 3-round ML distinguisher using Δ_6 as the input difference and the model is trained with the accuracy of 0.79 using Algorithm 4. The batch accuracy and loss are described in the Appendix A.

6.1.1 Construction

The probability of the 6-round classical differential distinguisher is 2^{-13}. Therefore, we will require at least 2^{13} data to make the predictions with the 9-round differential-ML distinguisher. We calculate T, C_T, and β as discussed in

[1] https://keras.io.
[2] https://colab.research.google.com.

Algorithm 5 and construct the 9-round differential-ML distinguisher $D^{CD \to ML}_{1 \cdots r+s}$ by extending the 6-round classical distinguisher. We draw the graphs for TP and TN values (Fig. 2) and calculate data complexity (β) and cutoff (C_T). We experimented with the various samples of different sizes and obtained a clear separation between true positive and true negative curves for a sample size of 2^{20}. We calculate the value of C_T as 73100 and β as 2^{20} with the help of graph (d) in Fig. 2.



(a) Sample Size - 2^{13}

(b) Sample Size - 2^{15}

(c) Sample Size - 2^{17}

(d) Sample Size - 2^{20}

Fig. 2. Calculation of C_T and data complexity (β) for SPECK32 ($D^{CD \to ML}_{1 \cdots r+s}$)

6.1.2 Prediction

We constructed the 9-round differential-ML distinguisher $D^{CD \to ML}_{1 \cdots r+s}$ in the previous subsection. The accuracy (α_{r+s}) of this differential-ML distinguisher for different experiments is mentioned in the Table 7.

In the experiments, we take 50 samples belonging to the plaintext difference Δ_0 (=0x0211 0A04) of the classical distinguisher and other 50 samples belonging to the random input differences. The differential-ML distinguisher $D^{CD \to ML}_{1 \cdots r+s}$

Table 7. Accuracy for SPECK32 with $T = 0.79$, $C_T = 73100$ and $\beta = 2^{20}$

Experiment no.	Sample size	Correctly distinguished (true positive, true negative)
1	100	98(50,48)
2	100	98(50,48)
3	100	99(49,50)
4	100	97(48,49)
5	100	96(49,47)

predicts whether the given sample belongs to the difference Δ_0 or not by using the Algorithm 5. We used 2^{20} data in each sample and achieved the accuracy (α_{r+s}) more than 96% in each experiment.

Therefore, the data complexity of the 9-round differential-ML distinguisher for SPECK32 is 2^{20}. However, the data complexity of the 9-round classical differential distinguisher is 2^{31} as presented in [1]. The best known differential characteristics for SPECK32 exists for 9-rounds with the data complexity of 2^{30} [7]. Using the differential-ML technique, we have constructed the 9-round distinguisher with the data complexity far less than the existing classical differential distinguisher.

6.2 Differential-ML Distinguisher: SIMON32

For SIMON32, we use the classical differential characteristic for 7 rounds as described in the Table 5. We have an output difference 0x1D014200 (Δ_7) after 7 rounds with the probability of 2^{-16}. We use Δ_7 as the input difference for the training phase of the 5-round ML distinguisher. We train the model with the accuracy of 0.57 using the Algorithm 4. The batch accuracy and loss are described in the Appendix A.

6.2.1 Construction

The probability of the 7-round classical differential distinguisher is 2^{-16}. So, we will require at least 2^{16} data for the 12-round differential-ML distinguisher of SIMON32 and additional data (2^{δ}) will be required to increase the accuracy of the differential-ML distinguisher. Similar to the SPECK32 case, we require T, C_T, and β to construct the 12-round differential-ML distinguisher $D^{CD \rightarrow ML}_{1 \cdots r+s}$ which extends the existing 7-round classical distinguisher. We calculate the data complexity (β) and cutoff (C_T) by using the Algorithm 5 and the graphs for TP and TN values (Fig. 3). It is observed from the graphs that a clear separation between true positive and true negative values exists for the sample size of 2^{22}. We calculated the value of C_T as 656300 and data complexity(β) as 2^{22} on the basis of this separation.

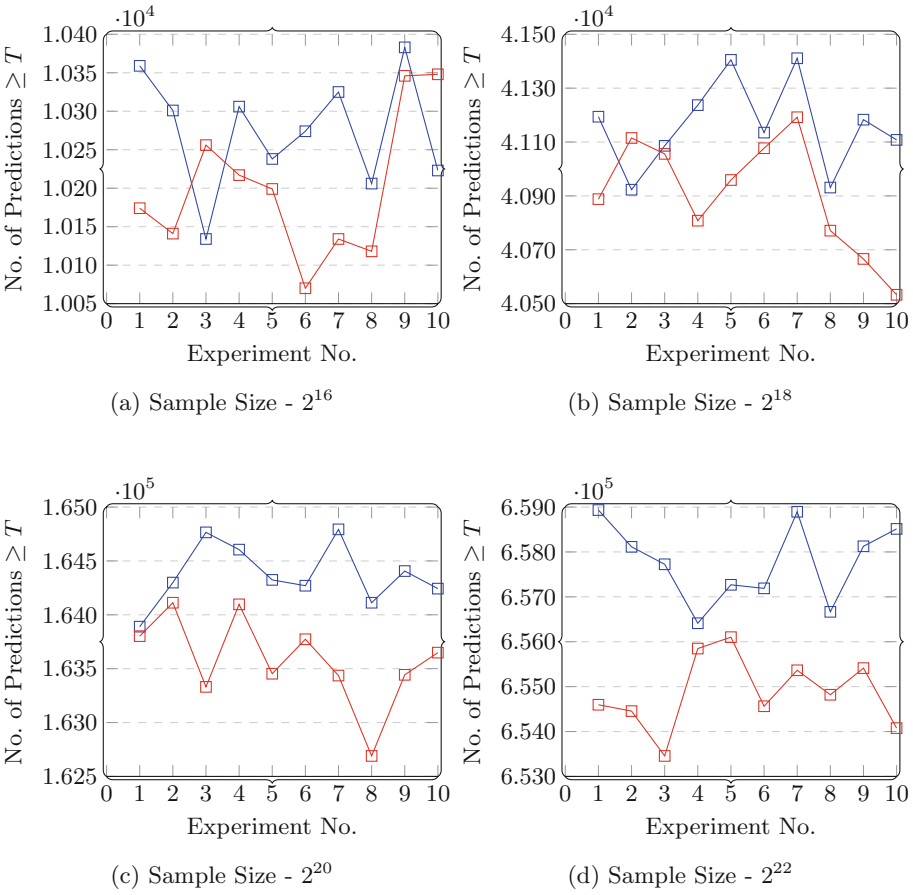


Fig. 3. Calculation of C_T and data complexity (β) for SIMON32 ($D^{CD\rightarrow ML}_{1\cdots r+s}$)

6.2.2 Prediction

The 5-round ML distinguisher ($D^{ML}_{r+1\cdots r+s}$) is trained with the validation accuracy of 0.57. It is used to extend the 7-round classical differential distinguisher. The accuracy of the 12-round differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ for different experiments is mentioned in the Table 8.

Similar to the previous case, we take 50 samples belonging to the initial input difference Δ_0 (=0x04001900) of the classical distinguisher and other 50 samples belonging to the random input differences. We make predictions with 2^{22} data using the value of C_T calculated in the previous step and the accuracy (α_{r+s}) greater than 97% is achieved in each experiment. From these experiments, 12-round differential-ML distinguisher $D^{CD\rightarrow ML}_{1\cdots r+s}$ with data complexity of 2^{22} is constructed, while data complexity for the 12-round classical differential distinguisher is 2^{34} (Table 5). In this case, we present the first 12-round distinguisher with the data complexity less than 2^{32}. This shows that the differential-ML

Table 8. Accuracy for SIMON32 with $T = 0.57$, $C_T = 656300$ and $\beta = 2^{22}$

Experiment no.	Sample size	Correctly distinguished (true positive, true negative)
1	100	98(48,50)
2	100	98(48,50)
3	100	98(49,49)
4	100	97(48,49)
5	100	98(48,50)

distinguisher provides the better results than the classical differential distinguisher in case of SIMON32 also.

6.3 Differential-ML Distinguisher: GIFT64

For GIFT64, we searched an optimal differential characteristic for 4 rounds which is described in the Table 6. We obtain the output difference after 4 rounds as Δ_4 = 0x0044000000110000 with the probability of 2^{-12}. The difference Δ_4 is used to train the 4-round ML based distinguisher. We train a model with the accuracy of 0.65 using the Algorithm 4. The batch accuracy and loss are described in Appendix A.

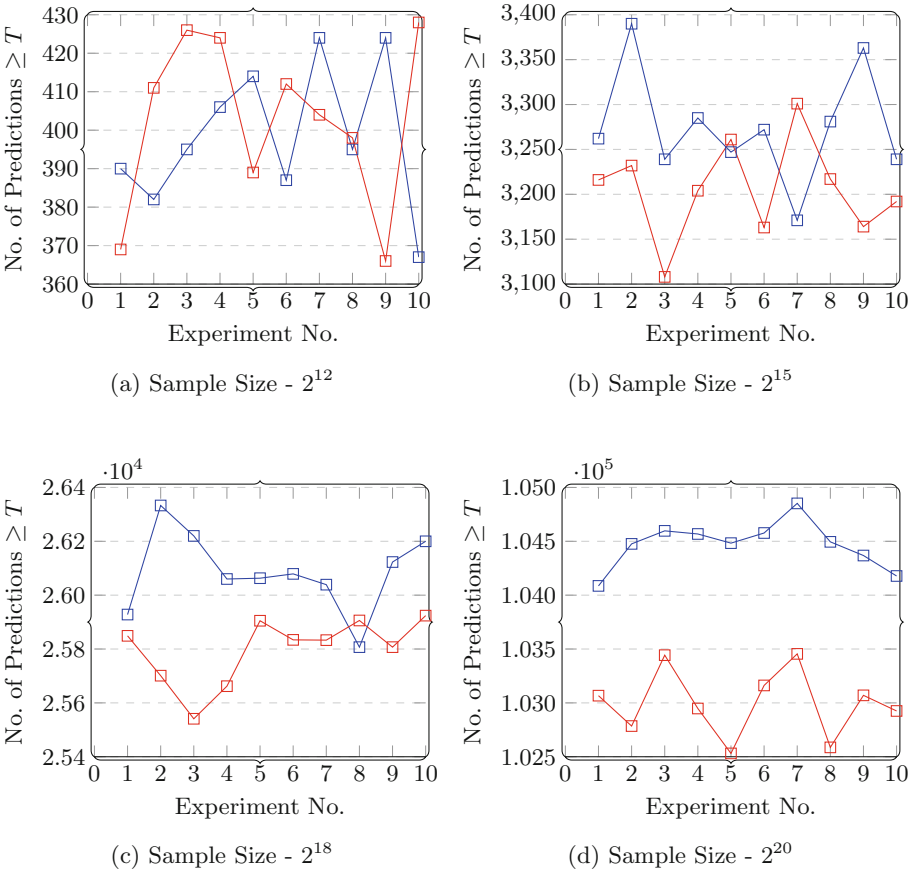
6.3.1 Construction

The probability of the 4-round classical differential characteristic is 2^{-12}. Therefore, data complexity of the 4-round differential distinguisher will be 2^{12}. So, the 8-round differential-ML distinguisher for GIFT64 will require at least 2^{12} data. We calculate T, C_T, and data complexity (β) by using Algorithm 5. These are required to construct the 8-round differential-ML distinguisher by extending the 4-round classical differential distinguisher. It can be easily inferred from the graphs depicted in Fig. 4 that a clear separation between true positive and true negative values exists for the sample size of 2^{20}. We use this separation to get the cutoff threshold ($C_T = 103750$) and data complexity ($\beta = 2^{20}$) for $D^{CD \to ML}_{1 \cdots r+s}$.

6.3.2 Prediction

The 4-round ML distinguisher $D^{ML}_{r+1 \cdots r+s}$ is trained with the validation accuracy of 0.65 and it is used to extend the 4-round classical differential distinguisher. Accuracy of the 8-round differential-ML distinguisher $D^{CD \to ML}_{1 \cdots r+s}$ for different experiments is mentioned in the Table 9.

Similar to SPECK32 and SIMON32 cases, we take 50 samples belonging to the input difference Δ_0 (=0x000000000000000A) of the classical distinguisher and other 50 samples belonging to the random input differences. For each sample, we use 2^{20} data to achieve the accuracy (α_{r+s}) greater than 98% in each experiment. Therefore, data complexity of the 8-round differential-ML distinguisher



(a) Sample Size - 2^{12}

(b) Sample Size - 2^{15}

(c) Sample Size - 2^{18}

(d) Sample Size - 2^{20}

Fig. 4. Calculation of C_T and data complexity (β) for GIFT64 ($D_{1\cdots r+s}^{CD\rightarrow ML}$)

Table 9. Accuracy for GIFT64 with $T = 0.65$, $C_T = 103650$ and $\beta = 2^{20}$

Experiment no.	Sample size	Correctly distinguished (true positive, true negative)
1	100	99(50,49)
2	100	100(50,50)
3	100	98(50,48)
4	100	100(50,50)
5	100	100(50,50)

is 2^{20}, while data complexity of the 8-round classical differential distinguisher was 2^{38} [20].

6.4 Comparison with the Classical Differential Distinguishers

We have constructed the differential-ML distinguishers for the block ciphers based on three different types of structures (Feistel, SPN, and ARX). We are able to distinguish the same number of rounds using less amount of data in comparison to the classical distinguisher. These results indicate that our technique provides better results for the block ciphers based on all types of structures. The source code for the above mentioned experiments is available on GitHub[3]. We provide a comparison of the data complexities between the differential-ML distinguisher and the classical differential distinguisher in Table 10.

Table 10. Summary of results

Cipher	Distinguisher	Round	Data complexity	Source
SPECK32	Differential	9	2^{30}	[7]
SPECK32	**Differential-ML**	**9**	$\mathbf{2^{20}}$	**Sec. 6.1**
SIMON32	Differential	12	2^{34}	[7]
SIMON32	**Differential-ML**	**12**	$\mathbf{2^{22}}$	**Sec. 6.2**
GIFT64	Differential[7]	8	2^{38}	[20]
GIFT64	**Differential-ML**	**8**	$\mathbf{2^{20}}$	**Sec. 6.3**

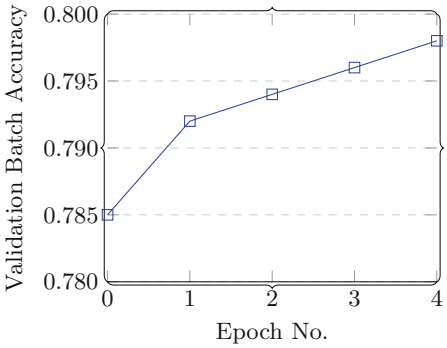
There exists differential distinguisher for 12 rounds with the data complexity of 2^{60}.

7 Conclusion

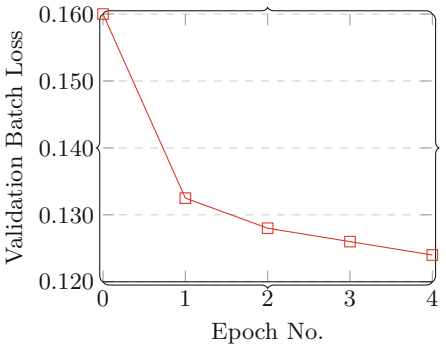
In this paper, we have proposed a novel technique to extend the classical differential distinguisher using machine learning. We have constructed the high accuracy (more than 96%) differential-ML distinguishers for 9-round SPECK32, 12-round SIMON32, and 8-round GIFT64. For SPECK32, we have extended the 6-round classical differential distinguisher with the 3-round ML distinguisher and the data complexity of 9-round differential-ML distinguisher is 2^{20}. For SIMON32, the classical differential distinguisher for 7-rounds is extended with the 5-round ML distinguisher and data complexity of the 12-round differential-ML distinguisher is 2^{22}. For GIFT64, the 8-round differential-ML distinguisher is constructed with the data complexity of 2^{20} whereas data complexity of the 8-round classical differential distinguisher was 2^{38}. The data complexity of the distinguishers for SPECK32, SIMON32, and GIFT64 is significantly reduced using differential-ML distinguishers in comparison to the classical distinguishers.

[3] https://github.com/tarunyadav/Differential-ML-Distinguisher.

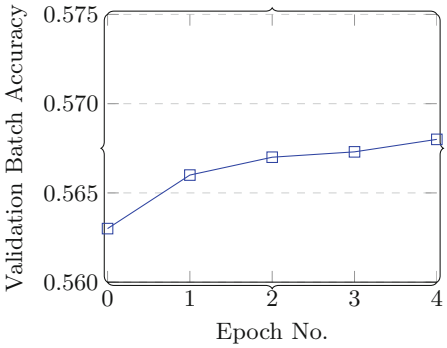
Appendix A - Accuracy and Loss Graphs



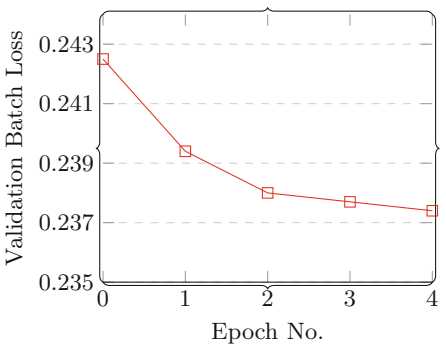
(a) SPECK32: Validation Batch Accuracy



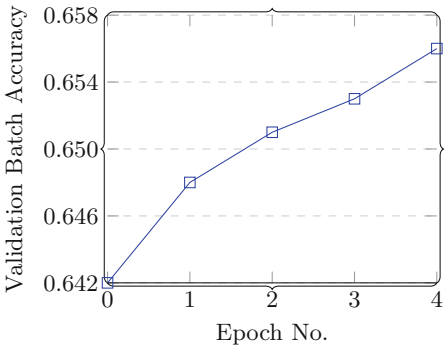
(b) SPECK32: Validation Batch Loss



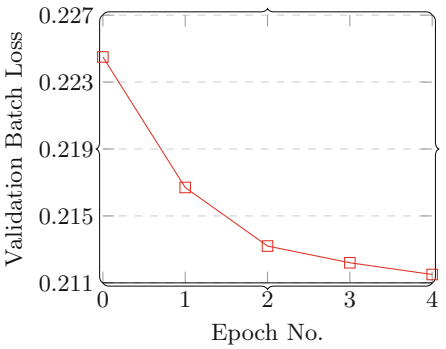
(c) SIMON32: Validation Batch Accuracy



(d) SIMON32: Validation Batch Loss



(e) GIFT64: Validation Batch Accuracy



(f) GIFT64: Validation Batch Loss

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential cryptanalysis of round-reduced Simon and Speck. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 525–545. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46706-0_27
2. Baksi, A., Breier, J., Dong, X., Yi, C.: Machine Learning Assisted Differential Distinguishers For Lightweight Ciphers (2020). https://eprint.iacr.org/2020/571
3. Banik, S., Pandey, S.K., Peyrin, T., Sasaki, Y., Sim, S.M., Todo, Y.: GIFT: a small present - towards reaching the limit of lightweight encryption. In: Cryptographic Hardware and Embedded Systems - CHES 2017–19th International Conference, Taipei, Taiwan, 25–28 September 2017, Proceedings, vol. 10529, pp. 321–345 (2017)
4. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404 (2013). https://eprint.iacr.org/2013/404
5. Bernstein, D.J., et al.: Gimli (2019)
6. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_2
7. Biryukov, A., Roy, A., Velichkov, V.: Differential analysis of block ciphers SIMON and SPECK. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 546–570. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46706-0_28
8. Biham, E., Shamir, A.: Differential cryptanalysis of the full 16-round DES. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 487–496. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_34
9. Bogdanov, A.: Analysis and design of block cipher constructions, Ph.D. thesis (2009)
10. Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 345–359. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_21
11. Daemen, J., Rijmen, V.: The Design of Rijndael. Springer, Heidelberg (2002). https://doi.org/10.1007/978-3-662-04722-4
12. Gohr, A.: Improving attacks on round-reduced Speck32/64 using deep learning. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 150–179. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_6
13. Hays, H.M.: A tutorial on linear and differential cryptanalysis. Cryptologia **26**(3), 188–221 (2002)
14. Knudsen, L., Robshaw, M.J.B.: Block Cipher Companion. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17342-4. ISBN 978-3-642-17341-7
15. Kumar, M., Suresh, T.S., Pal, S.K., Panigrahi, A.: Optimal differential trails in lightweight block ciphers ANU and PICO. Cryptologia **44**(1), 68–78 (2020)
16. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48285-7_33
17. Matsui, M.: On correlation between the order of S-boxes and the strength of DES. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 366–375. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0053451

18. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu, C.-K., Yung, M., Lin, D. (eds.) Inscrypt 2011. LNCS, vol. 7537, pp. 57–76. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34704-7_5
19. US National Bureau of Standards, Data Encryption Standard. Federal Information Processing Standards Publications, vol. 46 (1977)
20. Zhu, B., Dong, X., Yu, H.: MILP-based differential attack on round-reduced GIFT. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 372–390. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_19