Deep Learning-Based Rotational-XOR Distinguishers for AND-RX Block Ciphers: Evaluations on Simeck and Simon

Amirhossein Ebrahimi[1][0000−0001−8296−2729], David Gerault[2][0000−0001−8583−0668], and Paolo Palmieri[1][0000−0002−9819−4880]

[1] School of Computer Science & IT, University College Cork
{a.ebrahimimodhaddam,p.palmieri}@cs.ucc.ie
[2] Cryptography Research Centre, Technology Innovation Institute, Abu Dhabi, UAE,
david.gerault@tii.ael

Abstract. The use of deep learning techniques in cryptanalysis has garnered considerable interest following Gohr's seminal work in 2019. Subsequent studies have focused on training more effective distinguishers and interpreting these models, primarily for differential attacks. In this paper, we shift our attention to deep learning-based distinguishers for rotational XOR (RX) cryptanalysis on AND-RX ciphers, an area that has received comparatively less attention. Our contributions include a detailed analysis of the state-of-the-art deep learning techniques for RX cryptanalysis and their applicability to AND-RX ciphers like Simeck and Simon. Our research proposes a novel approach to identify DL-based RX distinguishers, by adapting the evolutionary algorithm presented in the work of Bellini et al. to determine optimal values for translation (δ) and rotation offset (γ) parameters for RX pairs. We successfully identify distinguishers using deep learning techniques for different versions of Simon and Simeck, finding distinguishers for the classical related-key scenario, as opposed to the weak-key model used in related work. Additionally, our work contributes to the understanding of the diffusion layer's impact in AND-RX block ciphers against RX cryptanalysis by focusing on determining the optimal rotation parameters using our evolutionary algorithm, thereby providing valuable insights for designing secure block ciphers and enhancing their resistance to RX cryptanalysis.

Keywords: AND-RX ciphers· Deep Learning · Cryptanalysis · Rotational-XOR cryptanalysis.

1 Introduction

Cryptography plays a crucial role in ensuring the security and privacy of information in modern communication systems. Block ciphers, in particular, are widely used to provide encryption for data transmission, ensuring that the content remains confidential and secure from unauthorized access. However, the effectiveness of block ciphers is always being tested, and researchers are continually exploring new ways to improve their resilience against attacks like differential [7], linear [19], algebraic attacks [3], etc. Among several cryptanalysis techniques, Rotational-XOR (RX) cryptanalysis has emerged as a powerful method to evaluate the security of block ciphers, particularly ARX and AND-RX ciphers such as Speck, Simon, and Simeck [1].

In recent years, artificial intelligence (AI) and deep learning have shown great potential in a variety of applications, including cryptanalysis. Their ability to analyze complex patterns and relationships in large datasets has motivated researchers to explore new techniques for breaking cryptographic algorithms[10, 6, 20]. This paper aims to investigate the application of deep learning in the RX cryptanalysis of AND-RX block ciphers, with a focus on Simon and Simeck, and proposes an approach to see the impact of diffusion layers in these ciphers.

The conventional cryptographic analysis techniques utilized in RX cryptanalysis commonly depend on weak-key models, wherein statistical methods are utilized to detect distinguishers and possible vulnerabilities. Nevertheless, these methods are constrained, as achieving a good distinction with a limited weak-key model may not be feasible. In this context, deep learning has been proposed as an alternative technique, offering the possibility of improved results in cryptanalysis tasks. Our proposed method has enabled us to acquire distinguishers for full-key classes concerning Simeck and Simon ciphers.

In addition to assessing the security of ciphers, finding the best parameters for diffusion layers is a crucial aspect of cipher design. The diffusion layer plays a significant role in ensuring that minimal alterations in plaintext or key inputs lead to substantial changes in the ciphertext output, making it challenging for adversaries to decipher the original data. In this paper, we propose a new approach that involves using a modified version of the optimizer in [5] to determine the best RX differential inputs and the optimal shift parameter for finding the longest round distinguisher with the aid of deep learning classifiers. Furthermore, we use this optimizer to identify the best set of rotations in the diffusion layer that works against deep learning optimizers, specifically for Simeck-like ciphers. Our approach ensures that deep learning distinguishers cannot find the optimal distinguishers, thereby enhancing the overall security of the ciphers. Therefore, our method demonstrates the potential for improving the security of ciphers while also enhancing the efficiency of the design process by utilizing deep learning classifiers in combination with an optimizer. Our findings contribute to the ongoing efforts to enhance the security of AND-RX block ciphers and highlight the potential of AI applications in Rotational-XOR cryptanalysis.

Our Deep Learning (DL)-based distinguishers demonstrate superior performance on the Simeck cipher compared to the Simon cipher. In order to juxtapose our achieved Deep Learning (DL)-based distinguishers with other related-key DL-based distinguishers for the Simeck cipher, the results are presented in Table 1. It's important to note that our distinguisher was trained exclusively on a single pair, whereas the existing literature offers distinguishers trained on eight pairs for the Simeck cipher. Consequently, we implemented the technique introduced in [11] to compute an amalgamated score for eight pairs. Furthermore, a comparison between our DL-based RX distinguishers and past RX distinguishers of the Simeck cipher can be found in Table 2.

Our work introduces a superior related-key DL distinguisher for Simeck 64/128 cipher and marginally behind for Simeck 32/64, according to Table 1. Furthermore, our research introduces a novel distinguisher that is specifically designed for RX cryptanalysis and trained on the entire key space. This distinguisher can be further scrutinized to assess how the accuracy of these distinguishers is affected by different keys.

Table 1: Comparison of related-key DL-based distinguishers for Simeck. RX: Rotational-Xor cryptanalysis, RD: Related-key Differential cryptanalysis. The Combined Accuracy Score [11] for m pairs is $\frac{1}{1+\prod_{i=1}^{m}\frac{1-p_i}{p_i}}$

	Round	Combined Accuracy Score	Pairs	Attack Type	Ref.
Simeck 32/64	13	0.9950	8	RD	[17]
	14	0.6679	8	RD	[17]
	15	0.5573	8	RD	[17]
	15	**0.5134**	**1**	**RX**	**This Work**
	15	**0.5475**	**8**	**RX**	**This Work**
Simeck 64/128	18	0.9066	8	RD	[17]
	19	0.7558	8	RD	[17]
	20	0.6229	8	RD	[17]
	20	**0.5212**	**1**	**RX**	**This Work**
	20	**0.6338**	**8**	**RX**	**This Work**

Table 2: Comparison of the RX distinguishers for different versions of Simeck

Cipher	Rounds	Data Complexity	Size of Weak Key Class	DL-based	Ref.
Simeck32/64	**15**	$\mathbf{2^{20}}$	**Full**	**Yes**	**This Work**
	15	2^{18}	2^{44}	No	[18]
	19	2^{24}	2^{30}	No	[18]
	20	2^{26}	2^{30}	No	[18]
Simeck48/96	**17**	$\mathbf{2^{20}}$	**Full**	**Yes**	**This Work**
	16	2^{18}	2^{68}	No	[18]
	18	2^{22}	2^{66}	No	[18]
	19	2^{24}	2^{62}	No	[18]
	27	2^{44}	2^{46}	No	[18]
Simeck64/128	**20**	$\mathbf{2^{20}}$	**Full**	**Yes**	**This Work**
	25	2^{34}	2^{80}	No	[18]
	34	2^{56}	2^{58}	No	[18]

1.1 Related Works

Simon [4] and Simeck [22] are lightweight block ciphers that have gained popularity due to their simplicity and efficiency. However, several attacks have been proposed against these ciphers, including related-key and weak-key attacks.

Liu et al. [16] propose an automatic search algorithm to find optimal differential trails in Simon and Simeck ciphers. The authors use Matsui's branch-and-bound algorithm to traverse input differences from low Hamming weight and break unnecessary branches. They also derive a more accurate upper bound on the differential probability of the Simon-like round function, which helps to improve the efficiency of the search algorithm. With this algorithm, they find the provably optimal differential trails for all versions of Simon and Simeck ciphers. In [21] a detailed analysis of Simon-like block ciphers and their related-key differential trails is presented. The authors identify that not only the Hamming weight but also the positions of active bits in the input difference affect the probability of differential trails. The authors proceed to reconstruct the Mixed Integer Linear Programming (MILP) model for Simon-like block ciphers,

eliminating quadratic constraints, and introducing an accurate objective function that reduces its degree to one through the inclusion of auxiliary variants. Additionally, they investigate and identify the optimal differential trails for Simon and Simeck, utilizing this model, and they obtain related-key differential trails. Their core findings encompass the discovery of optimal related-key differential trails for various versions of Simon and Simeck (Simon32/64, Simon48/96, Simon64/128, Simeck32/64, Simeck48/96, and Simeck64/128), along with the identification of impossible differentials for several iterations of Simon and Simeck.

Rotational cryptanalysis is a technique that explores the propagation of rotational pairs, which consist of pairs $(x, x \lll \gamma)$ where γ is the rotational offset. The success of this attack can be compromised when non-rotation-invariant constants are injected into the rotational pairs. Rotational-XOR (RX) cryptanalysis, a generalized attack method, accounts for these constants by incorporating their effect into the analysis of the propagation probability. RX-cryptanalysis considers an RX-pair of the form $(x, (x \lll \gamma) \oplus \delta)$ where δ is known as the translation.

Ashur and Liu [1] introduced the concept of an RX-difference, and demonstrated how RX-differences behave around modular addition. They presented a formula for computing the transition probability of RX-differences, which was verified experimentally using Speck32/64. Additionally, they provided guidance on the optimal choice of parameters and discussed two types of constants: round constants and constants that result from a fixed key.

Khovratovich et al [13] provided theoretical and practical support for the security of modular addition, rotation, and XOR-based (ARX) systems. They used rotational cryptanalysis to illustrate the best-known attack on reduced versions of the Threefish block cipher.

Lu et al. [18] extended RX-cryptanalysis to AND-RX ciphers that can be described using bitwise AND, XOR, and cyclic rotation operations. The authors formulated an equation for predicting the likelihood of RX-differences progressing through AND-RX rounds and established an SMT (Satisfiability Modulo Theories) model to investigate RX-characteristics in Simon and Simeck. They discovered RX-characteristics in Simeck across diverse block sizes, specifically for expansive groups of weak keys within the related-key model, and conducted an analysis of how the key schedule and the rotation quantities of the round function affect the propagation of RX-characteristics in Simon-like ciphers.

AI and ML methods have been utilized in various data security applications such as cryptographic algorithms, cryptanalysis, steganography, and others. At CRYPTO'19, Gohr introduced a novel cryptanalysis approach that harnessed the power of machine learning algorithms [10]. By employing deep neural networks, he successfully constructed a neural-based distinguisher, outperforming existing cryptanalysis on a version of the widely examined NSA block cipher Speck. This distinguisher could be incorporated into a broader key recovery attack scheme. He could perform an attack on 11 rounds of Speck with the help of the AI-based distinguishers. Subsequently, numerous other scholarly works have been published on the application and examination of AI and deep learning-based distinguishers for cryptanalysis, following Gohr's initial contribution.

Jaewoo So presents a novel approach to cryptanalysis in [20] wherein a generic model is established using deep learning (DL) to discover the key of block ciphers through analyzing known plaintext-ciphertext pairs. The author illustrates the effectiveness of the DL-based cryptanalysis model through successful attacks on lightweight block ciphers, including simplified DES, Simon, and Speck. The experimental outcomes suggest that DL-based cryptanalysis is capable of accurately retrieving key bits when the keyspace is limited to 64 ASCII characters. Baksi et. al in [2] describe two innovative approaches that utilize machine learning to identify distinguishers in symmetric key primitives. The authors demonstrate that their techniques can significantly reduce the complexity of differential cryptanalysis for round-reduced ciphers, resulting in an approximate cube root reduction in the claimed complexity. Through experiments on various non-Markov ciphers, the authors demonstrate the efficacy of their methods. The researchers also evaluate the selection of machine learning models and illustrate that even a shallow three-layer neural network can perform effectively for their purposes. This study serves as a proof of concept for how machine learning may be utilized as a comprehensive tool in symmetric key cryptanalysis.

Another research direction in AI-assisted cryptanalysis involves the interpretation of neural network distinguishers. Benamira et al. [6] provided a comprehensive analysis of a neural distinguisher proposed by Gohr. They analyzed classified sets of data to identify patterns and gain a better understanding of Gohr's results. Their findings revealed that the neural distinguisher primarily depends on differential distribution in ciphertext pairs, as well as differential distribution in the penultimate and antepenultimate rounds. The researchers subsequently developed a distinguisher for the Speck cipher, independent of any neural network use, which matched the accuracy and efficiency levels of Gohr's neural-based distinguisher. Furthermore, the researchers developed a machine learning-based distinguisher that utilized standard machine learning tools to approximate the Differential Distribution Table (DDT) of the cipher, similar to Gohr's neural distinguisher. This allowed for full interpretability of the distinguisher and contributed towards the interpretability of deep neural networks.

In [5], researchers presented a novel tool for neural cryptanalysis that comprises two components. Firstly, an evolutionary algorithm is proposed for the search of single-key and related-key input differences that are effective with neural distinguishers, thereby enabling the search for larger ciphers while eliminating the dependence on machine learning and prioritizing cryptanalytic methods. Secondly, DBitNet, a neural distinguisher architecture independent of the cipher structure, is introduced and demonstrated to outperform current state-of-the-art architectures. Using their tool, the researchers improved upon the state-of-the-art neural distinguishers for various ciphers and provided new neural distinguishers for others. The paper also provides a comparative review of the current state-of-the-art in neural cryptanalysis.

1.2 Our Contribution

In this paper, we present several contributions to the rotational-XOR cryptanalysis of AND-RX block ciphers such as Simon and Simeck. Our research advances the understanding of these ciphers and their resistance to attacks by incorporating deep learning techniques. Our main contributions are as follows:

1. We propose the first study of neural-assisted RX cryptanalysis. We modified the evolutionary algorithm presented in the work of Bellini et al. [5] to determine the optimal values for the translation parameter, denoted as δ, and the rotation offset parameter, represented as γ, for RX pairs, and by doing so we were able to find new RX distinguishers for Simon and Simeck ciphers
2. Our research successfully identifies RX distinguishers using deep learning techniques for different versions of Simon and Simeck in the related-key scenario, as opposed to the traditional weak-key model.
3. Our work contributes to finding the best parameters for diffusion layer for Simeck-like ciphers. The practical implications of these findings offer insights for designing secure AND-RX block ciphers and improving their resistance to RX cryptanalysis.

1.3 Outline

The current paper's structure is outlined as follows. The background concepts relevant to AND-RX ciphers, RX cryptanalysis, and deep learning-based cryptanalysis are discussed in Section 2. The methodology employed, which includes a modified evolutionary algorithm, is presented in Section 3. In Section 4, we report new distinguishes that have been discovered for Simon and Simeck. Section 5 proposes a technique for identifying the optimal permutation parameters for the diffusion layer of AND-RX block ciphers against DL-based attacks. Finally, Section 6 provides a concluding remark for the paper.

2 Preliminaries

In this section, we provide an overview of the key concepts and terms related to AND-RX ciphers, Rotational-XOR (RX) cryptanalysis, and deep learning techniques for cryptanalysis. Understanding these foundational concepts is essential for comprehending the methods and results presented in this paper.

2.1 AND-RX ciphers

Simon and Simeck are block ciphers intended for use in environments with limited resources, such as IoT devices. They utilize the AND-RX design paradigm, which employs only three basic operations: bitwise XOR ($\oplus$), bitwise AND ($\wedge$), and left circular shift ($\lll i$) by i bits. The general round function, R, of AND-RX ciphers can be defined by the following equation:

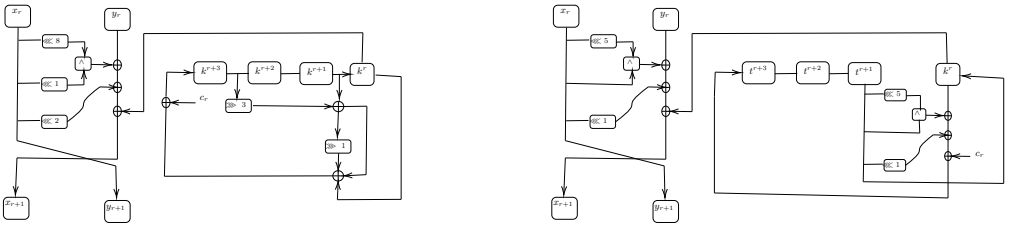
$$R(x, y) = (y \oplus f(x) \oplus k, x),$$

where $f(x) = ((x \lll a) \wedge (x \lll b)) \oplus x \lll c$ and k is the subkey for corresponding round.

In 2013, the NSA designed a family of lightweight block ciphers called Simon [4]. Each cipher in the family employs a word size of n bits, represented as Simon2n where $n \in \{16, 24, 32, 48\}$. Simon2n with a key size of $m \in 2, 3, 4$ words (mn bits) is denoted

as Simon$2n/mn$. For example, Simon32/64 operates on 32-bit plaintext blocks and utilizes a 64-bit key. In this paper, we focus on Simon$2n/4n$. The $f(x)$ function for Simon$2n$ encryption is $f(x) = ((x \lll 1) \wedge (x \lll 8)) \oplus x \lll 2$

Simon's key schedule produces r key words $k_0, \ldots, k_{r-1}$ from a given key, where r is the number of rounds. This process also involves using a sequence of 1-bit round constants to remove slide properties and circular shift symmetries.



Simon cipher for $m = 4$ Simeck cipher

Fig. 1: The Simon and Simeck ciphers

In this paper, we assess another lightweight block cipher known as Simeck [22]. It is represented by Simeck$2n/mn$, where the word size n must be either 16, 24, or 32, and $2n$ is the block size, while mn represents the key size.

The round function R used in Simeck is identical to the one used in the Simon cipher, as shown by the equation. However, Simeck's function f is distinct from Simon's and is defined as

$$f(x) = (x \wedge (x \lll 5)) \oplus (x \lll 1).$$

In Simeck cipher, the round key k_i is generated from a given master key K by first dividing the master key K into four words and using them as the initial states (t_2, t_1, t_0, k_0) of a feedback shift register. To produce round keys and update the registers, the round function f is utilized as well as a 1-bit round constant c_r. The number of rounds r for Simeck32/64, Simeck48/96, and Simeck64/128 are 32, 36, and 44, respectively. Figure 1 demonstrate the round function and key schedule of Simon and Simeck ciphers.

2.2 Rotational-XOR (RX) Cryptanalysis

Rotational cryptanalysis is a technique used to analyze the security of symmetric algorithms. Khovratovich and Nikolić introduced and formalized this approach for ARX structures in their work cited as [13], and subsequently applied it to scrutinize other ciphers like Skein [14]. In this technique, the attacker focuses on rotational pairs of plaintext and ciphertext, where the input values are related through a fixed rotation. The attacker then looks for statistical biases or patterns in the ciphertexts of these pairs that can be exploited to recover the secret key.

However, rotational cryptanalysis can be less effective in the presence of constants, as these fixed values can disrupt the rotational properties of the pairs. This is because the rotation operation alone does not account for the XOR operations that involve these constants, which can be present in many cryptographic algorithms. When constants are involved, the rotational relations between the input and output values might be obscured, making it harder to analyze the cipher using traditional rotational cryptanalysis techniques.

To address the limitations of traditional rotational cryptanalysis, Rotational-XOR cryptanalysis has been introduced as an extension. Ashur and Liu [1] have developed this technique to account for the XOR operations with constants that are commonly present in cryptographic algorithms. Rotational-XOR pairs and Rotational-XOR differences are defined to provide a more comprehensive framework for analyzing cryptographic primitives that involve both rotation and XOR operations with constants. The following are the definitions for RX pairs, RX difference, and RX cryptanalysis, respectively

Definition 1 (Rotational XOR Pair [1]). *An RX-pair is a rotational pair with rotational offset γ under translations δ_1 and δ_2, defined as the pair $x_0 \oplus \delta_1, (x_0 \lll \gamma) \oplus \delta_2$. However, for the sake of simplicity, a slightly different notation is used, where an RX-pair is represented by x_0 and $x_1 = (x_0 \lll \gamma) \oplus \delta$, or alternatively as $x_0, (x_0 \lll \gamma) \oplus \delta$, where $\delta = \delta_1 \oplus \delta_2$.*

Definition 2 (Rotational XOR Difference [1]). *An RX-difference of x_0 and x_1, denoted by $\Delta_\gamma(x_0, x_1)$, is formed by the rotational XOR of x_0 with a constant δ such that $x_1 = (x_0 \lll \gamma) \oplus \delta$, where $0 < \gamma < n$ and $\delta \in F_n^2$ is a constant. In another word*

$$\Delta_\gamma(x_0, x_1) = x_1 \oplus (x_0 \lll \gamma)$$

Definition 3 (Rotational XOR Cryptanalysis [1]). *Rotational XOR Cryptanalysis is a cryptanalytic technique that extends traditional rotational cryptanalysis to handle XOR operations with non-rotational-invariant constants between input and output pairs of a cryptographic primitive. This method aims to estimate the transition probability with respect to non-linear operations in block ciphers (like modular addition or $\wedge$ operation) of two input RX-differences to an output RX-difference. The technique introduces the concept of a (δ, γ)-Rotational-XOR-difference (or RX-difference), which represents a rotational pair with rotation γ under translation δ, i.e., $(x, (x \lll \gamma) \oplus \delta)$. The method seeks to analyze the propagation of RX-differences through the cryptographic primitive.*

The transmission of RX-differences through linear operations is known to be deterministic; however, this is not the case for nonlinear operations. Prior research conducted by Ashur et al. [1] and Lu et al. [18] delved into the investigation of the transmission of RX-differences through modular addition and AND ($\wedge$) operations, respectively.

2.3 Deep Learning and its Application on Symmetric Cryptography

Deep learning has proven to be a game-changer in various challenging tasks, including image recognition, natural language processing, and speech recognition, to name a few.

Although machine learning techniques have been applied to cryptography, much of the practical work has focused on side-channel analysis [12, 20, 23]. However, in 2019, Gohr explores the application of deep learning techniques for cryptanalysis [10], specifically for attacking the Speck [4] cipher. This approach aims to differentiate between real and random pairs of ciphertexts resulting from the encryption of plaintext pairs with fixed and arbitrary input differences, respectively. While pure differential distinguishers have traditionally been used for this purpose, his research has shown that deep learning (DL) can outperform their traditional counterparts. Gohr's study focused on Speck-32/64 and compared the accuracy of a pure differential distinguisher with a DL-based distinguisher for 5 to 8 rounds. The results demonstrated that the DL-based distinguisher achieved higher accuracy than the pure differential distinguisher, highlighting the potential of DL-based approaches in differential cryptanalysis.

Algorithm 1 is the algorithm employed by Gohr in training a deep learning (DL)-based distinguisher for differential attack. The algorithm considers a pair of plaintexts P_0 and P_1 with a predetermined input difference Δ, i.e., $P_0 \oplus P_1 = \Delta$. Additionally, $C_0 = E_k(P_0)$ and $C_1 = E_k(P_1)$, where E_k signifies encryption of plaintext P with key k. Furthermore, in the context of Feistel structured block ciphers, the left and right halves of a data block are typically referred to as L and R, respectively.

Algorithm 1 DL-based Differential Distinguisher for r rounds of Speck32/64

Input: r (number of rounds), AI machine, (C_0, C_1)
Output: Trained AI machine, differential distinguisher status
Generate 10^7 plaintext pairs (P_0, P_1) with $\Delta = (L_0 \oplus L_1, R_0 \oplus R_1) = (0x0040, 0x0000)$
Randomly allocate 10^7 labels $Y \in_r \{0, 1\}$ to the pairs
for each pair (P_0, P_1) with label Y **do**
    if $Y = 0$ **then**
        $P_1 \leftarrow P_1 \in_r \{0, 1\}^{32}$
    Encrypt the pairs with r rounds of Speck32/64 to get ciphertext pairs (C_0, C_1)
    Store (C_0, C_1) with corresponding labels in a dataset
Train DL-distinguisher using the dataset and their corresponding labels
Repeat steps 3-11 for another 10^6 pairs for testing
Measure the accuracy of the DL-based distinguisher
if accuracy $> 50\%$ **then**
    The machine is a DL-based differential distinguisher

In deep learning (DL)-based distinguishers, determining the optimal input difference can significantly improve their performance. Gohr [10] presented a novel algorithm for identifying appropriate input differences for neural network distinguishers, without requiring prior human knowledge. This algorithm employs few-shot learning, where a neural network learns features from a large dataset and a simpler machine learning algorithm is trained on a smaller set of samples.

In [5], Bellini et al. presented an alternative approach that does not rely on neural networks for finding the best input difference for DL-based distinguishers. In order to find the best input difference they had a bias score hypothesis which states that the optimal input difference for neural distinguishers cryptographically is the input

difference that maximizes the bias of output difference bits. Computing the bias score for block ciphers is infeasible due to the requirement of enumerating all keys and plaintexts. However, we can use an approximation derived from a limited number of samples t:

Definition 4 (Approximate Bias Score [5]). *Let $E : \mathbb{F}_2^n \times \mathbb{F}_2^k \to \mathbb{F}_2^n$ be a block cipher, and let $\Delta \in \mathbb{F}_2^n$ be an input difference. The Approximate Bias Score for Δ, denoted by $\tilde{b}^t(\Delta)$, is defined as the sum of the biases of each bit position j in the output difference, computed over t samples. Formally, we have:*

$$\tilde{b}^t(\Delta) = \left| \sum_{j=0}^{n-1} 2 \cdot \frac{\sum_{i=0}^{t} (E_{K_i}(X_i) \oplus E_{K_i}(X_i \oplus \Delta))_j}{t} - 1 \right|$$

The authors' hypothesis is confirmed, and they propose an evolutionary-based search algorithm that leverages the approximate bias score to explore a larger set of candidate input differences. The algorithm starts with a population of randomly generated input differences, and an approximate bias score is computed for each of them. The top 32 input differences with the highest score are retained for further evaluation. The algorithm then proceeds with 50 iterations, during which new individuals are derived and evaluated. To ensure the starting round's influence on the bias score is accounted for, the number of rounds is incremented if the maximum bias score obtained surpasses a threshold limit. At the end of the algorithm, the authors obtain a list of 32 input differences for each round. The final step involves computing a weighted cumulative bias score for all the obtained input differences from round 1 to round R. The authors' search algorithm based on the biased score demonstrates improved performance in identifying input differences compared to other methods and can be useful for cryptographic applications. Algorithm 2 can show their method. The algorithm has several key parameters, including the initial population size for each generation (P), the mutation probability (p_m), the approximate bias score sample size (t), and the relevance threshold (T). The specific values of these parameters can be adjusted as needed to optimize the algorithm's performance. Also $curr\_population_i$ indicates the ith bit of the current population we have.

3 Identification of Optimal RX Distinguishers in Cryptanalysis with Evolutionary Algorithm

In this section, we present a modified evolutionary algorithm that builds upon the algorithm of [5]. Our algorithm facilitates the discovery of novel RX differential pairs that can be leveraged to train deep-learning-based RX distinguishers. Initially, we discuss the artificial intelligence (AI) tools and deep learning model utilized in our study. Subsequently, we explore the relation between the rotational bias score and the accuracy of deep learning-based RX distinguishers. Drawing upon this insight, we introduce our evolutionary algorithm designed to identify the optimal RX input for training a deep learning-based distinguisher.

Algorithm 2 Evolutionary optimizer [5]

init_population $\leftarrow$ [RandomInt$(0, 2^n - 1)$ for 1024 times]
Sort *init_population* by $\tilde{b}_t(\cdot)$ in descending order
curr_population $\leftarrow$ first P elements of *init_population*
for *iter* $\leftarrow 0$ to 50 **do**
    cand $\leftarrow$ [ ]
    for $i \leftarrow 0$ to $P - 1$ **do**
        for $j \leftarrow i + 1$ to $P - 1$ **do**
            if RandomFloat$(0, 1) < p_m$ **then**
                $m \leftarrow 1$
            else
                $m \leftarrow 0$
            Add $curr\_population_i \oplus curr\_population_j \oplus (m \lll \text{RandomInt}(0, n-1))$ to *cand*
    Sort *cand* by $\tilde{b}_t(\cdot)$ in descending order
    curr_population $\leftarrow$ first P elements of *cand*
return *cand*

3.1 AI Tools and Model Development

In this study, we employ the Keras [9] library to develop our deep learning model, which is inspired by the architecture proposed by Aron Gohr in his groundbreaking CRYPTO'19 paper [10]. Gohr's model focuses on using a neural network to differentiate between pairs of Speck32/64 ciphertexts corresponding to fixed differences (non-random) and random message pairs (random). His neural distinguisher is a residual network comprising four main components, achieving remarkable accuracy for varying rounds of Speck32/64 and enabling practical key recovery attacks.

The input to Gohr's neural distinguisher consists of a 64-bit ciphertext pair from Speck32/64, which is reshaped and permuted into a 16-bit wide tensor with four channels. This input reshaping takes into account the unique 16-bit word structure of Speck32/64. The second component of the architecture involves a one-dimensional convolution, denoted as Conv1D with kernel size 1 and 32 filters, that slices through the four-channel bits.

Following the convolutional layer, batch normalization and ReLU activation function are applied as per conventional deep learning practices. The third component consists of residual blocks, with each block containing two convolutional layers, represented as Conv1D with kernel size 3 and 32 filters. The number of residual blocks in the network determines the depth of the neural distinguisher.

Lastly, a densely connected prediction head with ReLU activations is employed, along with an output layer featuring a single neuron with sigmoid activation. L2 regularization with a value of 10^{-5} is used throughout the network to penalize large weights and reduce the likelihood of overfitting. Also, the Adam optimization method [15] is used for this architecture.

The present research employs Gohr's neural distinguisher architecture to train RX differential distinguishers for analyzing AND-RX ciphers such as Simon and Simeck. The rationale behind this choice is that Simon and Simeck, specifically the 32-bit version, share the same 16-bit structure as Speck32/64, which is extensively investigated

in Gohr's previous research. The aim of this approach is to harness the potential of deep learning to discover new RX distinguishers and enhance our comprehension of the security of these ciphers.

We also employed a sequential training approach in which we increased the number of rounds in each iteration. Previous research has shown that this approach can improve model performance [5], as it utilizes knowledge learned in previous rounds. We trained our model using a dataset of 10^7 training samples and 10^6 validation samples, with a batch size of 1000. Our model had a depth of 1 and we used 5 epochs for training.

3.2 Training DL-based RX distinguishers

This section outlines the training process for our RX distinguishers, which is based on deep learning. Furthermore, it elaborates on the potential of DL-based RX distinguishers in gaining insights from pairs of ciphertext.

Training phase. The training process involves data preparation, model configuration, and evaluation of the trained model on AND-RX ciphers such as Simon and Simeck. The first step in training the RX distinguishers is data preparation. We generate a dataset consisting of pairs RX of ciphertexts, labeled as either non-random (for fixed (δ, γ)) or random (random message pairs). Since RX cryptanalysis is a related-key cryptanalysis, in addition to a translation δ that exists for the RX plaintext pairs, there may also be a translation for the keys used to encrypt each plaintext within the pair, which can be shown by δ_{key}. According to the research presented in [18], the propagation of the RX differential in AND-RX ciphers primarily depends on the Hamming weight of the difference and their rotations.

In this part, we chose to focus on an input difference zero for the initial training of the deep learning-based RX distinguisher for lightweight ciphers Simon32/64 and Simeck32/64. Specifically, we set both the input difference and key RX difference to $0x0000$ and $0x00000000$, respectively, using hexadecimal representation to represent the n-bit ($n \times m$-bit) binary representation of plaintexts (keys). By doing so, we aimed to first train the RX distinguisher as a starting point and subsequently analyze its behavior and performance in the context of Simon32/64 and Simeck32/64 ciphers, and then investigate possible improvements.

For that, we fixed $\delta = 0$ for both plaintexts and keys and iterated through all possible γs to determine the γ that would produce the best distinguisher for the longest number of rounds r for both Simon and Simeck. Our results indicated that for the case of Simeck32/64, $\gamma \in \{1, 15\}$ and for the case of Simon32/64, $\gamma \in \{4, 12\}$ produced the best distinguishers for 14 and 10 rounds, respectively. The detailed training method for r rounds is presented in Algorithm 3.

Machine Interpretation. In this part, we present an interpretation of the deep learning-based RX distinguishers trained on AND-RX ciphers, specially on Simeck32/64. Our objective is to investigate the factors affecting the accuracy of the distinguisher and its ability to analyze RX cryptanalysis.

Algorithm 3 DL-based RX Distinguisher for r rounds of a Cryptographic Primitive

Input: r (number of rounds), AI machine, (C_0, C_1)
Output: Trained AI machine, RX distinguisher status
Choose a rotational offset γ and constants δ_1, δ_2, with $\delta_\gamma = \delta_1 \oplus \delta_2$
Generate 10^7 plaintext pairs (P_0, P_1) with RX-difference $\delta_\gamma(P_0, P_1) = (L_1 \oplus (L_0 \lll \gamma), R_1 \oplus (R_0 \lll \gamma))$
Randomly allocate 10^7 labels $Y \in_r \{0, 1\}$ to the pairs
for each pair (P_0, P_1) with label Y **do**
    if $Y = 0$ **then**
        $P_1 \in_r \{0, 1\}^{32}$
    Encrypt the pairs with r rounds of the cryptographic primitive to get ciphertext pairs (C_0, C_1)
    Store (C_0, C_1) with corresponding labels in a dataset
Train DL-RX-distinguisher using the dataset and their corresponding labels
Repeat steps 3-11 for another 10^6 pairs for testing
Measure the accuracy of the DL-based RX distinguisher
if accuracy $\geq 50\%$ **then**
    The machine is a DL-based RX distinguisher

In prior studies on deep learning-based differential distinguishers, it has been shown that the bias of the output differential plays a crucial role in determining the accuracy and the number of rounds for which a distinguisher can be trained. To investigate the possible impact of bias on our RX distinguishers, we first define the Approximate Bias Score for RX attack, inspired by the definition presented in [5].

Definition 5 (Approximate RX Bias Score). *Let $E : \mathbb{F}_2^n \times \mathbb{F}_2^k \to \mathbb{F}_2^n$ be a block cipher, and let $\delta \in \mathbb{F}_2^n$ be an input RX-difference with a given rotational offset γ. The Approximate RX Bias Score for δ, denoted by $\tilde{b}^t(\delta, \gamma)$, is defined as the sum of the biases of each bit position j in the output RX-difference, computed over t samples. Formally, we have:*

$$\tilde{b}^t(\delta, \gamma) = \left| \sum_{j=0}^{n-1} 2 \cdot \frac{\sum_{i=0}^{t} ((E_{K_i}(X_i)) \oplus E_{(K_i \lll \gamma) \oplus \delta}((X_i \lll \gamma) \oplus \delta)))_j}{t} - 1 \right|$$

In our study, we investigated the relationship between bias score, accuracy of the RX distinguisher, and number of rounds for the trained distinguisher.

To do this, we trained distinguishers for a range of γ values while keeping δ fixed at 0. For each distinguisher, we then calculated its bias score and accuracy across various rounds. This process was repeated for multiple iterations to ensure the robustness of our results. the Pearson correlation coefficient and The resulting scatter plot (Figure 2) maps bias scores (x-axis) against distinguisher accuracy (y-axis), with each dot representing a different γ value for Simeck32/64. Different colors are used to denote the number of rounds for each distinguisher: red for 11 rounds, green for 12 rounds, and blue for 13 rounds.

Upon examining the plot, we observed a general trend: higher bias scores often corresponded to higher accuracy (or more rounds). There was one notable exception:

$\gamma = 11$ produced the highest bias, but no distinguisher could be trained above 12 rounds with this γ value.

In order to gain a deeper understanding of these patterns, we conducted a weighted correlation analysis. This analysis validated the presence of a positive connection between the bias score and accuracy. During our experiment, we utilized formula (1) for our analysis, and the resulting score was approximately 0.65. This score indicates that there is indeed a positive correlation between the bias and the accuracy of the trained machine. In the formula, ρ represents the Pearson correlation coefficient, and a represents the accuracy of the DL-based distinguisher. We further scrutinized outliers, such as the aforementioned $\gamma = 11$ case, and hypothesize that these may be due to variations in the cipher structure or other unidentified factors that warrant further investigation.

$$\text{correlation\_coefficient} = \rho\left(\tilde{b}^t, a \times e^{2r}\right) \tag{1}$$

These results underpin our claim that the output RX difference bias score is a key determinant of the accuracy of a deep learning-based RX distinguisher. They also prompted us to introduce an adapted evolutionary algorithm aimed at identifying optimal (δ, γ) pairs for RX plaintext.

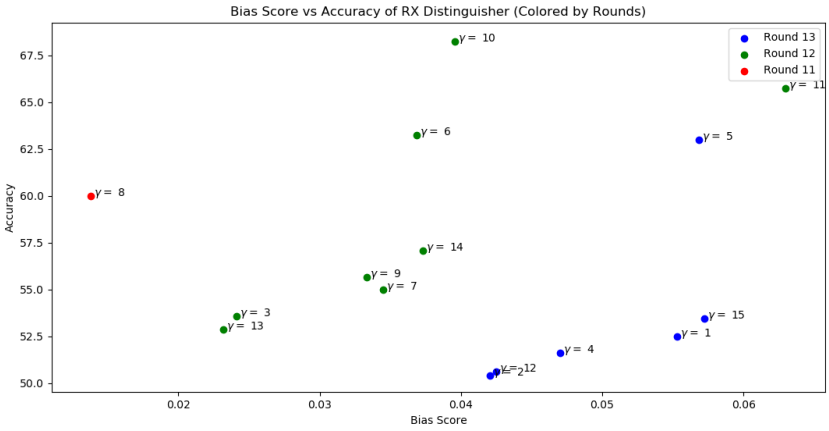


Fig. 2: Scatter plot of Bias Score vs Accuracy of RX Distinguisher (Colored by Rounds)

3.3 Evolutionary Optimization of Deep Learning RX Differential Distinguishers

Now, we propose an evolutionary optimization algorithm for finding the best RX input differences. The goal is to adapt the evolutionary-based search algorithm, leveraging the approximate RX bias score, to explore a more extensive set of candidate RX pairs.

The modified algorithm will account for the rotational offset γ and the XOR translation δ.

The main modification of this algorithm involves introducing a novel search strategy for the optimal shift parameter, γ, while also evaluating the impact of input difference. Notably, to the best of our knowledge, this approach is the first to simultaneously search for both the optimal δ and γ parameters, instead of solely searching for the best δ for a fixed γ value, which is commonly set to $\gamma = 1$ in the literature. To enable this search strategy, we have developed a methodology for generating a binary representation for the shift parameter based on the block size, with the final bits appended to each member of the population. For example, for the Simeck32/64, γ and γ_{key} represented by two 4-bit words. So, we increase the number of bits in the search by an additional 8, where the final 8 bits represent the value of γ.

The new algorithm starts with a population of randomly generated input differences and corresponding rotational offsets. For each of them, an approximate RX bias score is computed. The top 32 input differences with the highest score are retained for further evaluation. The algorithm proceeds with 50 iterations, during which new individuals are derived and evaluated. If the highest bias score returned is greater than a threshold, the number of rounds is incremented by one. At the end of the algorithm, a list of 32 input differences for each round is obtained. The final step involves computing a weighted cumulative RX bias score for all the obtained input differences from round 1 to round R.

Algorithm 4 shows the modified optimizer for RX attack. The algorithm has several key parameters, including the initial population size for each generation (P), the mutation probability (p_m), the approximate RX bias score sample size (t), and the threshold (T). The specific values of these parameters can be adjusted as needed to enhance the algorithm's performance. Also, $curr\_population_{i,\delta}$ and $curr\_population_{i,\gamma}$ indicate the value of ith bit of δ and γ, respectively.

Algorithm 4 Evolutionary optimizer for RX differential distinguishers

init_population $\leftarrow$ [RandomInt$(0, 2^n - 1)$||RandomInt$(1, n-1)$ for 1024 times]
Sort *init_population* by $\tilde{b}^t_{(\delta,\gamma)}(\cdot)$ in descending order
curr_population $\leftarrow$ first P elements of *init_population*
for *iter* $\leftarrow 0$ to 50 **do**
    cand $\leftarrow$ [ ]
    for $i \leftarrow 0$ to $P-1$ **do**
        for $j \leftarrow i+1$ to $P-1$ **do**
            $m_\gamma \leftarrow 1$
            if RandomFloat$(0,1) < p_m$ **then**
                $m_\delta \leftarrow 1$
            else
                $m_\delta \leftarrow 0$
            Add $(((curr\_population_{i,\delta} \lll \text{RandomInt}(0, n-1)) \oplus curr\_population_{j,\delta} \oplus m_\delta) || (curr\_population_{i,\gamma} \oplus m_\gamma)$ to *cand*
    Sort *cand* by $\tilde{b}_t(\cdot)$ in descending order
    curr_population $\leftarrow$ first P elements of *cand*
return *cand*

The modified algorithm presented in Algorithm 4 is specifically designed for optimizing RX differential distinguishers for cryptographic primitives such as Simon32/64 and Simeck32/64. By incorporating the rotational offset γ and the XOR translation δ, the search space for potential input differences is expanded, increasing the likelihood of discovering more effective RX input pairs for deep learning-based RX distinguishers. One significant improvement afforded by this method is the capability to identify effective RX distinguishers for full-key classes, which represents a marked advancement over prior research that only succeeded in identifying such distinguishers for classes that were not full-key.

4 Results and Discussion

In this section, we present the results of applying our evolutionary optimization method to different versions of Simon and Simeck block ciphers. Our goal is to determine the most effective RX input differences (δ) and rotational offsets (γ) for each cipher version, allowing us to train deep learning-based RX distinguishers.

In this study, we utilized an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz to run the evolutionary algorithm and employed Colab [8] for training the distinguishers. The experiments for the highest weight version of Simon and Simeck lasted approximately 3 hours each.

We used an evolutionary optimization approach, applying Algorithm 4 to the various versions of Simon and Simeck. The optimization was run 10 times, with each trial randomly initializing the values of δ and γ. The evolutionary algorithm subsequently updated these values to achieve higher distinguisher accuracy.

Each trial was trained on a dataset of 10^6 cipher text pairs, which were generated using different random keys. The key space was varied over the course of the trials to ensure thorough testing.

In the following subsections, we discuss our findings for each cipher version, highlighting the specific δ and γ values that led to the most effective RX distinguishers.

4.1 Simeck Cipher

For the Simeck cipher family, our evolutionary optimization method was successful in identifying effective RX distinguishers for 15, 17, 20 rounds of Simeck32/64, Simeck48/96, and Simeck/128, respectively. The results for each version of the Simeck cipher are detailed in Table 3, which includes the optimal RX input differences (δ), rotational offsets (γ), the number of rounds, and the corresponding distinguisher accuracy.

Our proposed method for training deep learning-based RX distinguishers demonstrates notable advantages despite not necessarily achieving the best possible distinguisher performance for the Simeck cipher family. Although there exist distinguishers with higher round coverage, As shown in Table 2, such as 20 rounds for Simeck32/64, 27 rounds for Simeck48/96, and 34 rounds for Simeck64/128, these distinguishers operate under significantly smaller weak key classes, specifically of size 2^{30}, 2^{46}, and 2^{58}, respectively, while our distinguishers cover the entire key space.

Table 3: Summary of the optimal RX input differences (δ), key differences (δ_{key}), rotational offsets (γ), the number of rounds, and distinguisher accuracy for different versions of Simeck block ciphers.

Cipher Version	δ	δ_{key}	γ	Number of Rounds	Accuracy
Simeck32/64	$(0, 0x0002)$	0002	1	15	51.34
				14	57.08
				13	70.57
Simeck48/96	$(0, 0x000002)$	0002	1	17	52.06
				16	57.67
				15	69.85
Simeck64/128	$(0, 0x00000002)$	0002	1	20	52.12
				19	57.01
				18	70.15

In our investigation, it was found that the Simeck cipher is more susceptible to RX cryptanalysis compared to Simon. While searching for vulnerabilities in Simon, we did not find any effective deep learning-based distinguishers with $\gamma \neq 0$ that perform better than conventional DL-differential distinguishers. However, this is not the case for Simeck, as shown by the results presented in Table 1, where DL-RX distinguishers can almost match the performance compared to related key distinguishers reported in the literature for the same round.

4.2 Simon Cipher

The results of applying our proposed method to the Simon cipher family, specifically Simon32/64, Simon64/128, and Simon128/256 is shown in Table 4. We obtained deep learning-based RX distinguishers for 11 rounds for Simon32/64, 13 rounds for Simon64/128, and 16 rounds for Simon128/256, respectively. It should be noted that these results exhibit worse performance in terms of round coverage when compared to existing distinguishers from the literature that also cover the full key space.

Additionally, we introduce RX distinguishers with rotational offsets γ other than 1, which, to the best of our knowledge, has not been previously explored. This highlights the potential of our proposed method to uncover new insights in the realm of RX cryptanalysis and contribute to the development of more secure cryptographic primitives.

In our study, we observed that the performance of the proposed method was worse for the Simon cipher family compared to the Simeck cipher family, even though both ciphers share the AND-RX design paradigm. One possible explanation could be the different structures of the diffusion layer in the round functions of Simon and Simeck ciphers. The different choices of shift parameters in these functions could result in different resistance to RX cryptanalysis. The rotation offsets in the $f(x)$ functions might interact differently with the proposed distinguishers, making it harder for the evolutionary search algorithm to find strong RX distinguishers for Simon compared to Simeck.

Table 4: Summary of the optimal RX input differences (δ), key differences (δ_k), rotational offsets (γ), the number of rounds, and distinguisher accuracy for different versions of Simon block ciphers.

Cipher Version	δ	δ_k	γ	Number of Rounds	Accuracy
Simon32/64	$(0x0, 0x0002)$	0002	3	11	54.45
				10	74.11
				9	98.48
Simon64/128	$(0x0, 0x0)$	0000	30	13	51.51
				12	73.15
				11	98.5
Simon128/256	$(0x0, 0x0)$	0000	60	16	50.62
				15	72.26
				14	96.87

5 Impact of the Diffusion Layer and Optimal Rotation Parameters

In the design of AND-RX ciphers, the choice of round constants and shift parameters are crucial in improving the security against RX cryptanalysis. While Lu et al. investigated the impact of round constants on RX cryptanalysis [18], the present study aims to extend this line of inquiry by examining the influence of shift parameters in AND-RX ciphers. Our primary focus is on identifying the ideal rotation parameters, (a, b, c) for $f(x)$ function of AND-RX ciphers that can be defined as below:

$$f(x) = ((x \lll a) \wedge (x \lll b)) \oplus x \lll c$$

In this section, our exploration of optimal parameters is centered on AND-RX ciphers with non-linear key schedules. The rationale behind this choice is based on our observation that Simon outperforms Simeck in resisting RX cryptanalysis. Consequently, our aim is to ascertain whether it is feasible to devise a variant of the Simeck-like cipher that could rival Simon in its defense against deep learning-based RX and differential attacks, or find other parameters that can enhance the security of Simeck-like ciphers.

In our pursuit of the optimal rotation parameters, we employed our previously discussed evolutionary algorithm (see Algorithm 4). This involved an iterative process where we tested various combinations of a, b, and c parameters for Simeck32/64. For each combination, we identified the highest bias score and the maximum number of rounds for which our algorithm could determine an appropriate input for the DL distinguisher. Notably, as our algorithm effectively searches for the best inputs with any γ values, even $\gamma = 0$, the optimal shift parameters identified also enhance resistance against both differential and RX differential attacks.

Among the shift sets found during our comprehensive exploration, (4, 6, 3) stood out due to its superior cumulative bias score, indicating enhanced resistance to these types of attacks. We could not find any distinguisher for more than 13 round for a Simeck-like cipher with these parameters as their shift parameter based on our optimizer.

Notably, for other optimal parameter sets, we were successful in training DL-based distinguishers for up to 14 rounds. These findings, including the six shift parameter sets that performed optimally in our experiments, are detailed in Table 5.

Table 5: Optimal rotation sets for AND-RX ciphers with non-linear key schedule and $n = 32$ determined by the evolutionary algorithm

Rotation Set	**Highest Cumulative Bias**	**Highest Round Distinguisher**
(4, 6, 3)	14.32	13
(4, 5, 7)	17.28	14
(6, 7, 4)	17.99	14
(3, 7, 2)	18.25	14
(3, 5, 6)	18.67	14
(3, 6, 1)	18.95	14

Our findings provide useful considerations for the design of block ciphers. The optimal rotation parameters we identified demonstrate how specific configuration choices can influence a cipher's behavior against RX and related-key differential attacks. It should be noted, however, that the parameters we found optimal for security may not directly apply to practical cipher design, as designers also have to consider other factors such as hardware efficiency. Therefore, our insights should be considered along with other factors like hardware efficiency during the selection of shift parameters in the development of AND-RX ciphers. This balanced approach could yield designs that optimize both security and efficiency, aligning with the core goals of ARX/AND-RX cipher designers.

6 Conclusion

This paper has investigated the application of deep learning techniques to the RX cryptanalysis of AND-RX block ciphers, with a particular focus on the Simon and Simeck families. We have uncovered distinguishers in the related-key model, which is opposed to the conventional weak-key models found for these ciphers.

Our deep learning models have shown a promising ability to identify effective RX distinguishers for various rounds of the Simeck cipher. Specifically, a combined accuracy score of 0.5475 was achieved for 15 rounds of Simeck32/64 and 0.6429 for 18 rounds of Simeck64/128. Moreover, optimal RX input differences, key differences, and rotational offsets for different versions of Simeck and Simon block ciphers were identified.

In addition, this study presented a novel approach to optimizing diffusion layers in AND-RX block ciphers. As a result, several optimal rotation sets for Simeck-like ciphers were identified.

References

1. Ashur, T., Liu, Y.: Rotational cryptanalysis in the presence of constants. IACR Transactions on Symmetric Cryptology pp. 57–70 (2016)

2. Baksi, A., Baksi, A.: Machine learning-assisted differential distinguishers for lightweight ciphers. Classical and Physical Security of Symmetric Key Cryptographic Algorithms pp. 141–162 (2022)
3. Bard, G.: Algebraic cryptanalysis. Springer Science & Business Media (2009)
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 (2013)
5. Bellini, E., Gerault, D., Hambitzer, A., Rossi, M.: A cipher-agnostic neural training pipeline with automated finding of good input differences. Cryptology ePrint Archive (2022)
6. Benamira, A., Gerault, D., Peyrin, T., Tan, Q.Q.: A deeper look at machine learning-based cryptanalysis. In: Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I 40. pp. 805–835. Springer (2021)
7. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. Journal of CRYPTOLOGY **4**, 3–72 (1991)
8. Bisong, E.: Google Colaboratory, pp. 59–64. Apress, Berkeley, CA (2019). https://doi.org/10.1007/978-1-4842-4470-8_7, https://doi.org/10.1007/978-1-4842-4470-8_7
9. Chollet, F.: Keras. https://github.com/fchollet/keras (2015)
10. 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, August 18–22, 2019, Proceedings, Part II 39. pp. 150–179. Springer (2019)
11. Gohr, A., Leander, G., Neumann, P.: An assessment of differential-neural distinguishers. Cryptology ePrint Archive (2022)
12. Hu, F., Wang, H., Wang, J.: Multi-leak deep-learning side-channel analysis. IEEE Access **10**, 22610–22621 (2022)
13. Khovratovich, D., Nikolić, I.: Rotational cryptanalysis of arx. In: Fast Software Encryption: 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers 17. pp. 333–346. Springer (2010)
14. Khovratovich, D., Nikolić, I., Rechberger, C.: Rotational rebound attacks on reduced skein. Journal of cryptology **27**, 452–479 (2014)
15. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Liu, Z., Li, Y., Wang, M.: Optimal differential trails in simon-like ciphers. IACR Transactions on Symmetric Cryptology pp. 358–379 (2017)
17. Lu, J., Liu, G., Sun, B., Li, C., Liu, L.: Improved (related-key) differential-based neural distinguishers for simon and simeck block ciphers. Cryptology ePrint Archive (2022)
18. Lu, J., Liu, Y., Ashur, T., Sun, B., Li, C.: Improved rotational-xor cryptanalysis of simon-like block ciphers. IET Information Security **16**(4), 282–300 (2022)
19. Matsui, M.: Linear cryptanalysis method for des cipher. In: Advances in Cryptology—EUROCRYPT'93: Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, May 23–27, 1993 Proceedings 12. pp. 386–397. Springer (1994)
20. So, J.: Deep learning-based cryptanalysis of lightweight block ciphers. Security and Communication Networks **2020**, 1–11 (2020)
21. Wang, X., Wu, B., Hou, L., Lin, D.: Automatic search for related-key differential trails in simon-like block ciphers based on milp. In: Information Security: 21st International Conference, ISC 2018, Guildford, UK, September 9–12, 2018, Proceedings 21. pp. 116–131. Springer (2018)
22. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Cryptographic Hardware and Embedded Systems–CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings. pp. 307–329. Springer (2015)
23. Zhang, L., Xing, X., Fan, J., Wang, Z., Wang, S.: Multilabel deep learning-based side-channel attack. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **40**(6), 1207–1216 (2020)