Improved Framework of Related-key Differential Neural Distinguisher and Applications to the Standard Ciphers

Ruitao Su[1], Jiongjiong Ren*[1], and Shaozhen Chen[1]

[1]Information Engineering University, ZhengZhou, P.R.China,

March 26, 2025

Abstract

In recent years, the integration of deep learning with differential cryptanalysis has led to differential neural cryptanalysis, enabling efficient data-driven security evaluation of modern cryptographic algorithms. Compared to traditional differential cryptanalysis, differential neural cryptanalysis enhances the efficiency and automation of the analysis by training neural networks to automatically extract statistical features from ciphertext pairs. As research advances, neural distinguisher construction faces challenges due to the absence of a unified framework capable of cross-algorithm generalization and feature optimization. There's no systematic way to build a framework from data formats and network architectures, which limits their scalability across diverse ciphers and and their suitability for combining different cryptanalysis methods. While neural network training is data-driven, we lack interpretable explanations for the quality of differentially generated datasets. Therefore, there is an urgent need to combine cryptographic theory with data analysis methods to systematically evaluate dataset quality.

This paper proposes a novel framework for constructing related-key neural differential distinguishers that integrates three core innovations: (1) multi-ciphertext multi-difference formats to enhance dataset diversity and feature coverage, (2) structural filtering for prioritizing high-probability differential paths aligned with cryptographic architectures, and (3) Deep Residual Shrinkage Network (DRSN) with adaptive thresholding to suppress noise and amplify critical differential features. By applying this framework to two standardized algorithms DES and PRESENT, our results demonstrate significant advancements. For DES, the framework achieves an 8-round related-key neural distinguisher and improves 6/7-round distinguisher accuracy by over 40%. For PRESENT, we construct the first 9-round related-key neural distinguisher, which outperforms existing neutral distinguishers in both round coverage and accuracy. Additionally, we employ kernel principal component analysis (KPCA) and K-means clustering to evaluate the quality of differential datasets for DES and PRESENT, revealing that clustering compactness strongly correlates with distinguisher performance. Furthermore, we propose a validation algorithm to verify differential combinations with cryptographic advantages from a machine learning perspective, identifying 'good' plaintext-key differential combinations. We apply

*Corresponding author. Email: jiongjiong_fun@163.com

this approach to the SIMECK algorithm, demonstrating its broad applicability. These findings validate the framework's effectiveness in bridging cryptographic analysis with data-driven feature extraction and offer new insights for automated security evaluation of block ciphers.

1 Introduction

Cryptography, the cornerstone of information security, enables critical functions including data encryption and integrity verification. Block ciphers—characterized by efficiency, implementability, and versatility—form the backbone of modern cryptographic systems. Their evolution originated in the 1970s alongside cryptographic standardization, marked by the 1977 establishment of DES(Data Encryption Standard)[1] , the first government-certified encryption algorithm that propelled cryptographic research globally. By 2000, the National Institute of Standards and Technology (NIST) adopted the Rijndael[2] algorithm as the Advanced Encryption Standard (AES), which has renowned for its robust security and computational efficiency. Recent demands from IoT and AI advancements have driven lightweight block cipher innovation for resource-constrained environments. The PRESENT[3] algorithm exemplifies this trend through its ultra-efficient design, achieving ISO/IEC 29192-2 standardization and widespread deployment in embedded/IoT systems. The standardization of cryptographic algorithms ensures their security and verifiability, rendering the cryptanalysis of such standardized ciphers critically important.

Deep learning[4, 5] models have demonstrated remarkable potential in cryptanalysis, primarily due to their ability to extract discriminative features from ciphertext pairs associated with input differentials. In 2019, a groundbreaking study by Gohr [6] presented at CRYPTO achieved the first theoretical integration of deep learning with traditional differential cryptanalysis. Targeting the lightweight block cipher SPECK32/64, this work designed a deep residual network (ResNet)[7] architecture to successfully construct an 8-round neural differential distinguisher, which demonstrated superior accuracy compared to conventional differential distinguishers at the same round count.

By leveraging deep neural networks for feature learning and pattern recognition on ciphertext data, deep learning models can precisely identify latent patterns in differential propagation, thereby constructing comprehensive differential path characteristic libraries. This data-driven approach to feature extraction enables deep learning techniques to achieve superior distinguishing performance compared to traditional analytical methods. Research findings demonstrate that integrating deep learning into differential cryptanalysis holds substantial theoretical and practical significance. Not only does it provide a novel technical pathway for cryptographic evaluation, but it also opens new research directions, potentially driving transformative innovations in cryptanalysis. Similar to classical differential cryptanalysis, deep learning-based differential analysis emphasizes the construction of neural distinguishers, where higher accuracy, extended round coverage, and reduced key recovery complexity directly correlate with improved cryptanalytic outcomes. Current research in this domain is categorized into three primary directions:

Training Data Optimization for High-Accuracy Distinguishers: Recent advancements in neural distinguishers have emphasized the critical role of training data optimization. Chen et al.[8] demonstrated that utilizing multiple ciphertext pairs as training samples significantly enhances the accuracy of 5–7-round neural distinguishers

for SPECK32/64. Building on this, Hou et al.[9] achieved further improvements for SPECK and SIMON by training networks on multi-pair output differentials, while Lu et al.[10] pioneered single-key and related-key neural distinguishers for SIMON through early-round output inference, attaining unprecedented precision. Bao et al.[11] improved SIMON32/64 distinguishers by incorporating linear combinations of previous-round outputs as network inputs. Wang et al. [12] improved the accuracy of neural distinguishers for SPECK and SIMON by replacing random differentials with fixed ones to generate negative samples, based on differential cryptanalysis and sample feature analysis.

Neural Architecture Innovation for Enhanced Distinguishers: Architectural innovations have been pivotal in advancing neural cryptanalysis. Jain et al.[13] replaced convolutional networks with multilayer perceptrons (MLPs) to pioneer 3–6-round neural distinguishers for PRESENT. Bao et al.[14] integrated dense connectivity (DenseNet) and squeeze-and-excitation (SENet) modules to extend SIMON32/64 distinguishers to 7–11 rounds, demonstrating the efficacy of channel attention mechanisms. Zhang et al.[15] further enhanced SPECK distinguishers using GoogleNet-inspired architectures, highlighting the role of multi-scale feature fusion.

Integration of Auxiliary Analysis Techniques: The fusion of classical cryptanalytic methods with deep learning has unlocked new frontiers. Lu et al.[10] first bridged related-key differential analysis with neural networks, constructing high-precision distinguishers for SIMON and SIMECK family. Bao et al.[14] augmented SIMON32/64 distinguishers using neutral bits for data augmentation, while Seok et al.[16] leveraged machine learning to identify optimal input differentials, improving SIMON and SPECK accuracy.

Motivation: Based on prior work, we argue that current neural cryptanalysis faces three critical limitations.

-Data format optimizations remain tailored to individual ciphers, lacking adaptability to diverse architectures, which impedes generalized related-key differential dataset construction. How can we design common data formats for different cryptographic algorithm structures to generate differential data sets?

-Despite advances in architecture, transformative tools such as attention mechanisms that can prioritize non-linear cryptographic features remain underexplored in crypto-specific designs. How might attention-driven neural architectures dynamically amplify cryptographically critical features while suppressing noise in differential propagation?

-The current landscape presents numerous auxiliary techniques and data analysis methodologies in deep learning. How might the effective integration of these technologies with intelligent cryptanalysis extend neural distinguishers or enhance interpretability analysis, and what aspects require deeper exploration?

Contributions: This paper introduces a unified framework combining data optimization, network design, and input differential selection. Test on the standard cipher DES and PRESENT, it improves neural distinguisher accuracy and round coverage over existing methods. Additionally, a validation algorithm is proposed to assess the quality of differential-generated datasets from both cryptographic and machine learning perspectives, providing interpretable explanations for differential selection.

1. This paper proposes a novel framework for constructing related-key neural differential distinguishers by synergizing classical differential cryptanalysis with deep learning. The framework systematically addresses three core challenges: (1) dataset construction, where we introduce a multi-differential multi-ciphertext pair format tailored to Feistel (DES) and SPN (PRESENT) architectures, leveraging round

function structures to extract inter-ciphertext correlations and optimize feature representation; (2) network architecture design, employing a Deep Residual Shrinkage Network (DRSN)[17] enhanced with adaptive threshold shrinkage units to suppress noise in cryptographic data while amplifying critical differential patterns; (3) differential selection, which integrates traditional related-key differential analysis to identify high-probability differential combinations, thereby reducing neural network training complexity.

2. Based on this framework, we constructed related-key neural differential distinguishers for DES and PRESENT algorithms. For DES, our framework improved the accuracy of 6/7-round distinguishers by over 40% compared to existing distinguishers. Notably, we reported the first 8-round related-key neural differential distinguisher for DES. For PRESENT, we constructed the first 9-round related-key neural differential distinguisher, achieving both increased round numbers and accuracy under identical conditions.

3. Furthermore, we employed data analysis techniques, including Kernel Principal Component Analysis (KPCA) and K-means clustering, to evaluate the quality of differential datasets. By applying this approach to the DES and PRESENT algorithms, we successfully identified high-quality input differential pairs, enabling the construction of high-accuracy related-key neural distinguishers. Building on this, we proposed a data-driven framework that integrates KPCA and K-means clustering to systematically recognize cryptographically advantageous ciphertext-key differential combinations. We validated the effectiveness of this methodology by applying it to the SIMECK algorithm. Our framework bridges cryptographic heuristics with machine learning interpretability, offering a systematic solution for differential selection in related-key neural cryptanalysis.

Comparison with Previous Work. Our neural distinguishers achieve significant advancements over existing results in both round coverage and accuracy. Key results are summarized in Table 1 and Table 2. For the DES algorithm, when $m = 1$ (one ciphertext pair per sample), the accuracy of the 6-round distinguisher improved from 0.549 to 0.979, while the 7-round distinguisher achieved over 40% higher accuracy compared to [18] at $m = 16$. Notably, the first 8-round neural distinguisher for DES was constructed at $m = 32$. For the PRESENT algorithm, the 7-round distinguisher results surpassed three existing distinguishers by over 20% in accuracy. For the 8-round distinguisher, the framework outperformed the results in [19] at $m = 1$, improving accuracy from 0.509 to 0.570, and established a 9-round distinguisher with 0.529 accuracy at $m = 4$. These results demonstrate the framework's capability to balance high-probability differential propagation with robust feature learning, enabling deeper penetration and enhanced precision in neural cryptanalysis.

In this paper, the available computing resources are: an Intel(R) Core(TM) i9-14900K CPU @ 3.20GHz, a graphics card (NVIDIA GeForce RTX 4080 SUPER).

Table 1: Comparison of neural differential distinguishers of DES

Cipher	Round	m	Acc	Source
DES	6	1	0.549	[8]
			0.979	This work
		2	0.565	[8]
			0.578	[18]
			0.999	This work
		4	0.557	[8]
			0.627	[18]
			0.999	This work
		8	0.551	[8]
			0.690	[18]
			0.999	This work
		16	0.553	[8]
			0.769	[18]
			0.999	This work
	7	1	0.535	This work
		2	0.562	This work
		4	0.666	This work
		8	0.823	This work
		16	0.511	[18]
			0.951	This work
	8	32	0.505	This work

Table 2: Comparison of neural differential distinguishers of PRESENT

Cipher	Round	m	Acc	Source
PRESENT	7	1	0.549	[8]
			0.563	[20]
			0.719	This work
		2	0.550	[8]
			0.574	[18]
			0.593	[20]
			0.804	This work
		4	0.585	[8]
			0.609	[18]
			0.635	[20]
			0.896	This work
		8	0.579	[8]
			0.658	[18]
			0.691	[20]
			0.965	This work
		16	0.583	[8]
			0.723	[18]
			0.765	[20]
			0.994	This work
	8	1	0.509	[19]*
			0.570	This work
		2	0.514	[18]
			0.607	This work
		4	0.520	[18]
			0.658	This work
		8	0.529	[18]
			0.728	This work
		16	0.542	[18]
			0.810	This work
	9	2	0.502	This work
		4	0.529	This work
		8	0.544	This work
		16	0.567	This work

In [19], the 9-round distinguisher is defined with the whitening key placed in the first round, meaning their '9-round' configuration is equivalent to the 8-round configuration in this work.

Section 2 of this paper describes the two standardized cryptographic algorithms, DES and PRESENT, along with their foundational concepts. Section 3 presents the construction framework for related-key neural differential distinguishers. Building upon the framework outlined in Section 3, Section 4 details the implementation of the improved related-key differential distinguishers for DES and PRESENT. In Section 5, we employ data analysis techniques to analyze the differential datasets of DES and PRESENT and propose a generic verification algorithm to validate the quality of these datasets. Finally, the concluding section summarizes the current research work.

2 PRELIMINARIES

2.1 Discriptions of DES and PRESENT

2.1.1 Specification of DES.

Data Encryption Standard (DES) algorithm, designed by the IBM team, was formally adopted as a Federal Information Processing Standard (FIPS) by the National Bureau of Standards (now NIST) in 1977, becoming one of the most widely used block ciphers in history.

DES employs the classical Feistel structure with a block size of 64 bits and a key length of 56 bits (with an actual input of 64 bits, where 8 bits are reserved for parity checks). The encryption process consists of 16 rounds. Initially, the 64-bit plaintext undergoes an Initial Permutation (IP), followed by 16 rounds of Feistel transformations. In each round, the right half is expanded to 48 bits via the Expansion Permutation, XORed with the subkey, and then processed through eight distinct 6-bit input/4-bit output substitution boxes (S-boxes). The output is permuted via the P Permutation, XORed with the left half, and the two halves are swapped. After the final round, the ciphertext is obtained through the Inverse Initial Permutation (IP^{-1}).

The key expansion algorithm of DES transforms the initial 64-bit master key (including 8 parity bits) into sixteen 48-bit subkeys for the Feistel network using a systematic key scheduling strategy. This process involves three stages:

1. Permuted Choice PC-1: Removes parity bits and rearranges the remaining 56 valid key bits into two 28-bit left and right halves.

2. Circular Shifts: Prior to each round, the two halves undergo specific circular shifts (1 or 2 bits depending on the round).

3. Permuted Choice PC-2: Compresses the shifted 56-bit intermediate key into a 48-bit subkey for the current round.

The encryption process and key expansion algorithm of DES is shown in Figure 1.

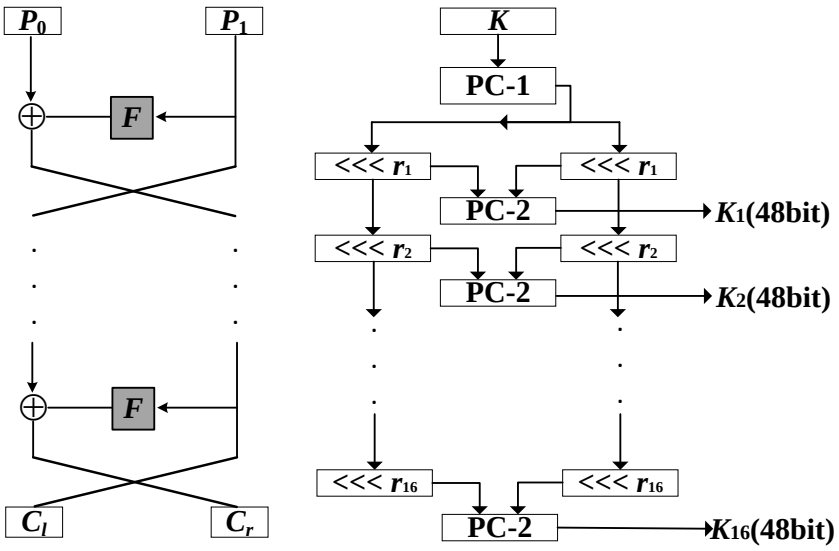

Figure 1: DES Encryption Process and Key Expansion Algorithm.

In the domain of deep learning-based differential cryptanalysis, [8] pioneered the construction of reduced-round DES neural differential distinguishers in 2021, achieving accuracy rates of 0.959 and 0.553 for 5-round and 6-round distinguishers, respectively. Subsequently, [18] developed a 6-round distinguisher with an accuracy of 0.769 and further reported the first 7-round distinguisher with an accuracy of 0.511. These advancements introduced novel perspectives for analyzing the security of the DES algorithm, demonstrating superior generality and feasibility compared to traditional differential cryptanalysis. However, intelligent analytical results under related-key conditions for DES remain absent in current research.

2.1.2 Specification of PRESENT.

The PRESENT algorithm, proposed by Bogdanov et al. at the CHES conference in 2007, is an ultra-lightweight block cipher specifically designed for resource-constrained devices such as RFID tags and sensor nodes. Currently standardized by ISO/IEC, PRESENT is widely adopted in security modules for IoT devices.

PRESENT employs a Substitution-Permutation Network (SPN) structure with a 64-bit block size, supporting two key lengths (80-bit or 128-bit), and executes 31 encryption rounds. Renowned for its exceptionally compact hardware footprint, it is hailed as one of the benchmarks in lightweight cryptography. The encryption process of PRESENT, illustrated in Figure 2, begins by loading the 64-bit plaintext into a state register, followed by 31 iterative rounds. Each round comprises three layers of operations:

1. AddRoundKey: The 64-bit state is XORed with the round key.
2. SBoxLayer: A single 4-bit input/4-bit output S-box is applied in parallel to 16 nibbles.
3. PLayer: A deterministic bitwise permutation achieves bit-level diffusion.

After the final round, the 64-bit ciphertext is directly output.

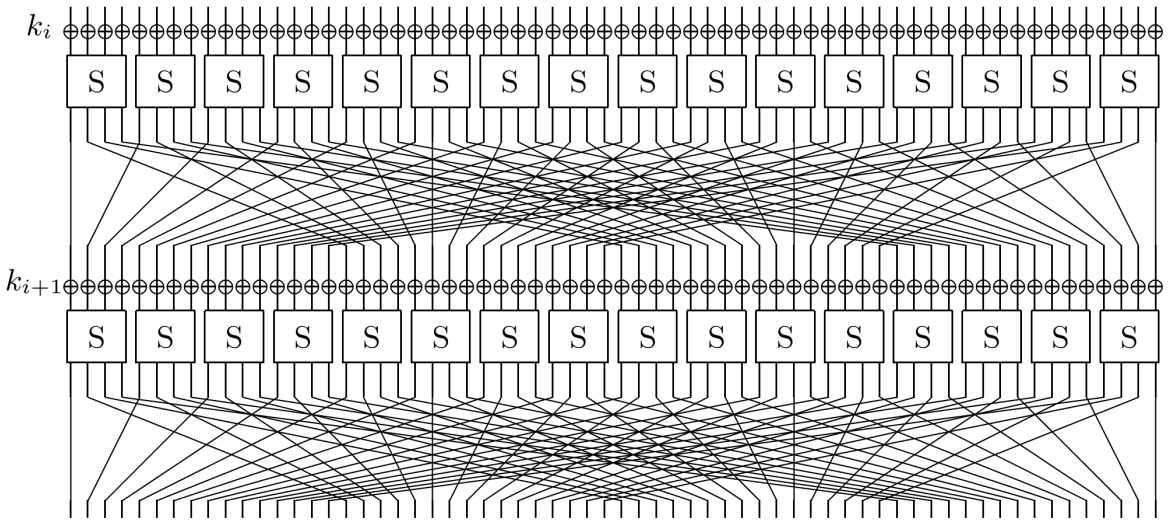

Figure 2: PRESENT Encryption Process.

The key expansion algorithm of PRESENT is an efficient round-key generation mechanism designed to produce unique 64-bit round keys for each encryption round. Its core principles involve cyclic shifts, nonlinear transformations, and round constant updates. For an 80-bit master key, denoted as $[k_{79}k_{78}\cdots k_0]$, the leftmost 64 bits of the current key register are extracted as the round key for the i-th encryption round. After extracting the round key, the key register is updated as follows:

$$1.[k_{79}k_{78}\cdots k_0] = [k_{18}k_{17}\cdots k_{19}]$$
$$2.[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$$
$$3.[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus round\_counter$$

Where S refers to the S-box in the encryption algorithm.

In the field of deep learning-based differential analysis, [18] constructed a neural differential distinguisher for 8-round PRESENT using 16 pairs of ciphertexts as training data, achieving a validation accuracy of 0.542. Subsequently, [19] developed a distinguisher model for 8-round PRESENT that attained a precision of 0.509 while utilizing only single ciphertext pairs, thereby dramatically reducing the data requirement. Additionally, PRESENT currently lacks reported intelligent analysis results under related-key conditions.

2.2 Differential Cryptanalysis and Related-key Differential Cryptanalysis

Differential cryptanalysis [21], proposed by Biham and Shamir, is a chosen-plaintext attack that analyzes how differences in plaintext pairs propagate through intermediate rounds of an iterative cipher. Resistance to differential cryptanalysis has become a fundamental criterion for evaluating the security of block ciphers. The core tool of differential cryptanalysis is the differential distinguisher. The process involves first identifying high-probability differential characteristics and then constructing distinguishers based on these characteristics.

Definition (Differential Characteristic): An i-round differential characteristic of an iterative block cipher refers to a sequence of differences $\Omega = (\beta_0, \beta_1, \cdots, \beta_{i-1}, \beta_i)$, where $X \oplus X^* = \beta_0$ is the input difference, and each intermediate state difference $Y_j \oplus Y_j^* = \beta_j (1 \leq j \leq i)$ satisfies specific propagation conditions through the cipher's rounds.

Definition (Differential Probability): The differentia probability $\mathrm{DP}(\alpha, \beta)$ of an i-round difference (α, β) is defined as the likelihood that an input pair with difference $X \oplus X^* = \alpha$, under uniformly distributed keys and plaintexts, produces an output pair with difference $Y_i \oplus Y_i^* = \beta$ after i rounds of encryption.

If an r round differential characteristic with probability higher than that of a random permutation is discovered, it enables distinguishing the r round cipher from a random permutation, thereby constructing an r round distinguisher. By leveraging this distinguisher, partial key information can be recovered through analyzing the distribution patterns of plaintext and ciphertext differences. A higher-probability distinguisher reduces the data complexity for key recovery, while a longer-round distinguisher lowers computational resource requirements.

Related-key differential cryptanalysis [22], introduced by Biham, exploits weaknesses in block cipher key expansion algorithms by constructing both plaintext differences (ΔP) and key differences (ΔK) to analyze ciphertext difference (ΔC) propagation and derive key correlations. Key steps include: Constructing differential paths aligning with key

scheduling properties; Filtering valid plaintext-ciphertext pairs; Recovering key bits via statistical or algebraic methods.

This method is particularly effective against ciphers with simplistic or linearly dependent key schedules, where active bits cancellation or subkey differential relationships reduce key space complexity. Additionally, there exist numerous analytical results of related-key attacks against standard ciphers. Beyond AES mentioned in [23], KASUMI in its full-round configuration can theoretically be broken by related-key rectangle attacks as demonstrated in [24]. However, its efficacy depends on the controllability of key differences and diminishes against ciphers with complex key expansion algorithms. Modern cipher designs mitigate such attacks by: Strengthening key schedules; Enhancing key dependency; Restricting differential propagation. As both a critical tool for cryptographic security evaluation and a catalyst for advancing key schedule complexity, related-key differential cryptanalysis remains pivotal in cipher analysis and design.

2.3 Differential Neural Distinguishers

The core mechanism of neural differential distinguishers lies in leveraging deep learning to achieve efficient classification of cryptographic data, sharing an inherent similarity with classification tasks in domains such as image recognition and speech processing. Gohr first introduced the concept of neural differential distinguishers, pioneering the integration of deep learning into cryptanalysis. These distinguishers train neural network models to accurately classify real ciphertext pairs (generated by encrypting plaintext pairs with fixed input differences) versus random ciphertext pairs (produced by encrypting random plaintext pairs). The trained distinguishers are then utilized for key recovery attacks, demonstrating superiority over traditional methods in both distinguisher accuracy and attack complexity. Notably, Gohr successfully recovered subkeys of 11-round SPECK32/64 using such neural distinguishers. The generation of the differential dataset for neural network training proceeds as follows: A plaintext pair $(P, P^*)$with input difference Δin is encrypted under a random master key KEY to produce a ciphertext pair (C, C^*), which serves as a training sample. During neural network training, each sample is assigned a binary label Y:

- Y=1: The ciphertext pair originates from a plaintext pair with input difference Δin.
- Y=0: The ciphertext pair is generated from random plaintext pairs.

The performance of such distinguishers is typically evaluated using two metrics: training accuracy and validation accuracy, where the latter is the critical indicator of practical efficacy. If the neural network achieves classification accuracy exceeding 50% after sufficient training, it is deemed an effective neural differential distinguisher.

Gohr's work demonstrates that a well-constructed neural differential distinguisher can leverage its output for key recovery attacks, with attack performance directly dependent on the distinguisher's accuracy. Consequently, developing high-accuracy neural distinguisher models holds significant importance for advancing cryptanalytic capabilities.

3 A Generalized Framework for Constructing Related-Key Differential Neural Distinguisher

The foundational framework for constructing related-key neural differential distinguishers comprises three components: dataset construction, differential selection, and network architecture design. In this section, we present the unified framework for building related-key neural differential.

3.1 Dataset Construction

Supervised learning, a critical branch of deep learning, relies on labeled training datasets where each sample is associated with a label representing the expected output of the model. These labels are pivotal for optimizing neural network parameters. Consequently, when constructing neural differential distinguishers, the selected data format should maximize diversity and comprehensively encapsulate potential scenarios to enhance distinguisher performance.

In Gohr's work, the input format for an n-round neural distinguisher (ND) is a single ciphertext pair (C_l, C_r, C_l^*, C_r^*), where the model learns features solely from ciphertext pairs. Benamira hypothesizes that the first convolutional layer in Gohr's neural network transforms the input(C_l, C_r, C_l^*, C_r^*) into linear combinations of $(C_l \oplus C_l^*, C_r \oplus C_r^*, C_l \oplus C_r, C_l^* \oplus C_r^*)$, and their interactions. Hou et al. extended this by designing an ND model that accepts multiple ciphertext pairs as input, denoted as (Δ_L^r, Δ_R^r), to leverage multi-differential information. In [10], LU et al. proposed an augmented data format $(\Delta_L^r, \Delta_R^r, C_l, C_r, C_l^*, C_r^*, \Delta_R^{r-1}, p\Delta_R^{r-2})$, incorporating intermediate results from preceding rounds into training samples.

Building on these insights, we introduce structure-specific input formats tailored to the Feistel and SPN architectures. These formats enhance the extraction of inter-ciphertext correlations and intrinsic cryptographic features, thereby improving distinguisher accuracy.

(1) For Feistel-Structured Ciphers

For Feistel-structured algorithms, we leverage their structural characteristics by partitioning the ciphertext into left and right halves, denoted as (C_l, C_r). Building on prior work, we design an input format $(\Delta_L^r, \Delta_R^r, C_l^{r-1}, C_l^{r-1^*}, C_l, C_r, C_l^*, C_r^*)$, where $F(C_l)$ represents the output of the F-function under a zero subkey, $C_l^{r-1} = F(C_l) \oplus C_r, C_l^{r-1^*} = F(C_l^*) \oplus C_r^*$. Fig.3 shows a schematic representation of these notations. For an n-bit block size, the input format comprises eight n/2-bit blocks. This format captures not only intra-ciphertext and inter-ciphertext pair features but also incorporates characteristics from preceding rounds. To enhance the neural distinguisher's performance, we associate each label with multiple ciphertext pairs encrypted under the same key, enabling the neural network to better extract correlated features between ciphertext pairs. In the context of the DES algorithm, this corresponds to an input format consisting of eight 32-bit blocks.

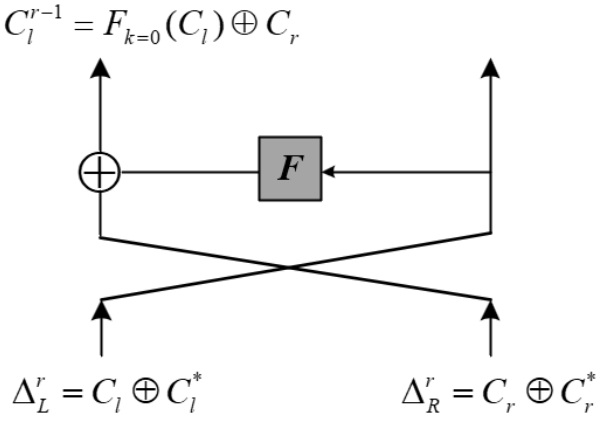

Figure 3: Feistel structured data format

(2) For SPN-Structured Ciphers

For SPN-based algorithms, where ciphertexts are single n-bit units, we adopt a multi-ciphertext input format $(\Delta C, C, C^*)$, This format explicitly encodes differential relationships between ciphertext pairs while retaining intrinsic features of individual ciphertexts. Figure 4 shows the SPN structured data format. By processing multiple ciphertext pairs, the model better captures inter-pair correlations, improving distinguisher accuracy. For PRESENT algorithm, this yields an input format of three 64-bit blocks.

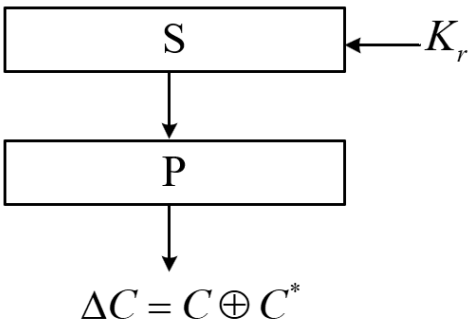

Figure 4: SPN structured data format

(3) Generalized Dataset Construction Workflow

For both architectures, we employ multi-ciphertext pair inputs with $(\Delta_L^r, \Delta_R^r, C_l^{r-1}, C_l^{r-1^*}, C_l, C_r, C_l^*, C_r^*)$ or $(\Delta C, C, C^*)$. During the data generation phase, fixed input differences Δin and key differences Δin^* are first determined. Let m denote the block length of the target algorithm. A total of n samples are generated, with each sample containing S ciphertext pairs. The generalized dataset construction workflow proceeds as follows:

a. Plaintext Generation:

- Generate $S * n$ random plaintexts P, partitioned into n groups.
- XOR each element in P with a fixed input difference Δin to create P^*.
- Generate a label vector Y of length n, populated with randomly distributed 0s and 1s.

b. Key Schedule:

- Generate n master keys KEY, each with length matching the algorithm's key size.
- Derive KEY^* by XORing each key in KEY with a fixed key difference Δin^*.

c. Label Assignment:

- For groups labeled Y=0, replace S plaintexts in P^* with random values.

d. Encryption:

- Encrypt P and P^* under KEY and KEY*, respectively, yielding ciphertext sets (C, C^*)

Compute ΔC, concatenate values according to the target format , and assign labels from Y. The workflow is illustrated in Figure 3.

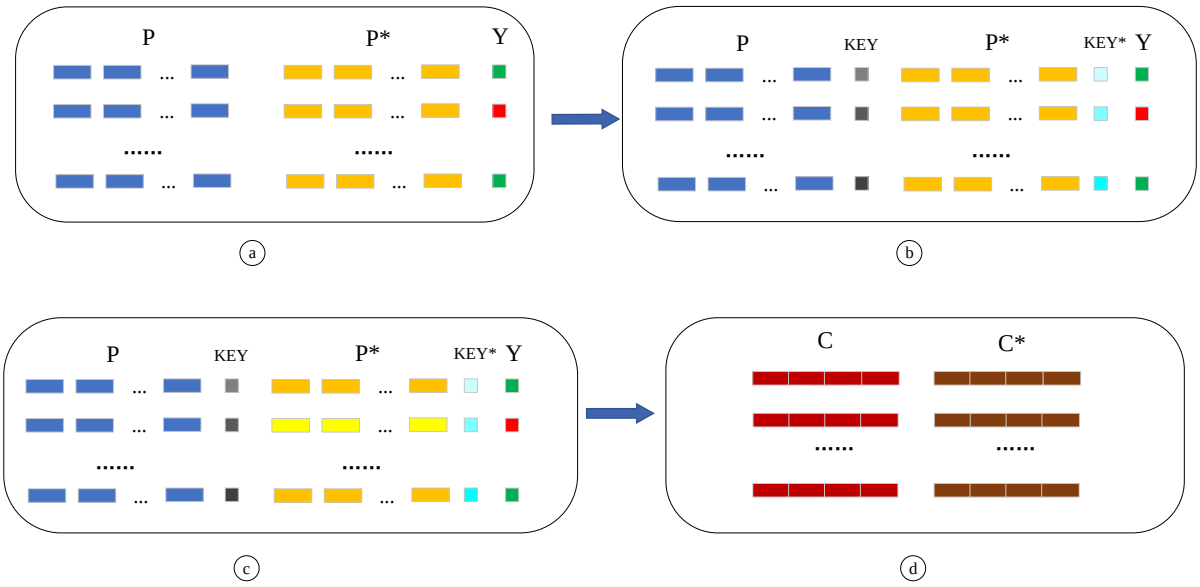

Figure 5: Dataset Construction Workflow

3.2 Network Architecture Design

In prior work, Gohr employed Residual Networks (ResNet) as the foundational architecture. ResNet, one of the most representative convolutional neural networks (CNNs)[25], addresses the vanishing gradient problem during CNN training and enables deeper network structures for higher accuracy. This architecture enhances training stability, computational efficiency, and classification performance. Subsequent studies by Baksi et al. [26] explored diverse neural network architectures, including multilayer perceptrons (MLPs), CNNs, and long short-term memory networks (LSTMs), achieving varying success in constructing distinguishers for different cryptographic algorithms.

The Deep Residual Shrinkage Network (DRSN), an enhanced ResNet variant, is specifically designed for noise-contaminated data and has demonstrated unique efficacy

in cryptographic distinguisher construction. Unlike generic architectures, DRSN integrates soft thresholding—a signal denoising technique—with adaptive threshold learning to suppress non-informative noise (e.g., random ciphertext fluctuations) while amplifying cryptographically critical features (e.g., high-probability differential paths). This dual mechanism aligns with the core requirements of distinguisher design: (1) filtering out stochastic interference inherent in encrypted data and (2) preserving deterministic non-random patterns. Key innovations include:

Soft Thresholding Layer: Adaptively zeroes features with magnitudes below a learned threshold and shrinks others toward zero. In cryptographic contexts, this selectively suppresses noise-corrupted activations such as low-probability differentials while retaining discriminative signals including biased key-dependent correlations, effectively mimicking traditional differential filtering but in a data-driven manner.

Adaptive Threshold Generation: A lightweight subnetwork generates sample-specific thresholds by analyzing global feature statistics. For ciphertext data, this allows dynamic adaptation to varying noise levels across different encryption rounds or key schedules, ensuring robustness against algorithmic nonlinearities.

Macro Architecture: Retains ResNet's backbone of stacked residual blocks with convolutional layers, batch normalization (BN), ReLU activation, and identity mapping. Each residual block embeds the soft thresholding layer and threshold generation subnetwork.

DRSN exhibits unique applicability in uncovering non-random cryptographic patterns due to its adaptive denoising and fine-grained feature enhancement capabilities. By synergizing residual learning, attention mechanisms, and soft thresholding, DRSN effectively extracts subtle yet critical non-random features from noisy ciphertext data.

The architecture comprises three components:

- **Input Layer:** Accepts data in 8×32-bit (DES) or 3×64-bit (PRESENT) blocks, aligned with the selected input formats.
- **BLOCK1:** A 1D convolutional layer (Conv1D) and two dense layers process fixed-length inputs.
- **BLOCK2:** DRSN-based residual shrinkage blocks perform feature extraction and compression via convolutional operations, soft thresholding, and residual connections.
- **Prediction Layer (BLOCK3):** A flatten layer transitions convolutional outputs to dense layers. Two hidden dense layers (64 neurons each, matching the block size) process features, followed by a Sigmoid activation for binary classification.

The network architecture and training hyperparameters are detailed in Figure 4 and Table 3.

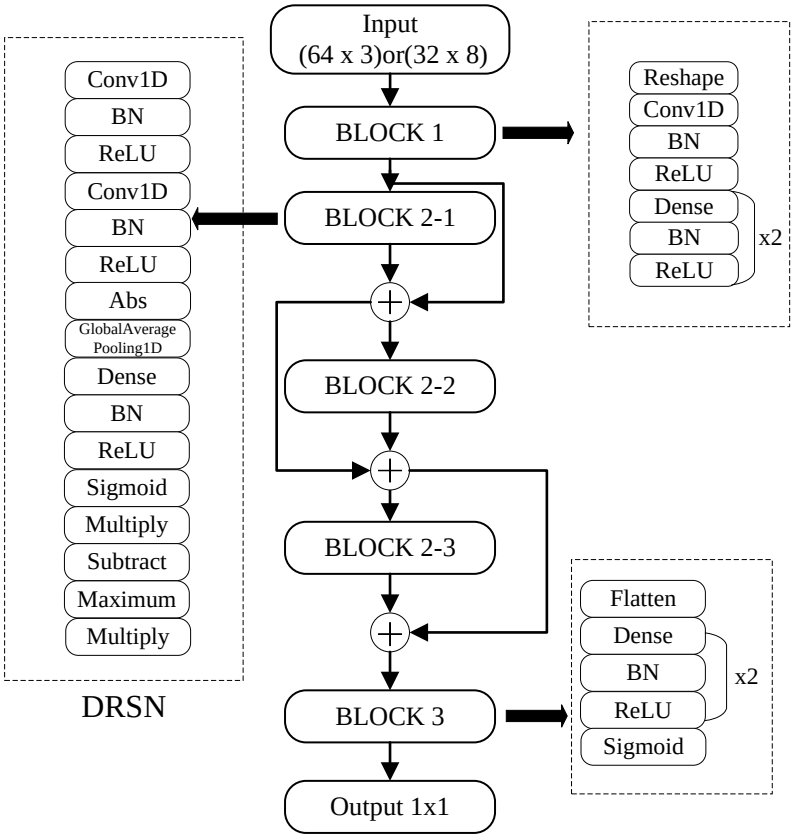

Figure 6: Network structure

Table 3: Parameter setting.

Training sample	10^7
Verify sample	10^6
Epoch	120
Batch size	10000
Learning rate	Cyclic learning rate
Step size	30
Base learning rate	0.0001
Maximum learning rate	0.002
Loss function	MSE
Optimizer	Adam
L2 regularization parameter	10^{-4}

4 Related-key Differential Neutral Distinguishers for Round-reduced DES and PRESENT

In this section, based on the framework proposed in the preceding section, we construct related-key neural differential distinguishers for two classical block ciphers, DES (Feistel structure) and PRESENT (SPN structure), addressing three critical aspects: differential selection, dataset construction, and network architecture design.

Selecting appropriate combinations of input differences (ΔP) and key differences (ΔK) is pivotal to constructing effective distinguishers. For each algorithm, we configure these differences according to its specific structure and key expansion mechanism. The goal is to ensure that round key differences interact with input differences in a manner that either cancels out or forms advantageous configurations during differential propagation. This strategy minimizes the emergence of unnecessary active bits caused by nonlinear operations and facilitates high-probability propagation patterns, enabling deeper analysis over more encryption rounds.

4.1 Differential Selection for DES

Based on the DES algorithm's structure and key expansion mechanism, we discard the parity bits of the master key and permute the remaining 56 bits to generate 16 subkeys. The subkey characteristics for the first five rounds are summarized in Table 4.

Table 4: Characteristics of DES Subkeys in the First Five Rounds

Subkey	Missing bit position
K1	C06 C07 C11 C12 C43 C46 C50 C52
K2	C03 C04 C35 C38 C42 C44 C61 C62
K3	C19 C22 C26 C45 C46 C52 C55 C57
K4	C03 C06 C10 C29 C30 C36 C39 C41
K5	C13 C14 C23 C25 C49 C52 C53 C59

To propagate low Hamming-weight differentials across as many rounds as possible, we analyze the differential propagation process in DES, leveraging the linearity of its key schedule and the differential propagation properties of the Feistel structure. By exploiting deterministic patterns in key expansion differentials, we align key differences (ΔK) with plaintext/ciphertext differences (ΔP/ΔC) to create synergistic cancellation or reinforcement effects. This strategy extends the effective differential chain length and constructs long-round propagation paths for low-weight differentials.

The DES key schedule (PC-1 permutation and circular shifts) exhibits predictable periodicity in subkey differential propagation due to its linear nature. Leveraging the Feistel structure's properties, key differences can cancel out with expanded right-half plaintext differences in the first or second round. By applying the inverse of the expansion permutation to key differences and constraining the number of active bits, we derive compatible plaintext differences.

Through filtering bits that meet specific criteria (e.g., absence in early rounds), we identify four active bits—C38, C46, C52, and C62—as optimal for differential propagation. Consequently, we define candidate plaintext-key differential pairs:

- Plaintext differences:

$$\Delta_1 = (0\text{x}0, 0\text{x}00000040)\,, \Delta_2 = (0\text{x}00002000, 0\text{x}0)\,,$$
$$\Delta_3 = (0\text{x}20000000, 0\text{x}00)\,, \Delta_4 = (0\text{x}0, 0\text{x}00000002)\,.$$

- Key differences:

$$\Delta_1^* = (0\text{x}0000000004000000)\,, \Delta_2^* = (0\text{x}0000000000040000)\,,$$
$$\Delta_3^* = (0\text{x}0000000000001000)\,, \Delta_4^* = (0\text{x}0000000000000004)\,.$$

When C52 (bit 52 of the key register) is active, the linear key schedule ensures its absence in subkeys of rounds 1, 3, and 5. For round 2, the subkey difference ΔK_2=($0x100$ 000000000) undergoes inverse expansion permutation, corresponds to right-half input difference ΔR_1= (0x20000000) .Thus, the plaintext difference is set to ΔL= (0x20000000) , ΔR = (0x0) .Analysis detail differential propagation, in the second round, the subkey K_2 uses bit C52 at position 4. The expanded right-half difference ΔR_1's 3rd bit cancels ΔK_2's 4th bit, justifying ΔL= (0x20000000). In the third round, with C52 inactive in K_3, no subkey difference exists, resulting in zero S-box output difference.

The full propagation path is illustrated in Figure 5. This analysis is verifiable by tracking plaintext and subkey differential transformations through the round function.

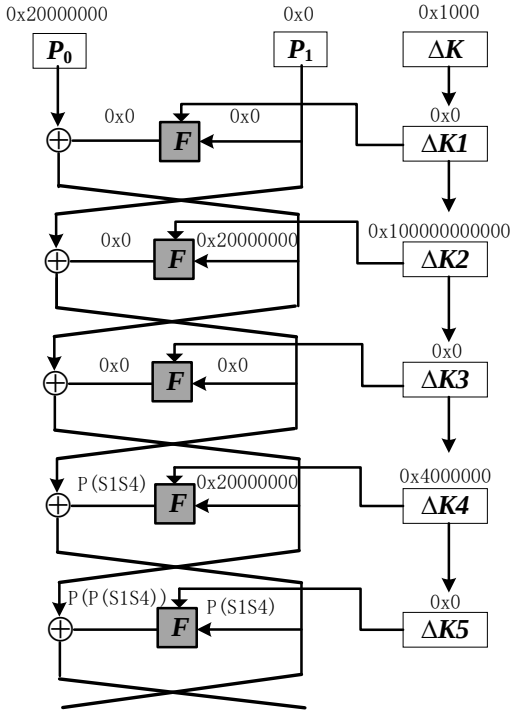

Figure 7: Differential propagation process

4.2 Related-Key Neural Differential Distinguishers for DES

For the related-key neural differential analysis of DES, we train distinguishers using the four plaintext-key differential combinations satisfying propagation constraints (Section 4. 1), employing the network architecture and hyperparameters defined in the preceding chapter. The accuracy of the trained 5-round and 6-round distinguishers is summarized in Table 5.

Table 5: Accuracy of Neural Distinguishers for 4 Candidate Differential Pairs

Plaintext difference	Round	Acc
(0x20000000, 0x00000000)	5	0.999
	6	0.979
(0x20000000, 0x00000000)	5	0.999
	6	0.951
(0x00000000, 0x40000000)	5	0.949
	6	0.500
(0x00000000, 0x20000000)	5	0.970
	6	0.501

From the training results, the combination of plaintext difference (0x20000000, 0) and key difference (0x0000000000001000) demonstrates superior performance compared to other candidates. The 7-round distinguisher trained on this combination achieves an accuracy of 0.5354. Notably, our method constructs a 7-round distinguisher using only a single ciphertext pair per sample, outperforming results in [8] and highlighting the advantage of related-key neural differential distinguishers.

To investigate the impact of data scale on model performance, we further conduct experiments for 6, 7, and 8-round DES with varying numbers of ciphertext pairs per sample (m=1, 2, 4, 8, 16). Key metrics—validation accuracy, true positive rate (TPR), and true negative rate (TNR)—are detailed in Table 6.

Table 6: Performance Metrics of Related-Key Distinguisher Models for DES

m	Round	TPR	TNR	Acc
1	6	0.967	0.995	0.979
	7	0.619	0.451	0.535
2	6	1	1	1
	7	0.644	0.480	0.562
4	6	1	1	1
	7	0.686	0.646	0.666
8	6	1	1	1
	7	0.798	0.848	0.823
16	7	0.941	0.960	0.951
32	8	0.041	0.962	0.505

For 6 and 7-round distinguishers, increasing m (ciphertext pairs per sample) significantly improves accuracy. The accuracy of the neural distinguisher consistently outperforms existing results across all m. At m=32, we construct the first 8-round related-key neural differential distinguisher for DES, achieving a validation accuracy of 0.505.

4.3 Differential Selection for PRESENT

By analyzing the structure and key expansion algorithm of PRESENT, we observe that the master key is iteratively updated to generate round subkeys. Considering key differences with a Hamming weight of 1, the characteristics of the subkeys in the first two rounds are summarized in Table 7.

Table 7: Characteristics of PRESENT Subkeys in the First Five Rounds

Active bit position	ΔK_1 Hamming weight	ΔK_2 Hamming weight
C1-C45	1	1
C46-C61	1	0
C62 C63	1	3
C64	1	2
C65	0	3
C66-C80	0	1

Similar to DES, constructing low Hamming-weight differential paths for PRESENT requires selecting input differences that activate the fewest S-boxes while leveraging the linearity of the key schedule to suppress differential propagation. Based on PRESENT's key expansion mechanism, when the active bit of the master key difference lies within C45–C60, the second-round subkey difference is guaranteed to be zero due to the key scheduling rules. Consequently, we identify 16 candidate input differences that satisfy the differential propagation requirements.

Through analyzing the differential propagation process, we set the plaintext difference to Δ= (0x00000000000010) and the key difference to ΔK= (0x00000000000000100000). In the first round, the active bit of subkey K_1 (bit 60) cancels out the active bit of the plaintext difference ΔP through XOR, resulting in a zero intermediate state difference after the first round. Thus, the plaintext difference is set to Δ= (0x00000000000010). In the second round, since the active bit C59 is absent, subkey K_2 has no difference, and the input difference remains zero, effectively suppressing S-box activation and differential propagation in this round.

4.4 Related-Key Neural Differential Distinguishers for PRESENT

Similar to the training of DES-related key neural differential distinguishers, based on the analysis results from earlier sections, we construct neural differential distinguishers using plaintext differences ΔP= (0x8 $\lll i$) and key differences ΔK= (0x80000 $\lll i$), where $i \in [0, 15]$. We train distinguishers on the qualified differential combinations, and the accuracy of the constructed 7-round distinguisher is shown in Table 8.

Table 8: Accuracy of Neural Distinguishers for 16 Candidate Differential Pairs

Plaintext difference	Acc
0x0000000000080000	0.670
0x0000000000100000	0.715
0x0000000000200000	0.584
0x0000000000400000	0.649
0x0000000000800000	0.677
0x0000000001000000	0.501
0x0000000002000000	0.584
0x0000000004000000	0.599
0x0000000008000000	0.625
0x0000000010000000	0.623
0x0000000020000000	0.570
0x0000000040000000	0.615
0x0000000080000000	0.646
0x0000000100000000	0.606
0x0000000200000000	0.563
0x0000000400000000	0.610

From the distinguisher results, the combination ΔP= (0x00000010) and ΔK=($0x0000$ 0000000000100000) achieves the highest accuracy. Using this differential pair, we construct an 8-round distinguisher with an accuracy of 0.5702. Furthermore, we conduct experiments on 7-, 8-, and 9-round PRESENT with varying numbers of ciphertext pairs per sample (m=1, 2, 4, 8, 16). The results are summarized in Table 9. Under related-key conditions, we construct the first 9-round neural differential distinguisher for PRESENT, achieving an accuracy of 0.5669.compared to neural distinguishers of [19], our results demonstrate improvements in both round coverage and accuracy.

Table 9: Performance Metrics of Related-Key Distinguisher Models for PRESENT

m	Round	TPR	TNR	Acc
1	7	0.828	0.608	0.719
	8	0.720	0.420	0.570
2	7	0.869	0.740	0.804
	8	0.709	0.505	0.607
	9	0.673	0.331	0.502
4	7	0.924	0.868	0.896
	8	0.746	0.571	0.658
	9	0.577	0.481	0.529
8	7	0.974	0.956	0.965
	8	0.786	0.670	0.728
	9	0.621	0.466	0.544
16	7	0.997	0.992	0.994
	8	0.845	0.776	0.810
	9	0.627	0.507	0.567

In this section, we first analyze the structural characteristics and key expansion algorithms of both DES and PRESENT. Based on their specific features, we derive candidate

combinations of plaintext differences (ΔP) and key differences (ΔK) tailored to each algorithm. Subsequently, we train related-key neural differential distinguishers for DES and PRESENT using these differential pairs.compared to existing results, our distinguishers achieve improvements in both round coverage and accuracy.

5 Interpretive of Differential Dataset: a Machine Learning Perspective

In the preceding section, we determined candidate combinations of plaintext differences (ΔP) and key differences (ΔK) by analyzing the cipher structures and key expansion algorithms from a cryptographic perspective, followed by an exhaustive evaluation to select optimal input differences. However, this approach lacks interpretability regarding the intrinsic reasons for differential quality. To address this, we leverage kernel principal component analysis (KPCA) and K-means clustering to assess the quality of differential datasets, aiming to identify high-accuracy input differences through machine learning-driven feature analysis.

5.1 Data Analysis Methods: Principal Component Analysis and K-means Clustering

Principal Component Analysis (PCA) is a classical linear dimensionality reduction technique that transforms a set of potentially correlated variables into linearly uncorrelated principal components via orthogonal transformations. Its core objective is to reduce data dimensionality while preserving maximal information, facilitating data analysis, visualization, and storage. Key steps include data centralization, covariance matrix computation, eigenvalue decomposition or singular value decomposition (SVD), principal component selection, and data projection. PCA is widely used in data preprocessing, feature extraction, and visualization—for example, extracting dominant features in data processing or identifying major variation patterns in gene expression data.

Kernel Principal Component Analysis (KPCA) extends traditional PCA to nonlinear dimensionality reduction using kernel methods. By employing the kernel trick, KPCA implicitly maps nonlinear data structures from the original low-dimensional space to a high-dimensional feature space, where linear PCA is performed to extract nonlinear principal components. Specifically, kernel functions compute pairwise similarities (inner products) between samples, circumventing explicit high-dimensional mapping and avoiding the "curse of dimensionality. " For instance, Gaussian kernels can map circularly distributed data into a linearly separable high-dimensional space, enabling PCA to identify principal directions that reveal nonlinear structures when projected back to low dimensions. Implementation Steps of KPCA:

1. Kernel Matrix Construction: Compute kernel function values for all sample pairs.

2. Kernel Matrix Centralization: Center the kernel matrix.

3. Eigenvalue Decomposition: Perform eigendecomposition on the centered kernel matrix and select eigenvectors corresponding to the largest eigenvalues as principal components.

4. Data Projection: Project data onto the principal component directions.

KPCA excels at capturing complex nonlinear patterns (e.g., spirals, clusters) and supports flexible kernel selection to adapt to data characteristics. It is widely applied to nonlinear feature extraction tasks such as handwritten digit recognition and image classification.compared to PCA, KPCA overcomes linear limitations and better captures nonlinear relationships but increases computational complexity and dependency on kernel selection.

K-Means Clustering is a machine learning algorithm that partitions data objects into k clusters based on shared features. Classified as an unsupervised machine learning method, its objective is to group data points such that intra-cluster similarity is maximized; equivalently, the distance between cluster centroids and their constituent objects is minimized.

The K-means algorithm operates as follows. Let x represent a data object and μ_i denote the centroid of cluster C_i. The criterion function E is defined as:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

The algorithm comprises two phases:

1. Initialization: For a given k, randomly select k initial centroids.

2. Iteration:

   - Assign each data point to the nearest centroid using the Euclidean distance:

$$d(x, \mu_i) = \sqrt{\sum_{j=1}^{n} (x_j - \mu_{ij})^2}$$

   - Recalculate centroids as the mean of all points in each cluster.
   - Repeat until convergence, i. e. , when the total intra-cluster distance E is minimized.

5.2 Analysis of Differential Datasets for DES

For the DES algorithm, we construct four candidate plaintext-key differential pairs and generate 6-round differential datasets of size 10^6 for each pair. We apply kernel principal component analysis (KPCA) for dimensionality reduction. To visualize the reduced-dimensional data, we project the datasets onto three eigenvectors corresponding to the largest eigenvalue ratios, serving as coordinate axes. The visualization results for random differentials are shown in Figure 6, while those for the four candidate differentials are presented in Figure 7.

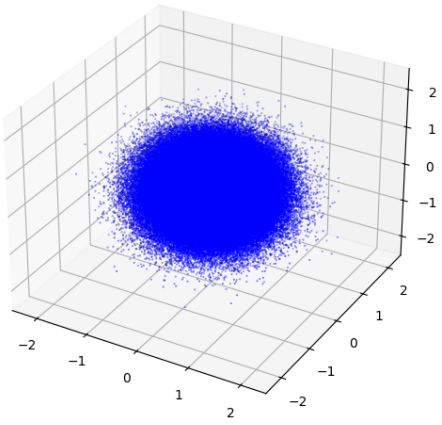

Figure 8: Random Situation Visualization Results

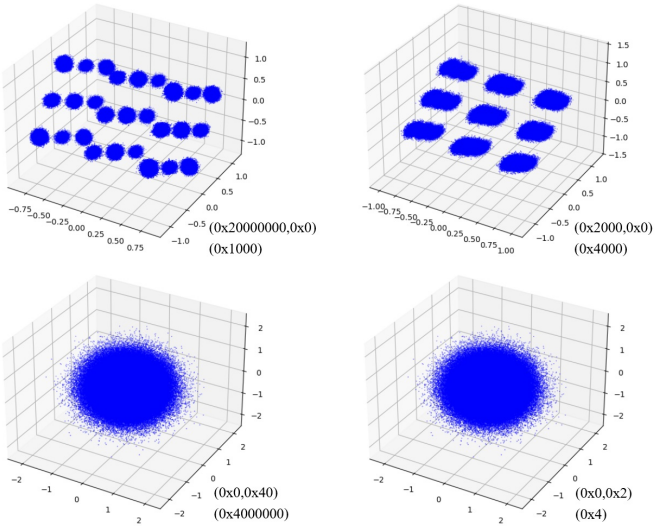

Figure 9: KPCA Visualizations of 4 Candidate Differences

For the case of random differences, the visualization shows a single spherical cluster, similar to data sampled from a Gaussian distribution. For candidate differences, the datasets form multiple high-density clusters, indicating that high-accuracy differentials induce aggregation of similar data points in the projected space, enabling clear separation between clusters. The dataset generated by the optimal plaintext difference $(0x20000000, 0)$ and key difference $(0x000000001000)$ forms 27 distinct clusters, demonstrating superior separability compared to other candidates.

To quantify clustering effectiveness, we perform K-means clustering on the projected data and evaluate cluster quality using the Silhouette Score, which measures intra-cluster compactness and inter-cluster separation. Additionally, We have evaluated the Silhouette Score and accuracy corresponding to the 6-rounds dataset generated by four candidate differences in Table 10.Based on visual analysis, we preset the cluster number to 27.combined with Table 10 and Figure 8, the K-means results and Silhouette Score distributions reveal a significant positive correlation between model accuracy and clustering quality. This implies that high-accuracy differentials exhibit stronger intra-cluster cohesion and

inter-cluster separation in the feature space, validating the intrinsic link between data representation quality and distinguisher performance.

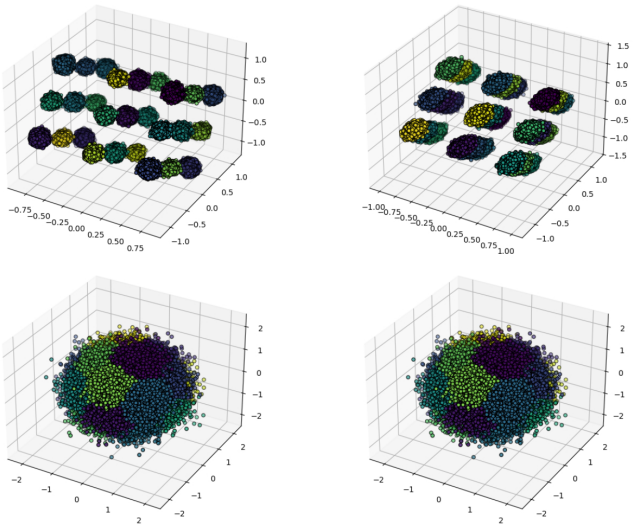

Figure 10: K-means clustering results for 4 candidate differences

Table 10: 6-round DES evalutation results and their training results

Plaintext difference	Silhouette score	Acc
(0x20000000, 0x00000000)	0.733	0.979
(0x20000000, 0x00000000)	0.683	0.951
(0x00000000, 0x40000000)	0.214	0.500
(0x00000000, 0x20000000)	0.217	0.501

This machine learning-driven framework provides an interpretable methodology for evaluating differential quality, complementing traditional cryptographic analysis. For High-Accuracy Difference, datasets corresponding to high-accuracy distinguishers exhibit prominent multi-cluster distribution characteristics in the feature space. These datasets form multiple high-density clusters with sharp inter-cluster boundaries, demonstrating strong intra-cluster cohesion and inter-cluster separability. The observed clustering patterns validate that "high-quality" differentials can be characterized by distinct geometric separability, which is directly linked to the classification performance of neural distinguishers. This provides data-driven quantitative metrics for evaluating differential quality, complementing traditional cryptographic criteria.

5.3 Analysis of Differential Datasets for PRESENT

Similar to the DES analysis, we identify 16 candidate plaintext-key differential pairs for the PRESENT algorithm that satisfy propagation constraints. For each candidate pair, we first construct a 6-round differential dataset of size 10^6 and apply kernel principal component analysis (KPCA) for nonlinear feature extraction and dimensionality reduction. To explore the latent structure of the reduced-dimensional data, we project the high-dimensional datasets onto three principal components with the highest variance explained ratios, enabling multi-view 3D visualization. The visualization results for all 16

candidate differentials are shown in Figure 9, which clearly delineates their distribution patterns and separability in the feature space. Figure 10 shows the visualized results of k-means clustering for 16 candidate differences. In Table 11, we perform experiments on all 16 pairs of candidate differences and obtain the accuracy of the corresponding 6-round distinguishers and give the Silhouette Score corresponding to its dataset.

The analysis reveals a strong positive correlation between Silhouette Scores and distinguisher accuracy, underscoring the cryptographic relevance of clustering quality in feature spaces. Notably, differentials such as 0x00000080 (Silhouette: 0.818, Accuracy: 0.962) and 0x00000800 (Silhouette: 0.761, Accuracy: 0.919) exhibit both high cluster separability and superior classification performance, indicating their alignment with cryptographically favorable properties like minimized active S-boxes and high-probability propagation paths. Conversely, poorly clustered differentials (e.g., 0x00000100, Silhouette: 0.417, Accuracy: 0.501) yield near-random accuracy. Our framework filters differential candidates through Silhouette scores, selecting those with high data quality and model performance.

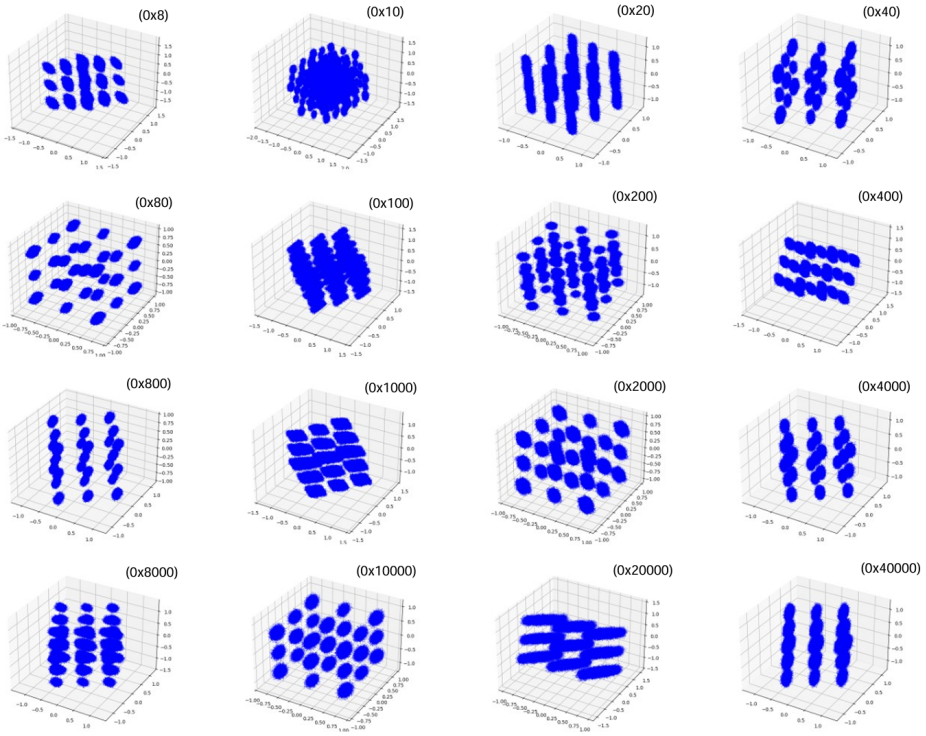

Figure 11: KPCA Visualizations of 16 Candidate Differences

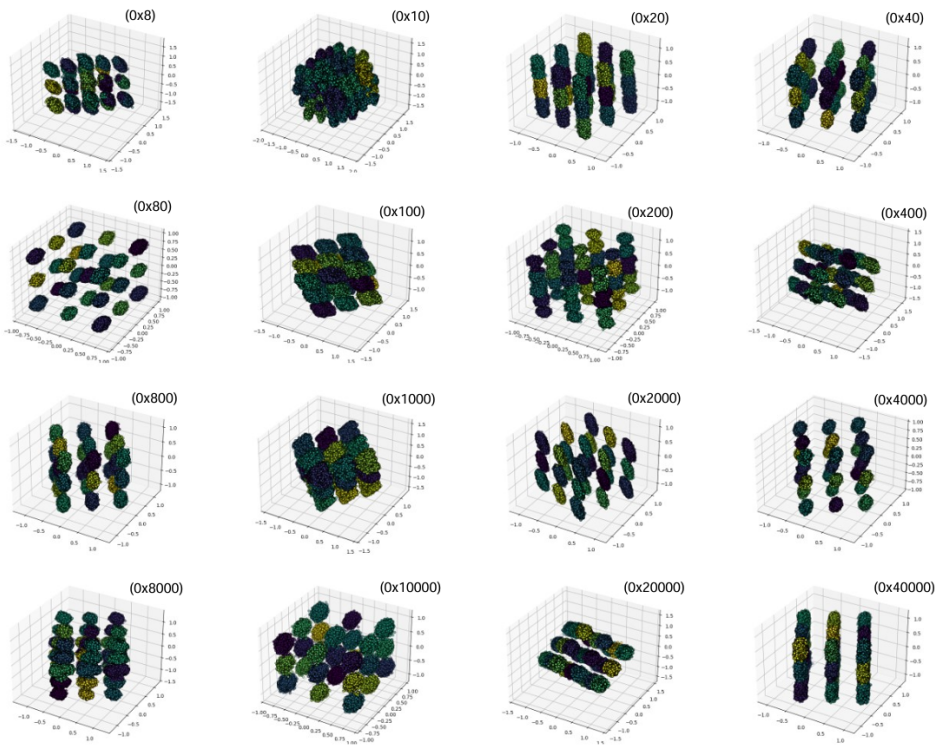

Figure 12: K-means clustering results for 16 candidate differences

Table 11: 6-round PRESENT evalutation results and training results

Plaintext difference	Silhouette score	Acc
0x00000008	0.695	0.915
0x00000010	0.479	0.930
0x00000020	0.470	0.819
0x00000040	0.747	0.932
0x00000080	0.818	0.962
0x00000100	0.417	0.501
0x00000200	0.587	0.881
0x00000400	0.653	0.890
0x00000800	0.761	0.919
0x00001000	0.695	0.921
0x00002000	0.701	0.850
0x00004000	0.720	0.911
0x00008000	0.648	0.924
0x00010000	0.711	0.912
0x00020000	0.495	0.845
0x00040000	0.557	0.883

5.4 Good Input Difference Verification for The Differential Dataset

In this section, we propose a validation algorithm to assess the quality of trainable related-key differential datasets. Building on the findings from Sections 5.2 and 5.3, we first configures suitable combinations of input differences and key differences by leveraging

cryptographic characteristics and key expansion mechanisms. This ensures that round key differences either cancel out or favorably interact with input differences, thereby minimizing unnecessary active bits induced by nonlinear operations and enabling high-probability differential propagation across extended rounds.

Subsequently, the algorithm performs kernel principal component analysis (KPCA) and K-means clustering to reduce the dimensionality of differential datasets generated with target differences. The reduced data is projected into a latent space for cluster analysis. Clustering quality is then quantified using the silhouette score, which evaluates intra-cluster cohesion and inter-cluster separation to identify optimal differential candidates.

The proposed algorithm systematically integrates cryptographic validation and machine learning-driven analysis to optimize differential selection. Lines 1–13 outline the systematic construction of differential datasets from candidate plaintext-key differential pairs (Δ_1, Δ_2), with Line 2 ensuring minimal S-box activation in early rounds to suppress unnecessary active bits and enhance differential propagation probability. Subsequently, kernel principal component analysis (KPCA) is applied: a Gaussian kernel constructs the nonlinear similarity matrix $\mathbf{K}$ (Line 14), which is centered and decomposed to extract the top three eigenvectors $\mathbf{V}$ (Line 15), enabling projection of the dataset $\mathcal{D}$ into a 3D latent space $\mathbf{Z} = \mathbf{KV}$ for visualization and dimensionality reduction (Line 16). The reduced data is then partitioned into T clusters via K-means (Line 17), which is a parameter derived from the KPCA visualization results, consistent with the previous observations of DES and PRESENT, where high-quality differentials naturally form distinct groupings. Finally, the silhouette score S quantifies clustering quality by measuring intra-cluster cohesion and inter-cluster separation (Line 18), with higher scores indicating cryptographically advantageous differentials—such as those minimizing active S-boxes or maximizing linear trail consistency. This pipeline bridges cryptographic heuristics with data-driven metrics, offering a reproducible framework for neural distinguisher design.

Algorithm 1 Good input difference verification algorithm

Input: ΔP(candidate plaintext difference)
ΔK(candidate key difference)
$n-$number of ciphertext pairs
$ks-$key size of target cipher
$bs-$block size of target cipher
$t-$quantity of required difference pair

Output: Optimal differential pair$(\Delta P^*, \Delta K^*)$with the highest t-position silhouette score

for each $(\Delta_1, \Delta_2) \in (\Delta P, \Delta K)$ **do**
  Validate propagation: Ensure $(\Delta_1 \oplus \Delta K_1)$ minimizes active S-boxes in initial rounds.
  $K \leftarrow Rand(0, 2^{ks} - 1)$
  $K^* = K \oplus \Delta_2$
  for $i \in n$ **do**
    $label \leftarrow Rand(0, 1)$
    if $label = 0$ **then**
      $P, P^* \leftarrow Rand(0, 2^{bs} - 1), rand(0, 2^{bs} - 1)$
    else
      $P \leftarrow Rand(0, 2^{bs} - 1), P^* \leftarrow P \oplus \Delta_1$
    end if
    $D \leftarrow Append(E_K(P) \parallel E_{K^*}(P^*))$
  end for
  Construct kernel matrix K using Gaussian kernel
  Center K and compute top 3 eigenvectors $\mathbf{V}$
  Project dataset $\mathcal{D}$ to 3D latent space:$\mathbf{Z} = \mathbf{KV}$
  $C \leftarrow Kmeans(\mathcal{D}, T)$
  $S \leftarrow SilhouetteScore(\mathcal{D}, cluster\_label = C)$
end for
Sort $(\Delta P^{(i)*}, \Delta K^{(i)*}, S^{(i)})$ by $S^{(i)}$ in descending order.
Select the first t entries:

$$\mathcal{T} = \left[(\Delta P^{(1)}, \Delta K^{(1)}, S^{(1)}), \ldots, (\Delta P^{(t)}, \Delta K^{(t)}, S^{(t)})\right]$$

return $\mathcal{T}$

To evaluate the effectiveness of our proposed algorithm in validating related-key differential datasets, we applied it to the SIMECK32/64 [27], a lightweight block cipher specifically designed for constrained environments. Featuring compact hardware implementation and low power consumption, SIMECK is particularly suitable for embedded systems and IoT devices. According to the findings in [27], when the plaintext difference $\Delta P = (0, 0x1 \lll i)$ and key difference $\Delta K = (0, 0, 0, 0x1 \lll i)$, where $i \in [0, 15]$, the plaintext and key differences cancel each other in the first round, and both remain zero for the subsequent three rounds. A key difference re-emerges only at the fifth round, which led to the identification of 16 candidate differential pairs. For all candidate differential pairs, we applied the validation algorithm to the 11-round SIMECK difference dataset, where the results for 16 pairs of candidate differences are summarized in Table 12. The KPCA visual results are shown in Figure 13, and the K-means clustering results are shown in Figure 14.

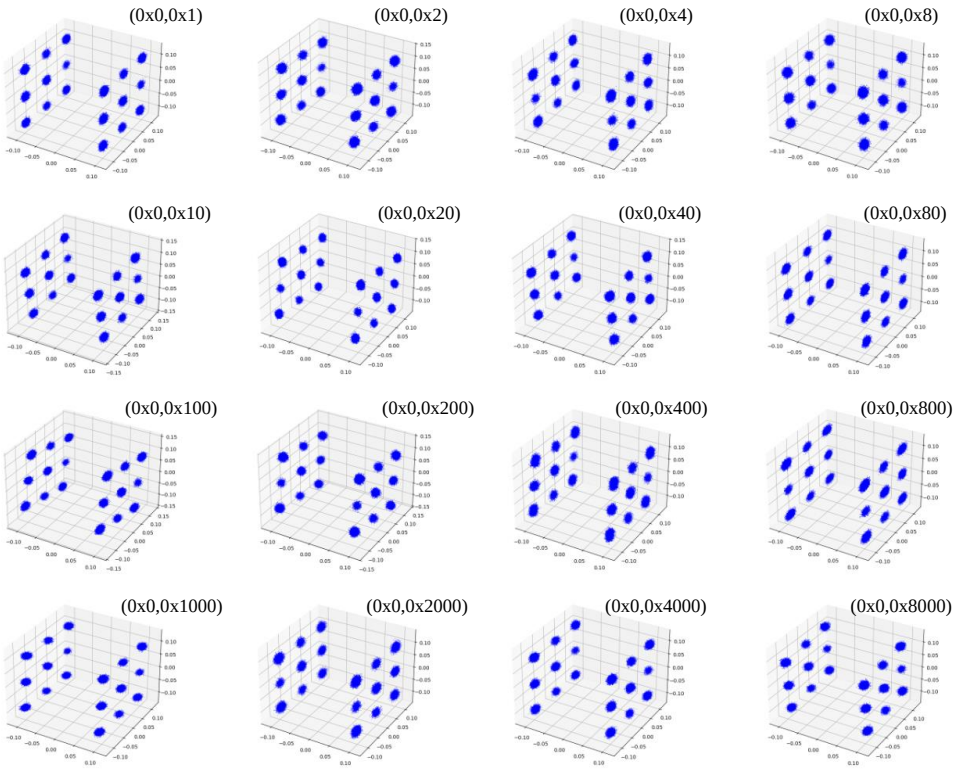

Figure 13: The results of PCA visualization for one-bit differential dataset

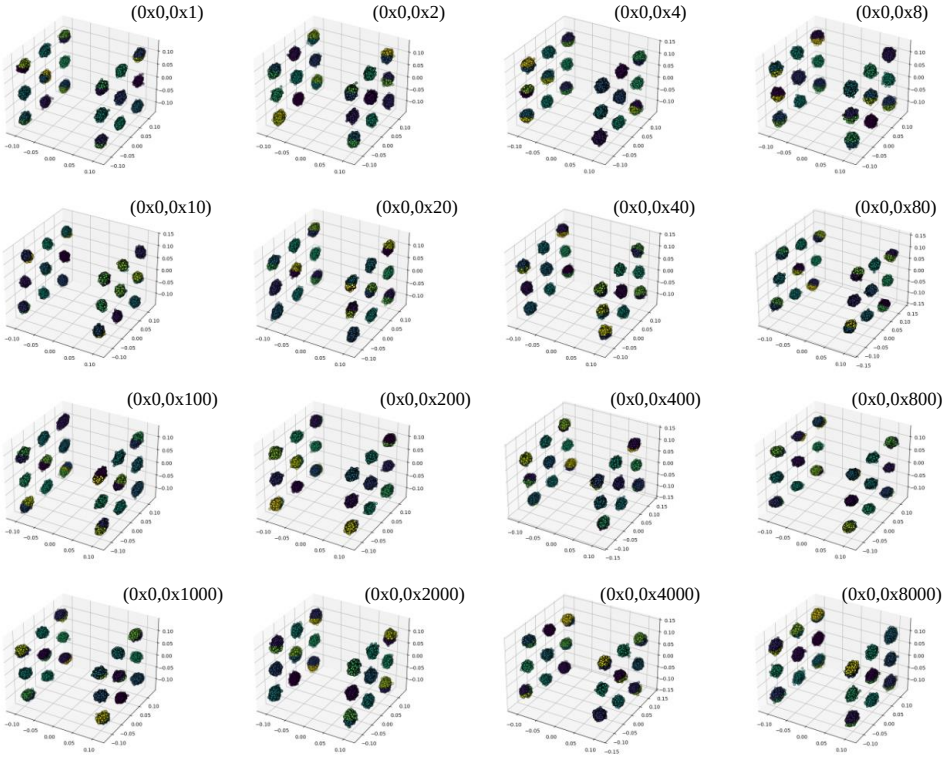

Figure 14: Visualization results of Kmeans clustering against one-bit differential dataset

Table 12: 11-round SIMECK evalutation results and training results

Difference	Silhouette Score	Acc
(0x0,0x00000001)	0.899	0.907
(0x0,0x00000002)	0.908	0.877
(0x0,0x00000004)	0.915	0.877
(0x0,0x00000008)	0.900	0.877
(0x0,0x00000010)	0.904	0.907
(0x0,0x00000020)	0.904	0.877
(0x0,0x00000040)	0.901	0.876
(0x0,0x00000080)	0.913	0.876
(0x0,0x00000100)	0.910	0.877
(0x0,0x00000200)	0.913	0.877
(0x0,0x00000400)	0.913	0.877
(0x0,0x00000800)	0.900	0.877
(0x0,0x00001000)	0.907	0.877
(0x0,0x00002000)	0.913	0.877
(0x0,0x00004000)	0.912	0.877
(0x0,0x00008000)	0.907	0.877

Analysis of the results revealed that when the active bits were positioned on the right side of the differential, satisfying the candidate differential conditions defined in [28], all 16 candidate differential pairs exhibited highly similar visualization patterns, demonstrated only marginal differences in their silhouette scores, and produced distinguishers with statistically equivalent accuracy rates. Consequently, in building an 11-round SIMECK related-key neural distinguisher, any of the 16 candidate differential pairs could be selected, yielding conclusions consistent with [28]. This further validates the effectiveness of the algorithm and provides empirical support for the dataset construction of related-key neural differential distinguishers.

6 CONCLUSION

In this paper, we propose a novel framework for constructing related-key neural differential distinguishers that synergizes traditional differential cryptanalysis with deep learning. The framework systematically optimizes three critical components: dataset construction, differential path optimization, and network architecture design. Applied to DES and PRESENT algorithms, our framework achieves superior performance compared to existing approaches.

Specifically, the constructed distinguishers surpassed results of [18] in both accuracy and round coverage, breaking previous records for the highest-round neural distinguishers under single-key conditions. These improvements stemmed from a dual analytical approach: cryptographic evaluation of differential propagation via algorithmic structures and key schedules, complemented by machine learning-driven dataset quality assessment using kernel PCA and K-means clustering. By correlating feature-space separability with distinguisher accuracy, we bridged cryptographic intuition and data-driven interpretability, validating that high-quality differentials exhibited strong intra-cluster cohesion and inter-cluster separation. The work underscores the potential of deep learning in advanc-

ing cryptanalysis and provides a scalable methodology for future research in intelligent cryptographic evaluation.

Looking ahead, this framework lays the groundwork for automated cryptanalysis tools capable of evaluating diverse cryptographic algorithms. Future directions will include integrating advanced techniques like UMAP or t-SNE for refined feature visualization, exploring semi-supervised learning to reduce data dependency, and extending the methodology to ARX or AES-like ciphers. By merging classical cryptanalysis with modern machine learning, this work not only enhances the efficacy of neural distinguishers but also advances their interpretability, offering a paradigm shift in block cipher security analysis.

acknowledgement This work was supported by National Natural Science Foundation of China (Grant Nos. 62206312).

References

[1] Standard D E. Data encryption standard[J]. Federal Information Processing Standards Publication, 1999, 112(3).

[2] Daemen J, Rijmen V. The design of Rijndael[M]. New York: Springer-verlag, 2002.

[3] Bogdanov A, Knudsen L R, Leander G, et al. PRESENT: An ultra-lightweight block cipher[C]. Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9. Springer Berlin Heidelberg, 2007: 450-466.

[4] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521(7553): 436-444.

[5] Bengio Y, LeCun Y, Hinton G. Deep learning for AI. Communications of the ACM, 2021, 64(7): 58-65.

[6] Gohr A. Improving attacks on round-reduced speck32/64 using deep learning[C]. Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39. Springer International Publishing, 2019: 150-179.

[7] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[8] Chen Y, Shen Y, Yu H, et al. A new neural distinguisher considering features derived from multiple ciphertext pairs[J]. The Computer Journal, 2023, 66(6): 1419-1433.

[9] Hou Z, Ren J, Chen S. Improve neural distinguisher for cryptanalysis[J]. Cryptology ePrint Archive, 2021.

[10] Lu J, Liu G, Sun B, et al. Improved (related-key) differential-based neural distinguishers for SIMON and SIMECK block ciphers[J]. The Computer Journal, 2024, 67(2): 537-547.

[11] Bao Z, Lu J, Yao Y, et al. More insight on deep learning-aided cryptanalysis[C]//International conference on the theory and application of cryptology and information security. Singapore: Springer Nature Singapore, 2023: 436-467.

[12] Wang G, Wang G, Sun S. Investigating and Enhancing the Neural Distinguisher for Differential Cryptanalysis[J]. IEICE TRANSACTIONS on Information and Systems, 2024, 107(8): 1016-1028.

[13] Jain A, Kohli V, Mishra G. Deep learning based differential distinguisher for lightweight block ciphers[J]. arxiv preprint arxiv:2112. 05061, 2021.

[14] Bao Z, Guo J, Liu M, et al. Enhancing differential-neural cryptanalysis[C]. International conference on the theory and application of cryptology and information security. Cham: Springer Nature Switzerland, 2022: 318-347.

[15] Zhang L, Wang Z. Improving differential-neural cryptanalysis[J]. Cryptology ePrint Archive, 2022.

[16] Seok B, Chang D, Lee C. A novel approach to construct a good dataset for differential-neural cryptanalysis[J]. IEEE Transactions on Dependable and Secure Computing, 2024.

[17] Zhao M, Zhong S, Fu X, et al. Deep residual shrinkage networks for fault diagnosis[J]. IEEE Transactions on Industrial Informatics, 2019, 16(7): 4681-4690.

[18] Zhang L, Wang Z, Chen Y. Improving the accuracy of differential-neural distinguisher for DES, chaskey, and PRESENT[J]. IEICE TRANSACTIONS on Information and Systems, 2023, 106(7): 1240-1243.

[19] Bellini E, Gerault D, Hambitzer A, et al. A cipher-agnostic neural training pipeline with automated finding of good input differences[J]. IACR Transactions on Symmetric Cryptology, 2023, 2023(3): 184-212.

[20] Gohr A, Leander G, Neumann P. An assessment of differential-neural distinguishers[J]. Cryptology ePrint Archive, 2022.

[21] Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems[J]. Journal of CRYPTOLOGY, 1991, 4: 3-72.

[22] Biham E. New types of cryptanalytic attacks using related keys[J]. Journal of Cryptology, 1994, 7: 229-246.

[23] Biryukov A, Khovratovich D. Related-key cryptanalysis of the full AES-192 and AES-256. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. ASIACRYPT 2009, 1-18

[24] Biham E, Dunkelman O, Keller N. A related-key rectangle attack on the full KASUMI. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. ASIACRYPT 2005, 443-461

[25] LeCun Y, Boser B, Denker J S, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4): 541-551.

[26] Baksi A, Baksi A. Machine learning-assisted differential distinguishers for lightweight ciphers[J]. Classical and Physical Security of Symmetric Key Cryptographic Algorithms, 2022: 141-162.

[27] Yang G, Zhu B, Suder V, et al. The simeck family of lightweight block ciphers[C]. International workshop on cryptographic hardware and embedded systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015: 307-329.

[28] Wang G, Wang G. Enhanced related-key differential neural distinguishers for SIMON and SIMECK block ciphers[J]. PeerJ Computer Science, 2024, 10: e2566.