Deep Learning-Based Neural Distinguisher for Format-Preserving Encryption Scheme FF3

Dukyoung Kim, Hyunji Kim, Kyungbae Jang, and Hwajeong Seo*

Hansung University, Seoul, South Korea
{dudejrdl123, khj1594012@, starj1023, hwajeong84}@gmail.com

Abstract

Distinguishing data that satisfies the differential characteristic (input/output difference) from random data is called a distinguisher attack (and this can be utilized in differential attacks). If the key can be inferred by analyzing the output difference according to the input difference, the encryption algorithm is designed to be insecure. At CRYPTO'19, Gohr presents the first deep learning-based distinguisher for round-reduced SPECK. Building upon Gohr's work, various works have been conducted. Among many other works, inspired by Baksi et al.'s work presented at DATE'21, we propose the first neural distinguisher using differences on format-preserving encryption (FPE) scheme. In a nutshell, our work achieves valid accuracy (0.98 in 8 rounds on the number domain and 0.55 in 2 rounds on the lowercase domain) with 08 can be achieved in both domains input difference model. Our work utilizes the differential characteristics employed in the classical distinguisher of presented in Dunkelman et al.'s ePrint'20 paper. They use SKINNY as the encryption algorithm for, whereas we employ a standard implementation with AES encryption. Nevertheless, it is worth noting that the $0||K$ input difference remains independent of the internal encryption function. This result demonstrates the validity of our distinguisher, indicating its applicability to various variants of.

Keywords: Deep Learing, Neural Distinguisher, Advanced Encryption Standard(AES)

1 Introduction

Differential cryptanalysis [1] is one of the primary cryptanalysis techniques. If it is possible to predict the key by analyzing the differential characteristic, the cryptographic algorithm can be considered insecurely designed. Distinguishing data that satisfies differential characteristics (input/output differentials) from random data is referred to as a distinguisher attack, which is more powerful than an exhaustive search.

Recently, with the development of deep learning, various studies on deep learning-based distinguishers have been presented [2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. Deep learning is well-suited for probabilistically distinguishing data that satisfies differential characteristics, as it has the capability to make probabilistic predictions on data. For this reason, many studies on neural distinguisher are being conducted, but research on deep learning-based distingusher for FPE scheme, has not yet been conducted. In this work, for the first time, we propose a neural distinguisher based on deep learning for considering input differences. Significantly, our results demonstrate that the deep learning-based discriminator is well-suited for format-preserving encryption schemes as well.

The 7th International Conference on Mobile Internet Security (MobiSec'23), Dec. 19- 21, 2023, Okinawa, Japan, Article No. 36
*Corresponding author: Department of IT Convergence Engineering, Hansung University, Seoul, 02876, Republic of Korea, Tel: +82-02-760-8033

1.1 Our Contribution

1.1.1 First neural distinguisher for

We propose the first neural distinguisher for. Our neural distinguisher works successfully in the number and lowercase domains, and can be effectively utilized for cryptanalysis using differential characteristics.

1.1.2 Neural distinguisher that can be used for distinguishing attacks against variants of

While format-preserving encryption includes an encryption function, the presence of differential characteristics remains independent of the specific encryption function. Consequently, our neural distinguisher can be effectively employed for distinguisher attacks targeting various variants of.

1.2 Organization

The remainder of this paper is organized as follows. In Section 2, the backgrounds of the format-preserving encryption and the neural distinguisher. In Section 3, we present the neural distinguisher for. In Section 4, we evaluate and analyze the performance of our neural distinguisher for. Finally, Section 5 concludes the paper.

2 Background

2.1 Format Preserving Encryption

Unlike general block ciphers, FPE [12] is an encryption scheme that ensures that the form of plaintext remains intact in the ciphertext. That is, when plaintext in a certain domain (number or character) is encrypted, the ciphertext also belongs to the same domain and has the same length. is a cipher designated as a NIST standard among FPE. It has 8 rounds, and its block size and key size of 32-bit and 128-bit, respectively. It is designed as a Feistel structure, and an encryption function (e.g., AES) is used as an internal round function. The choice of encryption function within the FPE framework can be altered [13, 14]. That is, there are various variants of the structure of FPE. FPE is particularly useful in cases where data format is critical for functionality or compatibility with existing systems. For example, when encrypting credit card numbers stored in a database, FPE can be used to ensure that the encrypted value remains similar to a valid credit card number, enabling processing by payment systems without modifications. Additionally, by encrypting data in a way that preserves its original format, the data using FPE can be seamlessly integrated with existing applications and databases that expect data to conform to a specific format.

2.2 Artificial Neural Network

Artificial neural networks [15] consist of multiple layers, each composed of multiple neurons. Neurons calculate their final values by summing the weighted values from the previous layer and passing them through an activation function. This process is repeated for each layer, starting from the input layer. The network learns by minimizing the difference between the predicted output and the actual labels using a loss function (e.g. Binary Crossentropy, Categorical Crossentropy, Mean Squared Error, etc.). In this process, an optimization function (e.g.

Stochastic gradient descent (SGD), RMSprop, Adam) is used for effective minimization. Once trained, the network can predict using its trained weights. A well-trained network can make robust predictions even on untrained data, and the design goal is to create such a robust neural network.

2.3 Differential Characteristic

Differential cryptanalysis [1] is a representative cryptanalysis method of block ciphers. The input difference (δ) is the XOR between the plaintext pairs (P_0, P_1), and the output difference (Δ) is the XOR between the ciphertext pairs. As in Equation 1, C_0 and C_1 are the results of encrypting (E) P_0 and P_1 , respectively. The output difference (Δ) can be obtained by XORing C_0 and C_1. Here, a differential characteristic means a pair of input and output differences (δ, Δ). In the case of an ideal cipher, when plaintext with any input difference is encrypted, the output difference should be uniform (like random). A weak cryptographic algorithm has a certain output difference corresponding to an input difference. If the probability of satisfying an output difference for an input difference is greater than the random probability, the ciphertext can be distinguished from the random. These characteristics have remained even when encryption is performed and can be inferred probabilistically.

$$\begin{aligned} &P_1 = P_0 \oplus \delta, \\ &C_0 = E(P_0), C_1 = E(P_1), \\ &\Delta = C_0 \oplus C_1 \end{aligned} \tag{1}$$

2.4 Neural Network-based Distinguisher for Differential Cryptanalysis

Deep learning is a good solution for distinguisher attacks, as it can probabilistically satisfy specific output differences for given input differences. Consequently, the neural distinguisher performs probabilistic prediction on data applied to distinguisher attacks using differential characteristics. Most of the ongoing works of neural distinguishers are derived from [2], and they focus on target ciphers and input differences. In [2], proposed at CRYPTO 2019, the first neural distinguisher is proposed for round-reduced `SPECK32/64`. Their neural distinguisher successfully distinguish cryptographic data from random data up to 7 rounds, and extended up to 8 rounds through transfer learning. In [3], two distinguisher models considering multi-input differential and single differential are presented. And the target ciphers are `GIMLI`, `ASCON`, `KNOT`, and `Chaskey`. The proposed MLP-based neural distinguisher successfully distinguish 8-round `GIMLI`, 3-round `ASCON`, 10/12-round `KNOT` (256/512-bit), and 4-round `Chaskey`. In addition, many works [9, 8, 10, 11, 4] on various cryptographic and differential characteristics are being conducted, focusing on `SPECK`.

3 Neural distinguisher for `FF3`

In this paper, we present a neural distinguisher specifically designed for the FPE scheme for the first time.

3.1 Dataset

Figure 1 shows the process of generating the dataset using a single input difference. First, random plaintexts (P_0 and P_1) are generated. Since we have to create a plaintext pair that

satisfies the input difference, P_0 is XORed with δ (input difference) to obtain plaintext P_2. Then, the ciphertexts C_0, C_1 and C_2 are calculated by encrypting the plaintexts P_0, P_1, and P_2. Here, C_0 and C_1 are ciphertexts obtained by encrypting random plaintexts that do not satisfy a differential characteristic. We assign the label 0 to the result of concatenating the two values $(C_0||C_1)$ indicating random data. On the other hand, C_0 and C_2 represent ciphertexts for plaintext that satisfy δ (input difference). Thus, since the concatenated value $(C_0||C_{C_2})$ corresponds to cipher data that satisfies Δ (output difference) with a certain probability, we assign the label 1 indicating cipher. Plaintext and ciphertext used in the encryption process are selected from the number domain (`0` to `9`) or lowercase letters domain (`a` to `z`). In addition, the dataset is consist of bits of ciphertext pairs ($C_0||C_1$ or $C_0||C_2$) . Finally, due to the characteristic of FPE, we use the input difference `0x0`$||K$ (K is a hexadecimal number ranging from `0x0` to `0xF`). According to Equation (3) in [16], when `0x0`$||K$ is used, the probability of differential is higher. So an experiment is conducted on the input differences presented in [16] (`0x0`$||K$). In addition, since these input differences are independent of the round function (encryption function), they can be used in all implementations of.

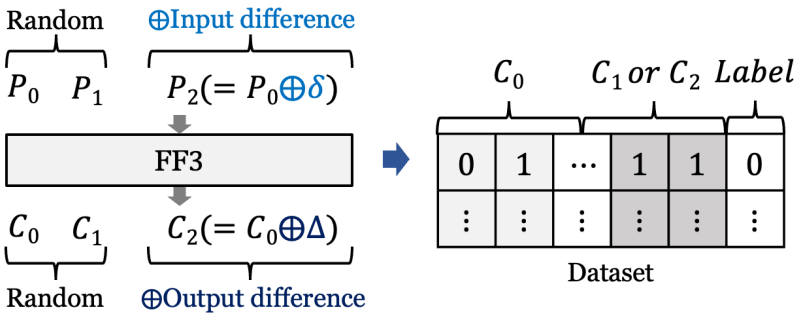

Figure 1: Dataset with one input difference.

3.2 Architecture and Training

Figure 2 shows the diagram of using one input difference. receives a ciphertext pair (random or differential) and classifies it into random and cipher data. First, each bit of the ciphertext pair in the dataset is assigned to each neuron of the input layer. Then, the output of the input layer passes through the hidden layer. In the output layer, a final value between 0 and 1 is calculated by applying a sigmoid activation function. Next, the loss of the final value and the actual value (0 or 1) is calculated. If training to distinguish input data is performed correctly, our model can work as a neural distinguisher for. To work as a valid distinguisher, it must achieve an accuracy greater than 0.5, which is a random probability.

Table 1 shows the hyperparameter of . Epoch is set to 15, and a dense layer with all nodes fully connected is used. performs binary classification because it needs to distinguish between differential data and random data. Thus, binary cross-entropy is used as the loss function. Plus, the Adam optimization function, known for its excellent performance, is employed in our model. For more sophisticated learning, the learning rate of the optimization function is adjusted during training (the learning rate starts at 0.001 and decreases to 0.0001).

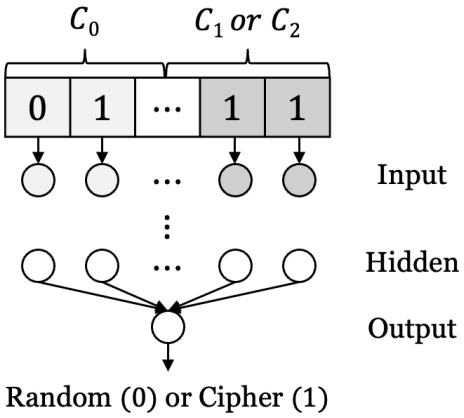

Figure 2: System diagram of .

Table 1: Hyperparameters of .

Hyperparameters	Descriptions
Epochs	15
Loss function	binary cross-entropy
Optimizer	Adam(0.001 to 0.0001) (Learning rate decay)
Activation function	ReLu (Hidden), Sigmoid (Output)
Batch size	32
Hidden layers	4 hidden layers with 128 units
Parameters	74,497

4 Evaluation

4.1 Experiment Environment

This experiment is performed on Google Colab, a cloud computing platform supporting Ubuntu 20.04.5 LTS and Tesla T4 (GPU) 12GB RAM. As the programming environment, TensorFlow 2.12.0 and Python 3.9.16 are used.

4.2 Result for One Input Difference

Table 2 shows the result of according to input difference. In the number domain, when `08` is used as the input difference, can distinguish data up to 8 rounds, and a high accuracy of 0.98 is achieved. However, when using other input differences, relatively low accuracy is achieved compared to `08`. In the lowercase domain, can distinguish data up to 2 rounds due to the increased number of cases for plaintext and ciphertext. It achieves an accuracy of 0.554 for `08`, which is lower compared to the number domain. The experiment on `01` shows a 0.01 lower accuracy compared to the `08` case. The reason for this result is that, as mentioned in [16], it has differential characteristics expected when `08` is used as an input difference. Through this experiment, it is confirmed that data with output differences for `0x0`$||K$ input differences can be predicted with high probability.

Table 2: Result of according to input difference.

	Number (8-round)				Lowercase (2-round)			
	Training	Validation	Test	Reliability	Training	Validation	Test	Reliability
01	0.629	0.624	0.623	0.123	0.545	0.544	0.543	0.043
02	0.829	0.825	0.825	0.325	0.552	0.548	0.545	0.045
03	0.783	0.769	0.771	0.271	0.52	0.514	0.513	0.013
04	0.761	0.756	0.757	0.257	0.523	0.52	0.517	0.017
05	0.773	0.752	0.747	0.247	0.539	0.538	0.537	0.037
06	0.758	0.748	0.75	0.25	0.523	0.519	0.523	0.023
07	0.756	0.739	0.74	0.24	0.532	0.529	0.529	0.029
orange!1508	orange!150.987	orange!150.976	orange!150.977	orange!150.477	green!150.556	green!150.554	green!150.554	green!150.054
09	0.962	0.942	0.941	0.441	0.547	0.543	0.549	0.049
0A	0.969	0.953	0.951	0.451	0.538	0.534	0.532	0.032
0B	0.97	0.965	0.966	0.466	0.53	0.526	0.522	0.022
0C	0.97	0.959	0.959	0.459	0.538	0.536	0.539	0.039
0D	0.968	0.965	0.966	0.466	0.532	0.524	0.518	0.018
0E	0.964	0.963	0.963	0.463	0.549	0.549	0.551	0.051
0F	0.965	0.939	0.941	0.441	0.528	0.524	0.524	0.024

In a work [16] of the classical distinguisher for, the authors use SKINNY as an internal encryption function. On the other hand, the we used is the default implementation using AES. The distinguisher attack using the input differences 0x0$||K$ succeeds despite the internal encryption function being changed. In addition, in our results, it can be also confirmed that the accuracy for 08 is higher and the accuracy for 01 is lower, relatively. This result seems to have come from the fact that the differential characteristic of FPE is independent of the inner encryption function. Thus, we believe that our neural distinguisher structure and differential characteristics may be applicable to other variants as well (naturally, training needs to be performed again according to the data).

5 Conclusion

In this work, we propose the first neural distinguisher for. In the , when 08 is used, a high accuracy of 0.98 is achieved for 8 rounds. In the lowercase domain, up to 2 rounds can be distinguished. Through our experiments, we confirme that the accuracy of 08 is higher, and the accuracy of 01 is low, relatively. In our implementation, a different internal encryption function is used than existing implementations, but the differential characteristic and probabilities appear to be maintained. That is, the input difference 0x0$||K$ remains independent of the inner encryption function. Thus, it seems that our distinguisher may be utilized for variants of. In future work, we will train our wider domains (e.g., uppercase letter, combination of each domain). Also, we plan to optimize the neural distinguisher to achieve high reliability.

6 Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00627, Development of Lightweight BIoT technology for Highly Constrained Devices, 100%).

References

[1] Howard M Heys. A tutorial on linear and differential cryptanalysis. *Cryptologia*, 26(3):189–221, 2002.

[2] Aron Gohr. Improving attacks on round-reduced speck32/64 using deep learning. In *Annual International Cryptology Conference*, pages 150–179. Springer, 2019.

[3] Anubhab Baksi. Machine learning-assisted differential distinguishers for lightweight ciphers. In *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*, pages 141–162. Springer, 2022.

[4] Anubhab Baksi, Jakub Breier, Vishnu Asutosh Dasu, Xiaolu Hou, Hyunji Kim, and Hwajeong Seo. New results on machine learning-based distinguishers. *IEEE Access*, 2023.

[5] Aayush Jain, Varun Kohli, and Girish Mishra. Deep learning based differential distinguisher for lightweight cipher present. *Cryptology ePrint Archive*, 2020.

[6] Reshma Rajan, Rupam Kumar Roy, Diptakshi Sen, and Girish Mishra. Deep learning-based differential distinguisher for lightweight cipher gift-cofb. In *Machine Intelligence and Smart Systems: Proceedings of MISS 2021*, pages 397–406. Springer, 2022.

[7] Girish Mishra, SK Pal, SVSSNVG Krishna Murthy, Ishan Prakash, and Anshul Kumar. Deep learning-based differential distinguisher for lightweight ciphers gift-64 and pride. In *Machine Intelligence and Smart Systems: Proceedings of MISS 2021*, pages 245–257. Springer, 2022.

[8] Yi Chen and Hongbo Yu. A new neural distinguisher model considering derived features from multiple ciphertext pairs. *IACR Cryptol. ePrint Arch.*, 2021:310, 2021.

[9] Adrien Benamira, David Gerault, Thomas Peyrin, and Quan Quan Tan. A deeper look at machine learning-based cryptanalysis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 805–835. Springer, 2021.

[10] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Cryptanalysis of round-reduced Simon32 based on deep learning. *Cryptology ePrint Archive*, 2021.

[11] Tarun Yadav and Manoj Kumar. Differential-ml distinguisher: Machine learning based generic extension for differential cryptanalysis. In *International Conference on Cryptology and Information Security in Latin America*, pages 191–212. Springer, 2021.

[12] William Stallings. Format-preserving encryption: Overview and nist specification. *Cryptologia*, 41(2):137–152, 2017.

[13] Wonyoung Jang and Sun-Young Lee. A format-preserving encryption ff1, ff3-1 using lightweight block ciphers lea and, speck. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 369–375, 2020.

[14] Hyunji Kim, Hyunjun Kim, Siwoo Eum, Hyeokdong Kwon, Yujin Yang, and Hwajeong Seo. Parallel implementation of pipo and its application for format preserving encryption. *IEEE Access*, 10:99963–99972, 2022.

[15] Simon Haykin. *Neural networks and learning machines, 3/E*. Pearson Education India, 2009.

[16] Orr Dunkelman, Abhishek Kumar, Eran Lambooij, and Somitra Kumar Sanadhya. Cryptanalysis of feistel-based format-preserving encryption. *Cryptology ePrint Archive*, 2020.