Journal of Information Processing Vol.31 550–561 (Sep. 2023)

[DOI: 10.2197/ipsjjip.31.550]

Regular Paper

A Deeper Look into Deep Learning-based Output Prediction Attacks Using Weak SPN Block Ciphers

Hayato Kimura[1,2,a)] Keita Emura[2] Takanori Isobe[2,3] Ryoma Ito[2] Kazuto Ogawa[2] Toshihiro Ohigashi[1,2]

Received: December 12, 2022, Accepted: June 5, 2023

Abstract: Cryptanalysis in a blackbox setting using deep learning is powerful because it does not require the attacker to have knowledge about the internal structure of the cryptographic algorithm. Thus, it is necessary to design a symmetric key cipher that is secure against cryptanalysis using deep learning. Kimura et al. (AIoTS 2022) investigated deep learning-based attacks on the small PRESENT-[4] block cipher with limited component changes, identifying characteristics specific to these attacks which remain unaffected by linear/differential cryptanalysis. Finding such characteristics is important because exploiting such characteristics can make the target cipher vulnerable to deep learning-based attacks. Thus, this paper extends a previous method to explore clues for designing symmetric-key cryptographic algorithms that are secure against deep learning-based attacks. We employ small PRESENT-[4] with two weak S-boxes, which are known to be weak against differential/linear attacks, to clarify the relationship between classical and deep learning-based attacks. As a result, we demonstrated the success probability of our deep learning-based whitebox analysis tends to be affected by the success probability of classical cryptanalysis methods. And we showed our whitebox analysis achieved the same attack capability as traditional methods even when the S-box of the target cipher was changed to a weak one.

Keywords: deep learning, block cipher, SPN

1. Introduction

Unlike public-key cryptography, where security is reduced to mathematically difficult problems, the security of symmetric-key cryptography is evaluated based on resistance against classical attacks, e.g., differential, linear, and integral attacks. Specifically, the corresponding statistical characteristics, e.g., differential, linear, and integral characteristics, are searched using automatic evaluation programs and tools, e.g., SAT and MILP solvers. If there is a considerable security margin against these characteristics, the cipher can be considered to be secure against such attacks. Generally, these evaluations require extensive knowledge about the target algorithms and state-of-the-art cryptanalysis techniques because automatic evaluation programs and tools must be customized for different target algorithms and attacks.

Recently, deep learning-based cryptanalysis has received considerable attention in the symmetric-key cryptography field [1], [5], [6], [7], [8], [10], [11], [12], [13], [14], [17], [21], [22], [25], [26], [34], [38], [39], [40]. Remarkably, such attacks do not require knowledge about the target ciphers, except for the algorithm interfaces, i.e., these attacks are feasible even if the adversary does not know the algorithm of the target ciphers. In a blackbox setting, such cryptanalysis is extremely strong, i.e., the adversary can mount an attack with minimum knowledge about the target ciphers and cryptanalysis techniques. Thus, we must consider deep learning-based cryptanalysis when designing symmetric-key ciphers. Recently, Benamira et al. [8] and Chen et al. [12] confirmed that the characteristics explored by Gohr [17] can be employed in classical distinguishing attacks. These results may be used to design deep learning-resistant symmetric-key ciphers; however, this may be insufficient because they did not identify any deep learning specific characteristic in such a manner that it affects the success probabilities of deep learning-based attacks but does not affect those of classical attacks such as linear/differential attacks.

1.1 Motivations

Kimura et al. [26] presented new deep learning-based attacks on block ciphers in a *blackbox setting* where the adversary does not know the algorithm of target ciphers with the exception of the algorithm interfaces, e.g., the key and block sizes. In a blackbox setting, deep learning-based cryptanalysis enables the use of pre-obtained input/output pairs to construct deep learning models for the proposed attacks, e.g., ciphertext prediction and plaintext recovery, and they used these models to evaluate the proposed attacks. Then, they examined the correlations between the evaluation results obtained by deep learning-based cryptanalysis and the characteristics of the target block ciphers. For this purpose,

[1] Tokai University, Minato, Tokyo 108–8619, Japan
[2] National Institute of Information and Communications Technology (NICT), Koganei, Tokyo 184–8795, Japan
[3] University of Hyogo, Kobe, Hyogo 650–0047, Japan
[a)] h_kimura@star.tokai-u.jp

The result presented in Ref. [26] was done when the first author, Hayato Kimura, was a master student at Tokai University, Japan, and was a research assistant at the National Institute of Information and Communications Technology (NICT), Japan. This work is an extension of Ref. [26]. See Section 1.2 for details.

they used a *whitebox analysis* technique in the evaluation phase using deep learning models. The whitebox analysis explores the relationship between the ability of deep learning-based attacks and classical attacks, e.g., linear/differential attacks; therefore, it may be possible to clarify the correlations between the evaluation results obtained by deep learning-based cryptanalysis and the characteristics of target block ciphers.

To obtain highly accurate results from the whitebox analysis in a blackbox setting, Kimura et al. performed comprehensive analyses using all input/output pairs; thus, it is not appropriate to target reduced-round block ciphers because they have the same block size as the original block ciphers (e.g., 64 or 128 bits). For this reason, they first focused on toy block ciphers with a small block size (e.g., 16-bit block variants of PRESENT [9] called small PRESENT-[4]) and performed the whitebox analysis against these toy block ciphers as preliminary experiments. Based on the preliminary experiments, they applied the proposed attacks to block ciphers with large block sizes (e.g., 32 and 64 bits) and considered the whitebox analysis against the target block ciphers. As a result, they demonstrated that the proposed deep learning-based attacks on small PRESENT-[4] have the same attack capabilities as classical attack methods. Moreover, they presented an additional analysis and measured the change in the probability of successful attacks on small PRESENT-[4], which swaps or replaces cryptographic components, e.g., the substitution and permutation layers. They found that swapping or replacing the internal components did not affect the success probability of their classical linear/differential attack, but it does affect the average success probability of the proposed deep learning-based attacks; thus, they obtained a deep learning specific characteristic. Finally, they considered what combination of components would be more resistant to the deep learning-based attacks by identifying such deep learning specific characteristics.

Finding such deep learning specific characteristics is important because exploiting such characteristics can make the target cipher vulnerable to deep learning-based attacks. Therefore, in this paper, we look deeper into such deep learning specific characteristics and extend the previous work [26] to explore clues to facilitate the design of symmetric-key cryptographic algorithms that are secure against deep learning-based attacks.

1.2 Our Contributions

In this paper, we extend the whitebox analysis proposed by Kimura et al. [26] to deeper look into deep learning-based attacks on block ciphers. Specifically, we employ small PRESENT-[4] with two weak S-boxes, which are known to be vulnerable to both differential and linear attacks [28], to clarify the relationship between classical and deep learning-based attacks. Thus, the purpose of this study is to analyze the influence of the differences in the characteristics of three S-boxes (i.e., the original PRESENT S-box and two weak S-boxes) on deep learning specific characteristics.

Deeper Look at Whitebox Analysis for Deep Learning-based Attacks. To perform the whitebox analysis against small PRESENT-[4], we construct two weak small PRESENT-[4] variants by replacing the original PRESENT S-box with known weak S-boxes. We select the weak S-box1 shown in Fig. 6.1 in the literature [28]. This is used in an example of differential cryptanalysis and is known to be vulnerable to differential attacks. We also select the weak S-box2 shown in Fig. 7.1 in the literature [28], which is used in an example of linear cryptanalysis and is known to be vulnerable to linear attacks. This allows us to accurately compare the effectiveness of the deep learning-based attacks proposed by Kimura et al. [26] with that of classical attacks. Their deep learning-based attacks can guess the ciphertext/plaintext from the corresponding plaintext/ciphertext without any knowledge about the keys.

Because of its small internal structures with a block size of 16 bits, we can develop deep learning models by exploiting the maximum number of plaintext/ciphertext pairs, and we can precisely calculate the linear/differential probability for each round. We demonstrate that the deep learning-based attacks are effective against a similar number of rounds in differential and/or linear attacks. Thus, the whitebox analysis against the small PRESENT-[4] with weak S-boxes can be summarized as follows:

- For the small PRESENT-[4] with weak S-box1, we successfully mount output prediction attacks on 11 rounds, and the number of rounds that the differential and linear distinguishing attacks can work is 11 and 9, respectively.
- For the small PRESENT-[4] with weak S-box2, we successfully mount output prediction attacks on 8 rounds, and the number of rounds that the differential and linear distinguishing attacks can work is 7 and 8, respectively.

Note that our attacks realize output predictions (i.e., ciphertext prediction and plaintext recovery) that are considerably stronger than distinguishing attacks even without knowing the algorithm of the target ciphers. From these results, we conclude that the resistance of the target ciphers to differential/linear attacks can affect the success probability of deep learning-based attacks.

1.3 Comparison with Existing Studies

Tables 1 and A·1 compare the proposed and existing deep learning-based attacks [1], [5], [6], [7], [8], [10], [11], [12], [13], [14], [17], [21], [22], [23], [25], [26], [34], [38], [39], [40]. Here, we focus on whether these attacks correspond to a deep learning-based attack in a *blackbox setting* and a deep learning-based attack with the *whitebox analysis*. When an adversary performs a deep learning-based attack in a non-blackbox setting, they must be familiar with the target ciphers and state-of-the-art cryptanalysis techniques. This degrades the original function of a deep learning-based attack such that it does not require any knowledge about the target ciphers and state-of-the-art cryptanalysis techniques, except for the algorithm interfaces. In addition, even if an adversary uses a whitebox analysis in a non-blackbox setting to perform a deep learning-based attack, this should not result in accurate evaluations of the attack. In summary, it is important to perform a deep learning-based attack with whitebox analysis in a blackbox setting. As shown in Table 1, the proposed attacks are deep learning-based output prediction attacks with *whitebox analysis* on vulnerable SPN structures in a *blackbox setting* that extend the method proposed by Kimura et al. [26].

Regarding the whitebox analysis, Danziger et al. presented

Table 1 Comparison of deep learning-based cryptanalysis. OP:=Output Prediction, PR:=Plaintext Recovery, KR:=Key Recovery, DD:=Differential Distinguisher, LD:=Linear Distinguisher, and DLD:=Differential-Linear Distinguisher.

Reference	Cipher (Block size)	Structures	Blackbox Setting	Target	#Round (#Full)	Whitebox Analysis
BSS08 [5]	Serpent (128 bits)	SPN	No	DD	7 (32)	No
AAAA12 [1]	Simplified DES (12 bits)	Feistel	Yes	OP	2 (N/A[2])	No
DH14 [14]	Simplified DES (12 bits)	Feistel	Yes	KR/DD	2 (N/A[2])	No
Gohr19 [17]	Speck32/64 (32 bits)	Feistel	No[1]	KR/DD	12 (22)	Yes
XHY19 [39]	DES (64 bits)	Feistel	Yes	PR	2 (16)	No
CY20 [10]	Speck32/64 (32 bits)	Feistel	No	KR/DD	13 (22)	Yes
CY20 [10]	DES (64 bits)	Feistel	No	KR/DD	8 (16)	Yes
HLZW20 [21]	DES (64 bits)	Feistel	No	KR/LD	5 (16)	No
So20 [34]	Simplified DES (8 bits)	Feistel	No	KR/LD	8 (8)	No
So20 [34]	Speck32/64 (32 bits)	Feistel	No	KR/LD	22 (22)	No
So20 [34]	Simon32/64 (32 bits)	Feistel	No	KR/LD	32 (32)	No
BBDC21 [6]	Gimli-Perm. (384 bits)	SPN	No	DD	8 (48)	No
BBDC21 [6]	ASCON-Perm. (320 bits)	SPN	No	DD	3 (16)	No
BBDC21 [6]	KNOT-256 (256 bits)	Feistel	No	DD	10 (28)	No
BBDC21 [6]	KNOT-512 (512 bits)	Feistel	No	DD	12 (52)	No
BBDC21 [6]	CHASKEY-Perm. (128 bits)	ARX	No	DD	4 (12)	No
BGLMT21 [7]	Speck32/64 (32 bits)	Feistel	No	KR/DD	13 (22)	Yes
BGLMT21 [7]	Simon32/64 (32 bits)	Feistel	No	KR/DD	16 (32)	Yes
BGPT21 [8]	Speck32/64 (32 bits)	Feistel	No	DD	7 (22)	No
BGPT21 [8]	Simon32/64 (32 bits)	Feistel	No	DD	8 (32)	No
CY21 [12]	CHASKEY-Perm. (128 bits)	ARX	No	DLD	4 (12)	Yes
CY21 [12]	DES (64 bits)	Feistel	No	DLD	6 (16)	Yes
CY21 [12]	Speck32/64 (32 bits)	Feistel	No	DLD	7 (22)	Yes
CY21 [13]	Speck32/64 (32 bits)	Feistel	No	KR/DD	13 (22)	Yes
CY21 [13]	Speck48/72 (48 bits)	Feistel	No	KR/DD	12 (22)	Yes
CY21 [13]	Speck48/96 (48 bits)	Feistel	No	KR/DD	12 (23)	Yes
CY21 [11]	DES (64 bits)	Feistel	No	DD	6 (16)	No
CY21 [11]	Speck32/64 (32 bits)	Feistel	No	KR/DD	11 (22)	No
CY21 [11]	PRESENT (64 bits)	SPN	No	DD	7 (31)	No
HRC21 [22]	Simon32/64 (32 bits)	Feistel	No	KR/DD	13 (32)	No
HRC21 [23]	Simon32/64 (32 bits)	Feistel	No	KR/DD	13 (32)	Yes
HRC21 [23]	Simon48/96 (48 bits)	Feistel	No	KR/DD	14 (36)	Yes
HRC21 [23]	Simon64/128 (64 bits)	Feistel	No	KR/DD	13 (44)	Yes
HRC21 [23]	Speck32/64 (32 bits)	Feistel	No	DD	8 (22)	Yes
HRC21 [23]	Speck48/96 (48 bits)	Feistel	No	DD	7 (23)	Yes
HRC21 [23]	Speck64/128 (64 bits)	Feistel	No	DD	8 (27)	Yes
ITYY21 [25]	TWINE (64 bits)	Feistel	No	DD	8 (36)	No
YK21 [40]	Speck32/64 (32 bits)	Feistel	No	DD	9 (22)	Yes
YK21 [40]	Simon32/64 (32 bits)	Feistel	No	DD	12 (32)	Yes
YK21 [40]	GIFT 64 (64 bits)	SPN	No	DD	8 (28)	Yes
WW21 [38]	Speck32/64 (32 bits)	Feistel	No	DD	12 (22)	Yes
WW21 [38]	Speck48/72 (48 bits)	Feistel	No	DD	15 (22)	Yes
WW21 [38]	Speck64/96 (64 bits)	Feistel	No	DD	18 (26)	Yes
KEIIOO22 [26]	PRESENT (64 bits)	SPN	Yes	OP	4 (31)	Yes
KEIIOO22 [26]	AES-like (64 bits)	SPN	Yes	OP	1 (N/A[2])	Yes
KEIIOO22 [26]	TWINE-like (64 bits)	Feistel	Yes	OP	3 (N/A[2])	Yes
This paper	**small PRESENT with weak S-box1**	SPN	**Yes**	**OP**	**11** (31)	**Yes**
This paper	**small PRESENT with weak S-box2**	SPN	**Yes**	**OP**	**8** (31)	**Yes**

[1] Gohr [17] stated that *we consider it interesting that this much knowledge about the differential distribution of round-reduced Speck can be extracted from a few million examples by black-box methods*. However, his *black-box methods* differ from our defined blackbox setting. Thus, we consider his model to be a non-blackbox setting.

[2] The simplified DES, AES-like, and TWINE-like ciphers, which are the modified versions of original ciphers, do not specify the number of full rounds; thus, we described the number of full rounds of these modified versions as 'N/A'.

deep learning-based attacks that predict key bits of 2-round DES from a plaintext/ciphertext set and analyzed the relationship between these attacks and the differential probability [14]. Here, they compared variants employing several types of S-boxes with different properties for differential attacks and concluded that there is a nontrivial relationship between the differential characteristics and success probability of their deep learning-based attacks. However, their results are extremely limited because they targeted a two-round Feistel construction, which is quite insecure even if the component is an ideal function. Consequently, it is un-

Table 2 Original S-box for PRESENT and small PRESENT-[n].

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Table 3 Weak S-box1 which is vulnerable to differential attack.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	6	4	C	5	0	7	2	E	1	F	3	D	8	A	9	B

Table 4 Weak S-box2 which is vulnerable to linear attack.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	F	E	B	C	6	D	7	8	0	3	9	A	4	2	1	5

clear how much the properties of the internal components affects the security of the entire construction. In addition to improving Gohr's deep learning-based attack [17], Benamira et al. [8] and Chen et al. [12] improved the success probability of classical distinguishers using characteristics that are expected to be reacted by Gohr's attack. Their work confirmed whether the characteristics explored by Gohr can be employed in classical distinguishing attacks, and they did not identify any deep learning specific characteristic. Kimura et al. [26] calculated and compared the abilities of the classical and their deep learning-based attacks to investigate the occurrence of their relationship. Then, they identified a deep learning specific characteristic of small-PRESENT-[n]. However, they did not clarify the effect of their proposed attack on multiple cryptographic components that differ in resistance to classical attack methods. To summarize, to the best of our knowledge, Kimura et al. [26] were the first to perform the whitebox analysis but their results did not demonstrate attack capabilities for multiple cryptographic components with different resistances to classical attacks. However, we found that deep learning-based attacks are affected by the success probability of classical attacks. These are the first results that compare the ability of precisely computed classical attacks with that of deep learning-based attacks.

Alani and Hu reported plaintext recovery attacks on DES, 3-DES, and AES [2], [24] that guess plaintexts from given ciphertexts. They claimed that attacks on DES, 3-DES, and AES are feasible with 2^{11}, 2^{11} and 1,741 ($\simeq 2^{10.76}$) plaintext/ciphertext pairs, respectively. However, Xiao et al. doubted the correctness of their results [2], [24] because they could not be reproduced. Baek et al. also pointed this out in the literature [4]. Therefore, these results are excluded in Table 1. Mishra et al. reported that they mounted output prediction attacks on full-round PRESENT; however, this did not work well [16]. In addition, certain results have yielded classical ciphers, e.g., Caesar cipher, Vigenere, and Enigma ciphers [15], [18], [19], [32].

Other machine learning-based analyses have been reported [30], [31]. Tan et al. demonstrated that deep learning can be used to distinguish ciphertexts encrypted by AES, Blowfish, DES, 3-DES, and RC5 [36], to detect the encryption algorithm utilized by the malware. Alshammari et al. attempted to classify encrypted Skype and SSH traffic [3].

Organization. The remainder of this paper is organized as follows. Our target ciphers, i.e., two toy SPN block ciphers, are introduced in Section 2. The deep learning-based output prediction attacks in a blackbox setting proposed by Kimura et al. [26] are introduced in Section 3. Our extended whitebox analysis is discussed in Section 4, which explores the evaluation results obtained by our attacks against two weak toy block ciphers. Finally, the paper is concluded in Section 5.

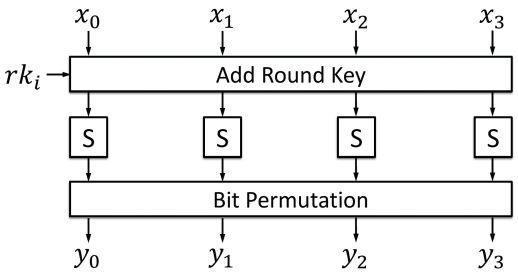


Fig. 1 Round Functions of small PRESENT-[4].

2. Preliminaries

In the following, we introduce an SPN block cipher (PRESENT [9]) and the correspoding toy cipher (small PRESENT-[n] [29]).

PRESENT and small PRESENT-[n]: PRESENT [9] is a lightweight SPN block cipher with a 64-bit block size, 31 rounds, and a key size of either 80 or 128 bits. To analyze PRESENT, a toy model of PRESENT called small PRESENT-[n] [29] has been proposed previously. We show the round function of small PRESENT-[n] in **Fig. 1**. Here, the block size is $4n$; thus, small PRESENT-[16] is equivalent to the original PRESENT. The variant n, which specifies the block size and round key length, allows us to control full diffusion rounds. The S-box has 4-bit input and output. We provide a correspondence table in **Table 2**, which maps $\mathbb{F}_2^4 \to \mathbb{F}_2^4$. The pLayer is described as a bit permutation $P(i)$, which is defined as follows. Note that this is a generalization of that of PRESENT and is equivalent to that of PRESENT when $n = 16$. Moreover, $P(i)$ is used for encryption and $P^{-1}(i)$ is used for decryption.

$$P(i) = \begin{cases} n \times i \bmod (4n-1) & (0 \le i < 4n-1) \\ 4n-1 & (i = 4n-1) \end{cases}$$

$$P^{-1}(i) = \begin{cases} 4 \times i \bmod (4n-1) & (0 \le i < 4n-1) \\ 4n-1 & (i = 4n-1) \end{cases}$$

For key scheduling, the key scheduling algorithm of PRESENT-80, which is a variant of PRESENT with a key length of 80, is executed; furthermore, the $4n$ rightmost bits are used as round keys rk_i.

Weak S-box1 and Weak S-box2: S-box1 (**Table 3**) and S-

box2 (**Table 4**) are known weak S-boxes [28]. S-box1 is vulnerable to a differential attack and S-box2 is vulnerable to a linear attack. We construct two variants of small PRESENT-[4] for our experiments by replacing the original S-box in small PRESENT-[4] with these S-boxes.

3. Methodology

In this section, we present the proposed deep learning-based output prediction attacks in a blackbox setting. To realize the proposed attacks, we construct deep learning models for ciphertext prediction and plaintext recovery. In the following, we first discuss the goals of these attacks and then we describe the construction of the deep learning models.

3.1 Attack Goals

Kimura et al. redesigned the 4-round small PRESENT-[4] by swapping or replacing the internal components, and they used the whitebox analysis technique to examine the relationship between the new target cipher designs and the success probability of the proposed attacks [26]. Consequently, they clarified that swapping or replacing the internal components did not affect the success probabilities of the classical linear/differential attacks, whereas this does affect the average success probabilities of the proposed deep learning-based attacks; thus, they obtained a deep learning specific characteristic.

However, we consider these experimental results to be insufficient to facilitate a discussion of the relationship between deep learning-based and classical attacks, because it remains unclear which vulnerabilities in the cryptographic components affect the accuracy of deep learning-based attacks. To clarify this relationship, we use two block ciphers that are vulnerable to two known classical attacks. Specifically, we apply the classical and deep learning-based attacks to three variants of small PRESENT-[4], as described in Section 2, and observe the differences in the capabilities of these attacks by comparing the success probabilities.

We evaluate the success probabilities of attacks using the following settings.

Known-plaintext attack setting: In this setting, the adversary is given multiple plaintext/ciphertext pairs relating to a single secret key, and the pairs are used as training data to construct a deep learning model.

Blackbox setting: In this setting, the adversary does not have knowledge about the target block ciphers, with the exception of the algorithm interfaces, e.g., the key and block sizes.

Note that the adversary is a very weak cryptographic attacker in both of these settings.

The blackbox setting assumes that the adversary does not know the internal structures of the cipher. In addition, the adversary does not know that the cipher is a permutation. The blackbox setting also assumes that the adversary only knows the input-output format and possesses deep learning knowledge.

Regarding attack settings, a ciphertext-only attack setting, which allows the adversary to obtain only the ciphertext, is the weakest setting. However, information-theoretically no information is provided to the adversary in the setting except for several special cases, e.g., the broadcast setting of RC4 [33]. In fact, the attack in this setting is practically impossible. The known-plaintext attack is the next weakest setting. In this setting, the adversary can obtain some information from the given plaintext/ciphertext pairs and use these pairs for the attacks. Other attack settings, e.g., chosen-plaintext attack setting, require the adversary to possess some knowledge about the ciphertext, and the adversary in this setting is stronger than that in the known-plaintext attack setting. Thus, we employ the known-plaintext attack setting.

In these settings, we determine the adversary's goal to output predictions (i.e., ciphertext prediction and/or plaintext recovery), and we evaluate the success probabilities of these attacks. The ciphertext prediction and plaintext recovery attacks are summarized as follows:

Ciphertext prediction attack: In this attack, the adversary obtains multiple plaintext/ciphertext pairs regarding a secret key, where n is the block size. Then, the adversary predicts a ciphertext of a plaintext not included in the previously given pairs.

Plaintext recovery attack: In this attack, the adversary obtains multiple plaintext/ciphertext pairs regarding a secret key, and then the adversary recovers a plaintext of a ciphertext that is not included in the previously given pairs.

If the ciphertext prediction attack is possible, forgery of the Cipher-based Message Authentication Code (CMAC) is possible. If the plaintext recovery attack is possible, the adversary can obtain the plaintext of any ciphertext without possessing the secret key used for encryption.

3.2 Neural Network and Hyperparameters

Unlike statistical machine learning techniques, e,g., Bayesian inference, deep learning methods allow us to automatically extract features. Deep learning treats nonlinear separable problems; thus, it appears to effectively function to simulate cryptographic functions with nonlinearity. Here, various hyperparameters, such as the initial learning rate, the number of hidden nodes (neurons), and the optimizers, are defined prior to performing the learning phase. Moreover, they are used to construct the corresponding models. These parameters affect the performance of the model; thus, they are optimized using assessment metrics.

In this paper, we consider ciphertext prediction and plaintext recovery as regression problems with supervised learning where plaintext/ciphertext pairs are used as the training data. To this end, we must extract numerous features from the plaintext/ciphertext pairs obtained under the known-plaintext attack; therefore, we employ long short-term memory (LSTM) which is a type of a recurrent neural network [20]. LSTM, which is a general technique for mapping sequences to sequences with neural networks, is frequently used in the field of machine translation [35]. LSTM can realize mapping between sequences in machine translation; thus, we consider that it can also be used to realize mapping between sequences (i.e., between plaintexts and ciphertexts) in encryption/decryption of permutation-based block ciphers. In addition, we consider that numerous features can be extracted from plaintext/ciphertext pairs, i.e., the inputs to our deep learning models, using LSTM, which enables long-term

memory of input sequences. In fact, Kimura et al. confirmed that using LSTM yields better experimental results than a convolutional neural network [26] [*1]. We then optimize the hyperparameters, e.g., the number of hidden nodes, the initial learning rates, the number of hidden layers, and the optimizers. **Table 5** shows the search range for each hyperparameter. During the hyperparameter optimization process, we use different secret keys from those used in the construction of deep learning models because we strictly evaluate the success probabilities of ciphertext prediction and plaintext recovery without depending on secret keys. In the following, the procedure followed to optimize hyperparameters is similar to constructing deep learning models, with the exception of the number of secret keys.

3.3 Deep Learning Models and Their Evaluation

We construct and evaluate deep learning models for ciphertext prediction according to the following procedure. Note that we show the plaintext recovery case in parentheses.

Step 1. The adversary obtains multiple plaintext/ciphertext pairs under the known-plaintext attack. In our experiments, we randomly select multiple plaintexts and generate ciphertexts corresponding to the selected plaintexts.

Step 2. The adversary uses the obtained plaintext/ciphertext pairs as training data to construct deep learning models. Then, the adversary constructs a deep learning model for ciphertext prediction (plaintext recovery) using the plaintexts (ciphertexts) as inputs and the ciphertexts (plaintexts) as the correct outputs.

Step 3. The adversary uses all or part of the remaining plaintexts (ciphertexts), which were not used as training data, to evaluate the constructed deep learning models. The adversary uses these plaintexts (ciphertexts) as the input to the constructed deep learning models. Then, the adversary predicts the unknown ciphertext (plaintext) corresponding to each plaintext (ciphertext).

Step 4. The adversary calculates the percentage of exact match between the predicted ciphertext (plaintext) and the correct ciphertext (plaintext) as the predicted probability.

In our method, all plaintext/ciphertext are represented as an array created from a bitstring. The model represents each predicted bit as a floating point between 0.0 and 1.0. It then rounds off its raw bit output, normalizes it to 0 or 1, and treats it as the final ciphertext or plaintext prediction result.

To evaluate the predicted probabilities, we use 2^x plaintext/ciphertext pairs as training data and 2^y plaintext/ciphertext of the remaining plaintext/ciphertext pairs as test data when applying the proposed attacks against the target block ciphers with a block size of $4n$ bits. It should be noted here that $2^x + 2^y \leq 2^{4n}$. In this case, if the predicted probability is greater than $(2^{4n} - 2^x)^{-1}$, we consider the proposed attacks to be successful. This means that an attacker without knowledge of the target algorithms can predict the output value with a higher probability than a random probability.

As a remark, our attack's priority target is to obtain some properties without knowledge of cryptography in order to establish a method of analysis that can be used by non-experts and we do not emphasize computational complexity. It should be noted that accurate estimation of computational complexity in deep learning-based attack methods is impossible because we cannot use the same criteria for estimating the computational complexity in classical attack methods. In classical attack methods, estimating the computational complexity using the computation time for encryption and the number of queries required for the attack is possible. On the other hand, our proposed method requires an estimate of the computational complexity of training the deep learning model and executing the learned model, in addition to the computation time complexity for encryption. Moreover, it is unknown on what basis to estimate the computational complexity for a fair comparison between deep learning-based attack and classical attack. Additionally, we do not emphasize data complexity. This is because theoretical attacks on modern symmetric-key cryptography allow using unlimited computational resources and unlimited plaintext/ciphertext pairs. In summary, our method does not take care of computational complexity and data complexity. We can estimate data complexity. In our experimental environment, we perform ciphertext prediction and plaintext recovery using as much data as the computational cost allows. But, we cannot estimate the computational complexity in deep learning models though, we show that our method can be attacked in practical time.

4. Deep Dive into Whitebox Analysis Using Weak S-boxes

In this section, we perform the in-depth whitebox analysis to explore the relationship between the respective abilities of the deep learning-based attacks and the classical attacks such as linear/differential attacks against two block ciphers based on our methodology presented in Section 3. We use two toy SPN block ciphers with a block size of 16 bits as a testbed for the proposed attacks. These two SPN block ciphers are the variant of small PRESENT-[4] using weak S-box1, which is vulnerable to differential attacks, and the variant of small PRESENT-[4] using

Table 5 Hyperparameters.

Hyperparameters	Search ranges
Number of hidden nodes	100, 200, 300, 400, 500
Initial value of learning rates	0.0001, 0.001, 0.01
Number of hidden layers	1, 2, 3, 4, 5, 6, 7
Optimizers	SGD, Adam [27], RMSprop [37]

Table 6 Experimental hyperparameters.

Hyperparameters	Values
Number of input layer nodes (i.e., block sizes)	16
Number of output layer nodes (i.e., block sizes)	16
Batch size	250
Number of epochs	100

[*1] Kimura et al. works [26] have conducted experiments on the toy ciphers based on PRESENT (SPN structure, bit permutation), AES (SPN structure, MDS Matrix), and TWINE (Feistel structure) using CNN as well as LSTM. Kimura et al. performed hyperparameter tuning on CNNs as well as LSTM processes and showed the probability of success with tuned parameters. As a result, they concluded that the experimental results for all targeted block ciphers are better when using LSTM compared to when using CNN. The conditions in our experiments are similar to those of Kimura et al. So we think that the results of LSTM shown by Kimura et al. to work better than CNN are valid for the experiments in this paper.

weak S-box2, which is vulnerable to linear attacks. The results of this experiment will provide more insight into the relationship between differential/linear attacks and whitebox analysis.

4.1 Application to Toy Block Ciphers

In this subsection, we apply the proposed attacks to two toy block ciphers, i.e., small PRESENT-[4] with weak S-box1 and small PRESENT-[4] with weak S-box2 as experiments. We first explain the experimental procedure for our whitebox analysis and then demonstrate experimental results to compare the number of rounds that the proposed attacks can be successful to that of existing classical attacks.

Our experimental procedure follows the Kimura et al's whitebox analysis [26]. In our experiments, we implement the pro-

Table 7 Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against small PRESENT-[4] with weak S-box1. We use 2^{15} training data and the remaining 2^{15} testing data. CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Category of Attack	# nodes of hidden layer	# layers of hidden layer	Initial learning rate	Optimizer	Succ. prob.
small PRESENT-[4] with weak S-box1	1	CP	200	1	0.001	Adam	1
		PR	100	3	0.01	Adam	1
	2	CP	500	7	0.001	RMSprop	1
		PR	200	4	0.001	RMSprop	1
	3	CP	200	4	0.001	RMSprop	$2^{-0.01}$
		PR	300	1	0.01	RMSprop	$2^{-0.01}$
	4	CP	500	1	0.01	RMSprop	$2^{-0.01}$
		PR	300	7	0.001	Adam	$2^{-0.97}$
	5	CP	500	3	0.001	Adam	$2^{-5.01}$
		PR	500	7	0.001	Adam	$2^{-4.52}$
	6	CP	200	4	0.01	Adam	$2^{-7.15}$
		PR	500	7	0.001	Adam	$2^{-7.00}$
	7	CP	500	6	0.001	RMSprop	$2^{-9.34}$
		PR	300	3	0.01	Adam	$2^{-9.75}$
	8	CP	400	7	0.001	Adam	$2^{-11.04}$
		PR	500	6	0.001	Adam	$2^{-10.90}$
	9	CP	400	1	0.001	RMSprop	$2^{-12.51}$
		PR	200	2	0.001	Adam	$2^{-12.84}$
	10	CP	500	1	0.001	RMSprop	$2^{-13.90}$
		PR	300	7	0.001	Adam	$2^{-13.54}$
	11	CP	200	6	0.001	RMSprop	$2^{-14.36}$
		PR	500	7	0.001	RMSprop	$2^{-14.66}$
	12	CP	100	3	0.001	Adam	$2^{-15.40}$
		PR	300	2	0.001	RMSprop	$2^{-14.99}$
	13	CP	200	3	0.01	RMSprop	$2^{-15.82}$
		PR	400	6	0.001	RMSprop	$2^{-15.69}$

Table 8 Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against small PRESENT-[4] with weak S-box2. We use 2^{15} training data and the remaining 2^{15} testing data. CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Category of Attack	# nodes of hidden layer	# layers of hidden layer	Initial learning rate	Optimizer	Succ. prob.
small PRESENT-[4] with weak S-box2	1	CP	100	2	0.001	Adam	1
		PR	400	1	0.001	Adam	1
	2	CP	300	5	0.001	Adam	1
		PR	200	5	0.001	Adam	1
	3	CP	300	4	0.001	RMSprop	$2^{-0.01}$
		PR	500	4	0.001	RMSprop	$2^{-0.01}$
	4	CP	500	1	0.01	RMSprop	$2^{-0.03}$
		PR	200	4	0.01	Adam	$2^{-0.97}$
	5	CP	300	3	0.01	Adam	$2^{-5.80}$
		PR	500	5	0.001	RMSprop	$2^{-8.74}$
	6	CP	500	7	0.001	RMSprop	$2^{-10.72}$
		PR	500	7	0.001	Adam	$2^{-11.16}$
	7	CP	300	1	0.001	Adam	$2^{-13.04}$
		PR	500	6	0.001	Adam	$2^{-13.01}$
	8	CP	500	6	0.001	RMSprop	$2^{-14.35}$
		PR	500	2	0.001	Adam	$2^{-14.57}$
	9	CP	100	4	0.001	Adam	$2^{-15.52}$
		PR	500	7	0.0001	Adam	$2^{-15.92}$

Table 9 Maximum differential probabilities of small PRESENT-[4] with weak S-box [*5].

Round	Maximum differential probability		
	small PRESENT-[4] with weak S-box 1	small PRESENT-[4] with weak S-box 2	small PRESENT-[4] with original S-box
1	$2^{-0.6}$	2^{-1}	2^{-2}
2	$2^{-2.8}$	2^{-4}	2^{-4}
3	$2^{-4.2}$	2^{-6}	2^{-8}
4	$2^{-5.6}$	2^{-8}	2^{-12}
5	$2^{-7.0}$	2^{-10}	2^{-14}
6	$2^{-8.4}$	2^{-12}	2^{-16}
7	$2^{-9.8}$	2^{-14}	–
8	$2^{-11.2}$	2^{-16}	–
9	$2^{-12.6}$	–	–
10	$2^{-14.0}$	–	–
11	$2^{-15.4}$	–	–
12	$2^{-16.8}$	–	–

posed attacks using Keras [*2], which is a deep learning library, and we employ TensorFlow as the backend. The following is our experimental environment: 8 Linux machines with 14 NVIDIA GPUs (RTX 2080 SUPER, GeForce GTX 1080 Ti, TITAN Xp, Tesla K40m, and Quadro P600 Mobile). For developing LSTM models by Keras, e.g., `model.add(LSTM(...))`, we specify only `units`, `input_shape`, and `return_sequences` as its arguments [*3]. As an initial setting, we use common experimental hyperparameter values (see **Table 6**). Our experiments involve the following two sub-experiments, i.e., Experiment 1 and Experiment 2.

Experiment 1: In each round, we optimize hyperparameters for the target block ciphers using the proposed attacks, as described in Section 3.2. For our hyperparameter optimization, we use Optuna [*4], which is an automatic optimization tool, and use its default search algorithm. The indication for our hyperparameter optimization is the success probability of ciphertext prediction or plaintext recovery. In other words, the objective function is to calculate the percentage of perfect matches between the predicted ciphertext and the correct ciphertext and to obtain the maximum value of it. In our hyperparameter optimization, we obtain 100 hyperparameter candidates from the plaintext/ciphertext pairs generated by 20 secret keys. From these candidates, we select the optimized hyperparameter with the highest average success probabilities of ciphertext prediction or plaintext recovery. To this end, we use 2^{15} plaintext/ciphertext pairs as training data and the remaining 2^{15} plaintext or ciphertext as testing data; thus, each average success probability is calculated from 2^{15} randomly generated plaintext/ciphertext pairs. If the average success probabilities of ciphertext prediction or plaintext recovery with the optimized hyperparameter, are greater than 2^{-15}, then the number of rounds for finding the optimized hyperparameter is incremented by one; otherwise, the second sub-experiment is executed using the optimized hyperparameter.

Experiment 2: We use randomly generated 100 secret keys and the optimized hyperparameters obtained in Experiment 1 to execute the proposed attacks for ciphertext prediction or plaintext recovery; then, we compute the average success probabilities of ciphertext prediction or plaintext recovery. The secret keys used in Experiment 2 are not the same as those used in Experiment 1. After clarifying the number of attacked rounds for target block ciphers by Experiment 2, we use experimental results and linear/differential probability of the target block ciphers to compare the proposed attacks to the classical linear/differential attacks.

4.2 Experimental Results

Tables 7 and **8** show the results of Experiment 2 using the optimized hyperparameters obtained in Experiment 1. Based on these experimental results, we discuss the extended whitebox analysis against two toy block ciphers, i.e., small PRESENT-[4] with weak S-box1 and small PRESENT-[4] with weak S-box2.

We compare the classical linear/differential and proposed attacks for small PRESENT-[4] with weak S-box1 and weak S-box2. From the experimental results, the proposed attacks succeed up to 11 rounds for ciphertext prediction and plaintext recovery against small PRESENT-[4] with weak S-box1 and the success probability for it is nearly 2^{-14}. Thus we consider that the proposed attacks can be successful for a maximum of 11 rounds on small PRESENT-[4] with weak S-box1. Similarly, the proposed attacks succeed up to 8 rounds for ciphertext prediction and plaintext recovery against small PRESENT-[4] with weak S-box2 and the success probability for it is nearly 2^{-14}. Thus we also consider that the proposed attacks can be successful for a maximum of 8 rounds on small PRESENT-[4] with weak S-box2.

On the other hand, by computing the exact differential and linear probabilities for small PRESENT-[4] with weak S-box1, weak S-box2, and original S-box, we obtain the maximum number of attackable rounds for the classical attacks (see **Tables 9** and **10**) and then compare it with the number of attackable rounds for our deep learning-based attack (see **Table 11**). This comparison shows that the deep learning-based attack capability is equivalent to the higher of the maximum number of attackable rounds for the two classical attacks (differential or linear attack). In other words, we consider that deep learning-based attacks respond to the weaker characteristic when the attacked target cipher is vul-

[*2] https://github.com/keras-team/keras
[*3] https://keras.io/ja/layers/recurrent/
[*4] https://github.com/optuna/optuna

[*5] We obtained the maximum differential/linear probabilities from approximate values by finding differential and linear paths assuming a white box analysis. It means Tables 9 and 10 show rounds less than 2^{-16} for single differential/linear probabilities.

Table 10 Maximum linear probabilities of small PRESENT-[4] with weak S-box [*5].

Round	Maximum linear probability		
	small PRESENT-[4] with weak S-box 1	small PRESENT-[4] with weak S-box 2	small PRESENT-[4] with original S-box
1	$2^{-0.8}$	$2^{-0.8}$	2^{-2}
2	$2^{-3.2}$	$2^{-2.4}$	2^{-4}
3	$2^{-4.8}$	$2^{-4.4}$	2^{-8}
4	$2^{-6.4}$	$2^{-7.2}$	2^{-12}
5	$2^{-8.0}$	$2^{-9.2}$	2^{-16}
6	$2^{-9.6}$	$2^{-10.8}$	–
7	$2^{-11.2}$	$2^{-12.8}$	–
8	$2^{-12.8}$	$2^{-15.6}$	–
9	$2^{-14.4}$	$2^{-17.6}$	–
10	$2^{-16.0}$	–	–

Table 11 Maximum attackable round of small PRESENT-[4] with weak/original S-boxes. CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Reference	Cipher	Maximum attackable round		
		deep learning-based attack	differential cryptanalysis	linear cryptanalysis
This paper	small PRESENT-[4] with weak S-box 1	11 (CP, PR)	11	9
This paper	small PRESENT-[4] with weak S-box 2	8 (CP, PR)	7	8
KEIIOO22 [26]	small PRESENT-[4] with original S-box	5 (CP), 4 (PR)	5	4

nerable to more than one classical attack.

From these experimental results, we consider that deep learning-based output prediction attacks will make it easier for anyone to estimate the resistance to differential and linear attacks, even if they have no knowledge of the target's cryptographic algorithms or cryptanalysis methods.

5. Conclusion

In this study, we have proposed deep learning-based output prediction attacks on two block ciphers with a block size of 16 bits in a blackbox setting. We clarified the following results by examining the relationship between the ability of deep learning-based attacks and classical attacks, e.g., linear/differential attacks:

- Our deep learning-based whitebox analysis achieved the same attack capability as traditional methods even when the S-box of the target cipher was changed to a weak one.
- We found that the success probability of our deep learning-based whitebox analysis tends to be affected by the success probability of classical cryptanalysis methods.
- We believe that output prediction attacks using deep learning will make it easier to estimate the resistance to differential and linear attacks, even without possessing knowledge about the target's cryptographic algorithm or cryptanalysis method.

In the future, we plan to do the followings:

- We will clarify why linear probability contributes more to the average success probability of the proposed deep learning-based attacks than differential probability.
- We will clarify how to feedback on our results for designing deep learning-resistant symmetric-key ciphers.

Acknowledgments This work was supported in part by the JSPS KAKENHI Grant Number 19K11971. The authors would like to thank anonymous reviewers for their invaluable comments and suggestions.

References

[1] Alallayah, K.M., Alhamami, A.H., AbdElwahed, W. and Amin, M.: Applying Neural Networks for Simplified Data Encryption Standard (SDES) Cipher System Cryptanalysis, *Int. Arab J. Inf. Technol.*, Vol.9, No.2, pp.163–169 (2012).

[2] Alani, M.M.: Neuro-Cryptanalysis of DES and Triple-DES, *ICONIP*, pp.637–646 (2012).

[3] Alshammari, R. and Zincir-Heywood, A.N.: Machine learning based encrypted traffic classification: Identifying SSH and Skype, *IEEE CISDA*, pp.1–8 (2009).

[4] Baek, S. and Kim, K.: Recent Advances of Neural Attacks against Block Ciphers, *SCIS* (2020), available from ⟨https://caislab.kaist.ac.kr/publication/paper_files/2020/scis2020_SG.pdf⟩.

[5] Bafghi, A.G., Safabakhsh, R. and Sadeghiyan, B.: Finding the differential characteristics of block ciphers with neural networks, *Information Sciences*, Vol.178, No.15, pp.3118–3132 (2008).

[6] Baksi, A., Breier, J., Chen, Y. and Dong, X.: Machine learning assisted differential distinguishers for lightweight ciphers, *DATE*, pp.176–181 (2021).

[7] Bao, Z., Guo, J., Liu, M., Ma, L. and Tu, Y.: Conditional Differential-Neural Cryptanalysis, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.719 (2021).

[8] Benamira, A., Gérault, D., Peyrin, T. and Tan, Q.Q.: A Deeper Look at Machine Learning-Based Cryptanalysis, *EUROCRYPT*, pp.805–835 (2021).

[9] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y. and Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher, *CHES*, pp.450–466 (2007).

[10] Chen, Y. and Yu, H.: Neural Aided Statistical Attack for Cryptanalysis, *IACR Cryptol. ePrint Arch.*, Vol.2020, p.1620 (2020).

[11] Chen, Y. and Yu, H.: A New Neural Distinguisher Model Considering Derived Features from Multiple Ciphertext Pairs, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.310 (2021).

[12] Chen, Y. and Yu, H.: Bridging Machine Learning and Cryptanalysis via EDLCT, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.705 (2021).

[13] Chen, Y. and Yu, H.: Improved Neural Aided Statistical Attack for Cryptanalysis, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.311 (2021).

[14] Danziger, M. and Amaral Henriques, M.A.: Improved cryptanalysis combining differential and artificial neural network schemes, *ITS*, pp.1–5 (2014).

[15] Focardi, R. and Luccio, F.L.: Neural Cryptanalysis of Classical Ciphers, *ICTCS*, pp.104–115 (2018).

[16] Mishra, G., Murthy, S.V. and Pal, S.: Neural Network Based Analysis of Lightweight Block Cipher PRESENT, Harmony Search and Nature Inspired Optimization Algorithms (2019).

[17] Gohr, A.: Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning, *CRYPTO*, pp.150–179 (2019).

[18] Gomez, A.N., Huang, S., Zhang, I., Li, B.M., Osama, M. and Kaiser, L.: Unsupervised Cipher Cracking Using Discrete GANs, *CoRR*, abs/1801.04883 (2018).

[19] Greydanus, S.: Learning the Enigma with Recurrent Neural Networks, *CoRR*, abs/1708.07576 (2017).
[20] Hochreiter, S. and Schmidhuber, J.: LONG SHORT-TERM MEMORY, *Neural Computation*, Vol.9, No.8, pp.1735–1780 (1997).
[21] Hou, B., Li, Y., Zhao, H. and Wu, B.: Linear Attack on Round-Reduced DES Using Deep Learning, *ESORICS*, pp.131–145 (2020).
[22] Hou, Z., Ren, J. and Chen, S.: Cryptanalysis of Round-Reduced SIMON32 Based on Deep Learning, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.362 (2021).
[23] Hou, Z., Ren, J. and Chen, S.: Improve Neural Distinguisher for Cryptanalysis, *IACR Cryptol. ePrint Arch.*, Vol.2021, p.1017 (2021).
[24] Hu, X. and Zhao, Y.: Research on Plaintext Restoration of AES Based on Neural Network, *Security and Communication Networks*, Vol.2018, pp.6868506:1–6868506:9 (2018).
[25] Idris, M.F., Teh, J.S., Yan, J.L.S. and Yeoh, W.-Z.: A Deep Learning Approach for Active S-Box Prediction of Lightweight Generalized Feistel Block Ciphers, *IEEE Access*, Vol.9, pp.104205–104216 (2021).
[26] Kimura, H., Emura, K., Isobe, T., Ito, R., Ogawa, K. and Ohigashi, T.: Output prediction attacks on block ciphers using deep learning, Zhou, J., Adepu, S., Alcaraz, C., Batina, L., Casalicchio, E., Chattopadhyay, S., Jin, C., Lin, J., Losiouk, E., Majumdar, S., Meng, W., Picek, S., Shao, J., Su, C., Wang, C., Zhauniarovich, Y. and Zonouz, S.A. (Eds.), *Applied Cryptography and Network Security Workshops - ACNS 2022 Satellite Workshops, AIBlock, AIHWS, AIoTS, CIMSS, Cloud S&P, SCI, SecMT, SiMLA, Lecture Notes in Computer Science*, Vol.13285, pp.248–276, Springer (2022).
[27] Kingma, D.P. and Ba, J.: Adam: A Method for Stochastic Optimization, *ICLR* (2015).
[28] Knudsen, L.R. and Robshaw, M.: *The Block Cipher Companion*, Information Security and Cryptography, Springer (2011).
[29] Leander, G.: Small Scale Variants Of The Block Cipher PRESENT, *IACR Cryptol. ePrint Arch.*, Vol.2010, p.143 (2010).
[30] Lee, T., Teh, J.S., Liew, J., Yan, S., Jamil, N. and Yeoh, W.-Z.: A Machine Learning Approach to Predicting Block Cipher Security, *CRYPTOLOGY* (2020).
[31] Lee, T.R., Teh, J.S., Jamil, N., Yan, J.L.S. and Chen, J.: Lightweight Block Cipher Security Evaluation Based on Machine Learning Classifiers and Active S-Boxes, *IEEE Access*, Vol.9, pp.134052–134064 (2021).
[32] Liu, Y., Chen, J. and Deng, L.: Unsupervised Sequence Classification using Sequential Output Statistics, *NIPS*, pp.3550–3559 (2017).
[33] Mantin, I. and Shamir, A.: A Practical Attack on Broadcast RC4, *Fast Software Encryption*, pp.152–164 (2001).
[34] So, J.: Deep Learning-Based Cryptanalysis of Lightweight Block Ciphers, *Security and Communication Networks*, Vol.2020, p.3701067 (2020).
[35] Sutskever, I., Vinyals, O. and Le, Q.V.: Sequence to Sequence Learning with Neural Networks, *NIPS*, pp.3104–3112 (2014).
[36] Tan, C. and Ji, Q.: An approach to identifying cryptographic algorithm from ciphertext, *ICCSN*, pp.19–23 (2016).
[37] Tieleman, T. and Hinton, G.: Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural Networks for Machine Learning*, Vol.4, No.2, pp.26–31 (2012).
[38] Wang, G. and Wang, G.: Improved Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Analysis, *ICICS 2021*, pp.21–38 (2021).
[39] Xiao, Y., Hao, Q. and Yao, D.D.: Neural Cryptanalysis: Metrics, Methodology, and Applications in CPS Ciphers, *IEEE DSC*, pp.1–8 (2019).
[40] Yadav, T. and Kumar, M.: Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Cryptanalysis, *LATINCRYPT 2021*, pp.191–212 (2021).

Appendix

A.1 Hyper-parameters of Existing Deep Learning-based Attacks

Table A·1 shows the hyper-parameters of existing deep learning-based attacks.

Table A·1 Parameters of deep learning-based cryptanalysis. UNK:=Unknown, CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Reference	Target	Network Type	Batch size	Initial Learning rate	Epoch	Number of hidden Layers	Number of neuron
BSS08 [5]	7-round Serpent (128 bits)	RNN	UNK	UNK	UNK	UNK	1024 (SUM)
AAAA12 [1]	Simplified DES (12 bits)	Multilayer feedforward networks	UNK	UNK	7869	2	1024 (SUM)
DH14 [14]	Simplified DES (12 bits)	MLP	UNK	UNK	UNK	UNK	UNK
Gohr19 [17] *6	Speck32/64 (32 bits)	ResNet	5000	drops from 0.001 to 0.0001	200	12	UNK
XHY19 [39]	DES (64 bits)	multi-layer fully connected neural network	1000	UNK	350	4	128, 128, 128, 128
XHY19 [39]	DES (64 bits)	cascade fully connected neural networks	1000	UNK	350	4	128, 256, 256, 128
CY20 [10] *7	Speck32/64 (32 bits)	ResNet	10000	UNK	10	UNK	UNK
CY20 [10] *7	DES (64 bits)	ResNet	10000	UNK	10	UNK	UNK
HLZW20 [21]	3-round DES (64 bits)	customized universal neural network	1000	UNK	$5 * 10^3$	5	UNK
HLZW20 [21]	4-round DES (64 bits)	customized universal neural network	1000	UNK	$5 * 10^4$	5	UNK
HLZW20 [21]	5-round DES (64 bits)	customized universal neural network	1000	UNK	$5 * 10^4$	10	UNK
So20 [34]	Simplified DES (8 bits)	Multilayer feedforward networks	UNK	UNK	5000	5	512, 512, 512, 512, 512
So20 [34]	Speck32/64 (32 bits)	Multilayer feedforward networks	UNK	UNK	5000	5	512, 512, 512, 512, 512
So20 [34]	Simon32/64 (32 bits)	Multilayer feedforward networks	UNK	UNK	5000	5	512, 512, 512, 512, 512
BBDC21 [6]	Gimli-Perm. (384 bits)	MLP *8	UNK	UNK	20	5	296, 258, 207, 112, 160
BBDC21 [6]	ASCON-Perm. (320 bits)	MLP *8	UNK	UNK	20	4	128, 1024, 1024, 1024
BBDC21 [6]	KNOT-256 (256 bits)	MLP *8	UNK	UNK	20	4	128, 1024, 1024, 1024
BBDC21 [6]	KNOT-512 (512 bits)	MLP *8	UNK	UNK	20	4	256, 1024, 1024, 1024
BBDC21 [6]	CHASKEY-Perm. (128 bits)	ResNet *9	5000	UNK	10	UNK	UNK
BGLMT21 [7]	13-round Speck32/64 (32 bits)	UNK	UNK	UNK	UNK	UNK	UNK
BGLMT21 [7]	7, 8, 9, 10-round Simon32/64 (32 bits)	ResNet DenseNet SENet	5000	drops from 0.001 to 0.0001	300	UNK	UNK
BGLMT21 [7]	10, 11-round Simon32/64 (32 bits)	SENet *10	UNK	10^{-4}	10	UNK	UNK
BGPT21 [8] *11,*13	Speck32/64 (32 bits)	LGBM *12	UNK	UNK	UNK	UNK	UNK
BGPT21 [8] *11,*13	Simon32/64 (32 bits)	UNK *12	UNK	UNK	UNK	UNK	UNK
BGPT21 [8] *11,*13	Speck32/64 (32 bits)	2D-CNN *12	UNK	UNK	UNK	UNK	UNK
CY21 [12] *13	CHASKEY-Perm. (128 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [12] *13	DES (64 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [12] *13	Speck32/64 (32 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [13]	Speck32/64 (32 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [13]	Speck48/72 (48 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [13]	Speck48/96 (48 bits)	UNK	UNK	UNK	UNK	UNK	UNK
CY21 [11]	DES (64 bits)	Customized convolution layer based neural network	1000	drops from 0.002 to 0.0001	10	UNK	UNK
CY21 [11]	Speck32/64 (32 bits)	Customized convolution layer based neural network	1000	drops from 0.002 to 0.0001	10	UNK	UNK
CY21 [11]	PRESENT (64 bits)	Customized convolution layer based neural network	1000	drops from 0.002 to 0.0001	10	UNK	UNK
HRC21 [22]	Simon32/64 (32 bits)	ResNet	UNK	UNK	100	UNK	UNK
HRC21 [23]	Simon32/64 (32 bits)	ResNet	10000	UNK	100	UNK	UNK
HRC21 [23]	Simon48/96 (48 bits)	ResNet	10000	UNK	100	UNK	UNK
HRC21 [23]	Simon64/128 (64 bits)	ResNet	10000	UNK	100	UNK	UNK
HRC21 [23]	Speck32/64 (32 bits)	ResNet	10000	UNK	100	UNK	UNK
HRC21 [23]	Speck48/96 (48 bits)	ResNet	10000	UNK	100	UNK	UNK
HRC21 [23]	Speck64/128 (64 bits)	ResNet	10000	UNK	100	UNK	UNK
ITYY21 [25]	TWINE (64 bits)	fully connected neural networks	32	0.001	200	4	512, 512, 512, 512
YK21 [40]	Speck32/64 (32 bits)	MLP	UNK	UNK	UNK	2	1024, 1024
YK21 [40]	Simon32/64 (32 bits)	MLP	UNK	UNK	UNK	2	1024, 1024
YK21 [40]	GIFT 64 (64 bits)	MLP	UNK	UNK	UNK	2	1024, 1024
WW21 [38]	Speck32/64 (32 bits)	ResNet	5000	drops from 0.001 to 0.0001	50	4	UNK
WW21 [38]	Speck48/72 (48 bits)	ResNet	5000	drops from 0.001 to 0.0001	50	4	UNK
WW21 [38]	Speck64/96 (64 bits)	ResNet	5000	drops from 0.001 to 0.0001	50	4	UNK
KEIIOO22 [26]	3-round PRESENT (64 bits) (CP)	LSTM	250	0.001	100	6	300, 300, 300, 300, 300, 300
KEIIOO22 [26]	3-round PRESENT (64 bits) (PR)	LSTM	250	0.001	100	5	300, 300, 300, 300, 300
KEIIOO22 [26]	1-round AES-like (64 bits) (CP)	LSTM	250	0.001	100	4	300, 300, 300, 300,
KEIIOO22 [26]	1-round AES-like (64 bits) (PR)	LSTM	250	0.001	100	4	200, 200, 200, 200
KEIIOO22 [26]	2-round TWINE-like (64 bits) (CP)	LSTM	250	0.001	100	4	400, 400, 400, 400
KEIIOO22 [26]	2-round TWINE-like (64 bits) (PR)	LSTM	250	0.001	100	3	500, 500, 500
This paper	**small PRESENT with weak S-box1**	See Tables 6 and 7					
This paper	**small PRESENT with weak S-box2**	See Tables 6 and 8					

*6 These parameters are used for basic Training pipeline. See details in https://github.com/agohr/deep_speck

*7 https://github.com/AI-Lab-Y/NASA

*8 Baksi et al. [6] tried CNN, LSTM, and MLP-based distinguishers for block ciphers. And they reported that fine-tuned MLP-based distinguishers showed the best accuracy than others.

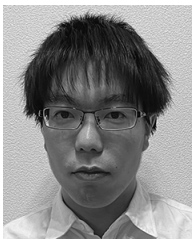
*9 Baksi et al. [6] used the ML model from Gohr's work [17] for CHASKEY-Perm, except the number of epochs is set to 10.

*10 Bao et al. [7] trained the ML model using the Staged Training Method based on 8-round DNN-based distinguisher of SIMON32/64.

*11 https://github.com/AnonymousSubmissionEuroCrypt2021/A-Deeper-Look-at-Machine-Learning-Based-Cryptanalysis

*12 Each work from Sections 5.1, 5.5 and 6 of Benamira et al.'s work [8].

*13 Both Benamira et al.'s and Chen et al.'s works [8], [12] have helped understand Gohr's approach [17]. They propose an efficient distinguisher, but it isn't hyperparameter-focused work.



Hayato Kimura received his B.E. and Master of Information and Telecommunication Engineering degrees from Tokai University in 2019 and 2021, respectively. He joined LINE Corporation in 2021. Since 2023, he has also been a doctoral candidate in the Graduate School of Information Science, University of Hyogo. His current research interests include cryptography, information security, network security, and network protocol. He received the IWSEC 2017 Best Poster Award in 2017. He is a member of IPSJ.

Keita Emura Keita Emura received his M.E. degree from Kanazawa University in 2004. He was with Fujitsu Hokuriku Systems Ltd., from 2004 to 2006. He received his Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology (JAIST) in 2010, where he was with the Center for Highly Dependable Embedded Systems Technology as a postdoctoral researcher in 2010–2012. He has been a researcher with the National Institute of Information and Communications Technology (NICT) since 2012, has been a senior researcher at NICT since 2014, and has been a research manager at NICT since 2021. His research interests include public-key cryptography and information security. He was a recipient of the SCIS Innovation Paper Award from IEICE in 2012, the CSS Best Paper Award from IPSJ in 2016, the IPSJ Yamashita SIG Research Award in 2017, and the Best Paper Award from ProvSec 2022. He is a member of IEICE, IPSJ, and IACR.

Takanori Isobe received his B.E., M.E., and Ph.D. degrees from Kobe University, Japan, in 2006, 2008, and 2013, respectively. From 2008 to 2017, he worked at the Sony Corporation. From 2017 to 2022, he has been an Associate Professor at University of Hyogo. Since 2023, he has been a Professor at University of Hyogo. His current research interests include information security and cryptography.

Ryoma Ito received his B.E. degree from National Defence Academy of Japan in 2009, his M.S. degree from Japan Advanced Institute of Science and Technology in 2015, and his Ph.D. degree from Osaka University in 2019. From 2009 to 2020, he worked at the Japan Air Self-Defense Force, Ministry of Defence, Japan. Since 2020, he has been a senior researcher at National Institute of Information and Communications Technology, Japan. His current research interests include information security and cryptography.

Kazuto Ogawa received his B.E. and Ph.D. degrees from the University of Tokyo in 1987 and 2008, respectively. He joined NHK (Japan Broadcasting Corporation) in 1987. Since 2020, he has been a senior researcher of National Institute of Information and Communications Technology. He has mainly engaged in research and development on video image processing systems and digital rights management systems.

Toshihiro Ohigashi received his B.E. and M.E. degrees from the University of Tokushima, and his Ph.D. degree from Kobe University in 2002, 2004, and 2008, respectively. From 2008 to 2015 he was an Assistant Professor in the Information Media Center, Hiroshima University. From 2015 to 2018, he was a Junior Associate Professor in Tokai University. From 2018 to 2023, he was an Associate Professor in Tokai University. He is currently a Professor in Tokai University. His current research interests include cryptography, information security, and network protocol. He received the SCIS 20th Anniversary Award and SCIS 2013 Innovation Paper Award from ISEC group of IEICE in 2003 and 2014, respectively. He also received the CSS 2014 best paper award from CSEC group of IPSJ in 2014, the IPSJ Yamashita SIG Research Award from IPSJ in 2015, and the Best Paper Award from IEICE in 2015. He is a member of IPSJ and IEICE.