Output Prediction Attacks on Block Ciphers using Deep Learning*

Hayato Kimura[¶, §], Keita Emura[§], Takanori Isobe[†, §], Ryoma Ito[§], Kazuto Ogawa[§], and Toshihiro Ohigashi[¶, §]

[¶]Tokai University, Japan.
[§]National Institute of Information and Communications Technology (NICT), Japan.
[†]University of Hyogo, Japan.

May 15, 2022

Abstract

In this paper, we propose deep learning-based output prediction attacks in a blackbox setting. As preliminary experiments, we first focus on two toy SPN block ciphers (small PRESENT-[4] and small AES-[4]) and one toy Feistel block cipher (small TWINE-[4]). Due to its small internal structures with a block size of 16 bits, we can construct deep learning models by employing the maximum number of plaintext/ciphertext pairs, and we can precisely calculate the rounds in which full diffusion occurs. Next, based on the preliminary experiments, we explore whether the evaluation results obtained by our attacks against three toy block ciphers can be applied to block ciphers with large block sizes, e.g., 32 and 64 bits. As a result, we demonstrate the following results, specifically for the SPN block ciphers: (1) our attacks work against a similar number of rounds that the linear/differential attacks can be successful, (2) our attacks realize output predictions (precisely ciphertext prediction and plaintext recovery) that are much stronger than distinguishing attacks, and (3) swapping or replacing the internal components of the target block ciphers affects the average success probabilities of the proposed attacks. It is particularly worth noting that this is a deep learning specific characteristic because swapping/replacing does not affect the average success probabilities of the linear/differential attacks. We also confirm whether the proposed attacks work on the Feistel block cipher. We expect that our results will be an important stepping stone in the design of deep learning-resistant symmetric-key ciphers.

1 Introduction

Unlike public-key cryptography, where security is reduced to mathematically difficult problems, the security of symmetric-key cryptography is evaluated by 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 by 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 these attacks. Generally, these evaluations require the deep knowledge of target algorithms and state-of-the-art cryptanalysis techniques because automatic evaluation programs and tools must be customized for different target algorithms and attacks.

*This work was done when the first author, Hayato Kimura, was a master student at the Tokai University, Japan, and was a research assistant at the National Institute of Information and Communications Technology (NICT), Japan.

Recently, deep learning-based cryptanalysis has received considerable attention in the symmetric-key cryptography field [1, 5–8, 10–14, 17, 21, 22, 25, 33, 37–39]. Remarkably, this type of attack does not require the knowledge of target ciphers, except algorithm interfaces, i.e., the attack is feasible even if the adversary does not know the algorithm of target ciphers. Such cryptanalysis in a blackbox setting is extremely strong, i.e., the adversary can mount an attack with the minimum knowledge of target ciphers and cryptanalysis techniques. In this context, we must consider deep learning-based cryptanalysis when designing symmetric-key ciphers. However, previous studies have not clarified the features or internal structures that affect the success probabilities. Recently, Benamira et al. [8] and Chen et al. [12] confirmed whether characteristics explored by Gohr [17] can be employed in the classical distinguishing attacks. These results may be used to design deep learning-resistant symmetric-key ciphers; however, it does not seem to be sufficient 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. Such a deep learning specific characteristic is important because it may cause vulnerabilities against deep learning-based cryptanalysis. Thus, the usage of previous results of these attacks to design such deep learning-resistant symmetric-key ciphers is difficult.

1.1 Our Contribution

In this study, we present new deep learning-based attacks on block ciphers in a *blackbox setting* where the adversary does not know the algorithm of target ciphers, except algorithm interfaces such as key and block sizes. In a blackbox setting, deep learning-based cryptanalysis allows us to use pre-obtained input/output pairs to construct deep learning models for our attacks, such as ciphertext prediction and plaintext recovery, and then we can use these models to evaluate these attacks. The next step is to examine correlations between evaluation results obtained by deep learning-based cryptanalysis as well as the characteristics of target block ciphers. For this purpose, we apply a *whitebox analysis* technique to our evaluation phase using deep learning models. The whitebox analysis explores the relationship between the ability of deep learning-based attacks and classical attacks such as linear/differential attacks; therefore, it may be possible to clarify correlations between 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, we should perform comprehensive analyses using all input/output pairs, i.e., it is not appropriate to target the reduced-round block ciphers because they have the same block size as the original block ciphers, e.g., 64 or 128 bits, and we cannot use all input/output pairs. For this reason, we first focus on toy block ciphers with a small block size such as 16 bits and perform the whitebox analysis against these toy block ciphers as preliminary experiments. Based on the preliminary experiments, we apply the proposed attacks to block ciphers with large block sizes, e.g., 32 and 64 bits, and consider the whitebox analysis against the target block ciphers. The details of our contributions in this study are given as follows.

1.1.1 New Deep Learning-based Output Prediction Attacks.

To perform the whitebox analysis against block ciphers with large block sizes, we first focus on two toy SPN block ciphers (16-bit block variants of PRESENT [9] called small PRESENT-[4] and an AES-like cipher called small AES-[4]) and one toy Feistel block cipher (a type-II generalized Feistel structure with 4 branches called small TWINE-[4]). This allows us to accurately compare the effectiveness of the proposed deep learning-based attacks, which guess the ciphertext/plaintext

from the corresponding plaintext/ciphertext without any knowledge of keys with that of classical attacks. Due to the page limitation, our target ciphers, two SPN block ciphers and one Feistel block cipher (and their toy ciphers), are introduced in Appendix A.

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 linear/differential probability for each round. We demonstrate that the proposed attacks are effective against the similar number of rounds as linear/differential attacks. For small PRESENT-[4], we successfully mount output prediction attacks on 4 rounds, while the number of rounds that the differential distinguisher can work is also 4. For small AES-[4] and small TWINE-[4], we can mount prediction attacks on 1 and 3 rounds, while differential distinguishing attacks can reach 2 and 7 rounds, 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 target ciphers. Nevertheless, for small TWINE-[4], the number of rounds that the proposed attacks can be successful is significantly less than that of linear/differential attacks. To clarify this cause, additional studies will be required in future.

Next, based on evaluation results for toy block ciphers, we apply the proposed attacks to the target block ciphers with a block size of 64 bits, i.e., PRESENT [9], AES-like, and TWINE-like ciphers. Consequently, we consider that by increasing the amount of training data, the whitebox analysis against block ciphers with large block sizes can be regarded as equal to or greater than the whitebox analysis against toy block ciphers with a block size of 16 bits; thus, the whitebox analysis against the target block ciphers with large block sizes can be summarized as follows:

- For PRESENT, the maximum number of rounds that the proposed attacks can be successful is at least equal to that of classical linear/differential attacks.
- For AES-like and TWINE-like ciphers, we conjecture that the maximum number of rounds that the proposed attacks can be successful also becomes equal to that of classical linear/differential attacks when the amount of training data increases more.

In addition, we conduct additional experiments with 10,000 trials (rather than 100 trials) to confirm the accuracy of the success probability calculated from the proposed attacks. Consequently, we demonstrate that the additional experiments with a small number of secret keys are sufficient to obtain the best success probability, and therefore the proposed attacks lead to reliable results.

1.1.2 Whitebox Analysis for Deep Learning-based Attacks.

We swap or replace internal components on the toy SPN block cipher, particularly on the 4-round small PRESENT-[4], to investigate the relationship between the internal components and success probability of our deep learning-based attacks, and evaluate the impact of these modifications on the success probability of the prediction attacks. The toy Feistel block cipher, i.e., small TWINE-[4], is excluded from this investigation because Feistel block ciphers generally use the same components for both encryption and decryption algorithms. Consequently, we find that swapping or replacing the internal components significantly affects the average success probabilities of the proposed attacks. It is particularly worth noting that this is a deep learning specific characteristic because component swapping and replacing that we did in this study did not affect success probabilities of linear/differential attacks. We expect that our results will be an important foundation in the design of deep learning-resistant symmetric-key ciphers.

1.2 Comparison with Existing Studies

Due to the page limitation, we give a comparison among the proposed attacks and existing deep learning-based attacks [1, 5–8, 10–14, 17, 21–23, 25, 33, 37–39] in Table B.1 in Appendix B. For comparison, we particularly focused 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, the adversary must be familiar with the target ciphers as well as state-of-the-art cryptanalysis techniques. This degrades the original function of a deep learning-based attack in such a way that it does not require any knowledge of target ciphers and state-of-the-art cryptanalysis techniques, except algorithm interfaces. In addition, even if an adversary uses the 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 the whitebox analysis in a blackbox setting. As shown in Table B.1, the proposed attacks are the first deep learning-based output prediction attacks with *whitebox analysis* on both SPN and Feistel structures in a *blackbox setting*.

1.2.1 Organization.

This paper is organized as follows. The proposed deep learning-based output prediction attacks in a blackbox setting is introduced in Sect. 2. Our whitebox analysis is performed in Sect. 3 that explores the evaluation results obtained by our attacks against three toy block ciphers can be applied to block ciphers with large block sizes. An extended whitebox analysis on small PRESENT-[4] is introduced in Sect. 4. Finally, Sect. 5 concludes this study.

2 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, respectively. In the following, we first discuss the goals of these attacks and then explain the construction of deep learning models and their evaluation.

2.1 Goals of Attack

To date, the relationship between the abilities classical attacks and deep learning-based ones has not been clarified. Here, we focus on clarifying this relationship. We then revisit the common sense in previous works using deep learning-based attacks. The targets of this work are summarized as follows:

1. We clarify the difference in capabilities between the classical and deep learning-based attacks. Specifically, we compare the success probabilities of deep learning-based attacks with those of classical attacks.

2. Swapping or replacing the internal components in the target block ciphers does not affect the success probability of linear/differential cryptanalysis. We clarify how such modifications to cipher's algorithms affect the success probability of deep learning-based attacks.

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, except algorithm interfaces such as key and block sizes.

In both of these settings, the adversary is a very weak cryptographic attacker.

The blackbox setting assumes that the adversary does not know the internal structures of the cipher. In addition, the adversary does not know 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 [31]. 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. The 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 the adversary in the known-plaintext attack setting. Thus, we employ the known-plaintext attack setting.

In these settings, we decide 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 pairs given previously.

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.

2.2 Neural Network and Hyperparameters

Deep learning allows us to automatically extract features unlike statistical machine learning techniques, e.g., Bayesian inference. Deep learning treats nonlinear separable problems; thus, it appears to work well for simulating cryptographic functions with nonlinearity. Hyperparameters such as the initial learning rate, number of hidden nodes (neurons), and optimizers, are defined prior to the learning phase and are used to construct models. These parameters affect model performance; 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 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 recurrent neural networks (RNN) [20]. The LSTM, which is a general technique for mapping sequences to

Table 1: 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 [26], RMSprop [36]

sequences with neural networks, is used in the field of machine translation [34]. As the LSTM can realize the mapping between sequences in machine translation, we consider that it can also realize the 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, by using the LSTM, which enables long-term memory of input sequences. In fact, we have confirmed that the use of the LSTM induces better experimental results than that of the convolutional neural network (CNN), as described in Appendix C for more details. We then optimize hyperparameters, e.g., number of hidden nodes, initial learning rates, number of hidden layers, and optimizers. Table 1 shows the search range for each hyperparameter. During the hyperparameter optimization, 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 to optimize hyperparameters is similar to constructing deep learning models, with the exception of the number of secret keys.

2.3 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.

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

Table 2: Experimental hyperparameters

Hyperparameters	Values
Number of input layer nodes	1
Number of output layer nodes (i.e., block sizes)	16, 32, 64
Batch size	250
Number of epochs	100

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.

3 Whitebox Analysis

In this section, we perform the whitebox analysis to explore the relationship between the ability of deep learning-based attacks and the classical attacks such as linear/differential attacks against three block ciphers based on our methodology presented in Sect. 2. We first use three toy block ciphers with a block size of 16 bits as a testbed for the proposed attacks. Based on these preliminary experiments, we then apply the proposed attacks to block ciphers with large block sizes, such as 32 and 64 bits. Finally, we conduct additional experiments to ensure that our whitebox analysis is accurate.

3.1 Application to Toy Block Ciphers

In this subsection, we apply the proposed attacks to three toy block ciphers, i.e., small PRESENT-[4], small AES-[4], and small TWINE-[4], as preliminary 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.

3.1.1 Experimental Procedure.

In our experiments, we implement the proposed attacks using Keras[1], 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[2]. As an initial setting, we use common experimental hyperparameter values (see Table 2). 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 Sect. 2.2. For our hyperparameter optimization, we use Optuna[3], 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

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

recovery. 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 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 is 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.

3.1.2 Experimental Results.

Table 3 shows the experimental results of Experiment 2 using the optimized hyperparameter obtained in Experiment 1. Based on these experimental results, we discuss the whitebox analysis against three toy block ciphers, i.e., small PRESENT-[4], small AES-[4], and small TWINE-[4].

First, we compare the proposed and classical linear/differential attacks for small PRESENT-[4]. From the experimental results, the proposed attacks succeed up to 5 rounds for ciphertext prediction and up to 4 rounds for plaintext recovery against small PRESENT-[4]. Although the average success probability of ciphertext prediction for the 5-round small PRESENT-[4] is nearly 2^{-15}, the average success probability of plaintext recovery for the 4-round small PRESENT-[4] is sufficiently greater than 2^{-15}. In other words, we consider that the proposed attacks can be successful for a maximum of 4 rounds. On the other hand, from the precisely calculated differential probability of small PRESENT-[4] (see Table D.1 in Appendix D), the maximum number of rounds that the differential attack can be successful is 4. Similarly, based on the precisely calculated linear probability, the maximum number of rounds that a linear attack can be successful is also 4. Therefore, for small PRESENT-[4], the maximum number of rounds that the proposed attack can be successful is equal to that of classical linear/differential attacks.

Next, we compare the proposed and classical linear/differential attacks for small AES-[4]. From Table 3, we evaluate the maximum number of rounds that the proposed attacks can be successful is 1. From the precisely calculated linear/differential probabilities, the maximum number of rounds that the differential attack can be successful is 2, whereas that of the linear attack is 3. Similarly, we compare the proposed attacks and classical linear/differential attacks for small TWINE-[4]. We discovered that the proposed attack can be successful for a maximum of 3 rounds with the differential attack lasting 7 rounds and the linear attack lasting 9 rounds. In summary, for small AES-[4] and small TWINE-[4], the maximum number of rounds that the proposed attacks can be successful is less than that of the classical linear/differential attacks. It should be noted here that the proposed attacks realize much stronger ciphertext prediction and plaintext recovery than the distinguishing attacks of the classical linear/differential cryptanalysis. Nevertheless, for small TWINE-[4], the maximum number of rounds that the proposed attacks can be successful is significantly smaller than that of the classical linear/differential attacks. This cause will be clarified in a future study.

Table 3: Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against three toy block ciphers with a block size of 16 bits. 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]	1	CP	400	5	0.001	Adam	1
		PR	100	2	0.001	RMSprop	1
	2	CP	400	4	0.001	RMSprop	1
		PR	400	1	0.001	Adam	1
	3	CP	300	6	0.001	RMSprop	1
		PR	300	5	0.001	RMSprop	1
	4	CP	300	4	0.01	Adam	$2^{-5.63}$
		PR	300	1	0.01	Adam	$2^{-14.50}$
	5	CP	200	7	0.001	Adam	$2^{-14.08}$
		PR	300	6	0.001	Adam	$2^{-15.73}$
small AES-[4]	1	CP	300	4	0.001	RMSprop	1
		PR	200	4	0.001	RMSprop	1
	2	CP	300	1	0.01	Adam	$2^{-16.02}$
		PR	200	2	0.01	Adam	$2^{-15.00}$
small TWINE-[4]	1	CP	300	3	0.001	RMSprop	1
		PR	300	2	0.001	Adam	$2^{-0.01}$
	2	CP	400	4	0.001	RMSprop	$2^{-0.01}$
		PR	500	3	0.001	RMSprop	$2^{-0.01}$
	3	CP	300	2	0.001	RMSprop	$2^{-10.46}$
		PR	400	2	0.001	RMSprop	$2^{-9.72}$
	4	CP	200	4	0.001	RMSprop	$2^{-14.61}$
		PR	100	1	0.01	RMSprop	$2^{-15.49}$
	5	CP	300	5	0.01	RMSprop	$2^{-15.64}$
		PR	500	4	0.001	RMSprop	$2^{-15.16}$

3.1.3 Whitebox Analysis with the Smaller Amount of Training Data.

To perform the whitebox analysis with the smaller amount of training data against three toy block ciphers (i.e., 1-, 2-, 3-, 4-round small PRESENT-[4], 1-round small AES-[4], and 1-, 2-, 3-round small TWINE-[4]), we conduct additional experiments in the same procedure described above, but we vary the amount of training data in the range of from 2^2 to 2^{14} and use all the remaining plaintexts or ciphertexts as testing data. In these additional experiments, we use the optimized hyperparameters obtained in Experiment 1 (see Table 3).

We show the details of the additional experimental results in Appendix E. Table E.1 shows the minimum amount of training data required for successful ciphertext prediction/plaintext recovery against three toy block ciphers. In addition, Table E.2 shows more detailed results regarding the average success probabilities of ciphertext prediction/plaintext recovery by the proposed attacks against three toy block ciphers with a block size of 16 bits. If the predicted probability is greater

than 2^{-15}, we consider the proposed attacks to be successful[4]. Consequently, we demonstrate successful ciphertext prediction/plaintext recovery with a smaller amount of training data than 2^{15} against three toy block ciphers, with the exception of the 4-round small PRESENT-[4].

3.2 Application to Block Ciphers with Large Block Sizes

In this subsection, we apply the proposed attacks to three block ciphers with large block sizes based on the preliminary experiments as described in Sect. 3.1. To examine the evaluation results obtained by our whitebox analysis against three toy block ciphers can be applied to the target block ciphers with large block sizes, we conduct Experiment 2 in the same procedure as described in Sect. 3.1.1, but we change the block sizes of the target block ciphers, e.g., 32 and 64 bits. In our experiments, we use the optimized hyperparameters obtained in Experiment 1 (see Table 3 in Sect. 3.1.2).

We show the details of the experimental results in Appendix F. Tables F.1 and F.2 show the minimum amount of training data required for successful ciphertext prediction/plaintext recovery against three block ciphers with block sizes of 32 and 64 bits, respectively. We vary the amount of training data in the range of from 2^8 to 2^{17} or from 2^{10} to 2^{19} and use 2^{16} of the remaining plaintexts or ciphertexts as testing data against three toy block ciphers with block sizes of 32 or 64 bits, respectively; thus, if the predicted probability is greater than the threshold derived by equation $(2^{4n} - 2^x)^{-1}$ shown in Sect. 2.3, we consider the proposed attacks to be successful for both cases. In this case, those thresholds are $(2^{32} - 2^8)^{-1}$ to $(2^{32} - 2^{17})^{-1}$ or from $(2^{64} - 2^{10})^{-1}$ to $(2^{64} - 2^{19})^{-1}$. In addition, Tables F.3 and F.4 in Appendix F show more detailed results regarding the average success probabilities of ciphertext prediction/plaintext recovery by the proposed attacks against three block ciphers with block sizes of 32 and 64 bits.

From Tables F.1 and F.3, we report that the average success probabilities of ciphertext prediction/plaintext recovery by the proposed attacks against the target block ciphers with a block size of 32 bits are not zero, excluding the 4-round small PRESENT-[8]. Expressed differently, this fact should indicate that the proposed attacks against the target block ciphers with large block sizes can be successful by simply increasing the amount of training data; thus, we consider that the proposed attack against the target block ciphers with additional rounds could be successful by using more training data than 2^{17}.

From Tables F.2 and F.4, we can confirm that except for the 4-round small PRESENT-[16] and the 3-round small TWINE-[16], the average success probabilities of ciphertext prediction/plaintext recovery by the proposed attacks against the target block ciphers with a block size of 64 bits are not zero. In these cases, we consider that the proposed attacks against the target block ciphers with additional rounds could be successful with more training data than 2^{19}.

As demonstrated by these results, the proposed attacks can be performed regardless of the block size of the target block ciphers by simply increasing the amount of training data. In addition, as the amount of training data increases, the larger the block size, the greater the rate of increase in the success probability (see Tables E.2, F.3, and F.4 for more details). Therefore, we consider that by increasing the amount of training data, the whitebox analysis against block ciphers with large block sizes can be regarded as equal to or greater than the whitebox analysis against toy block ciphers with a block size of 16 bits. As discussed in Sect. 3.1.2, the maximum number of rounds that the proposed attacks can be successful against small PRESENT-[4] is equal to that of the classical linear/differential attacks, while the maximum number of rounds that the proposed attacks can be successful against small AES-[4] and small TWINE-[4] is less than that of the

[4]This assumption is strictly incorrect, but we use it for simple discussion.

classical linear/differential attacks. Nevertheless, we consider that the whitebox analysis against the target block ciphers with large block sizes can be summarized as follows, based on the above consideration:

- For small PRESENT-[16] (i.e., PRESENT), the maximum number of rounds that the proposed attacks can be successful is at least equal to that of the classical linear/differential attacks.
- For small AES-[16] (i.e., AES-like) and small TWINE-[16] (i.e., TWINE-like), we conjecture that the maximum number of rounds that the proposed attacks can be successful also becomes equal to that of classical linear/differential attacks. To clarify the correctness of this conjecture, we should conduct additional experiments with a larger amount of training data than 2^{19}. This will be our future work.

3.3 Accuracy of Experimental Results

In Sect. 3.1, we have presented the experimental results of Experiment 1 with 20 secret keys and Experiment 2 with 100 secret keys. These experimental results may appear to be correct. However, because of the small number of secret keys used in these experiments, we should have an additional discussion to ensure that the experimental results are accurate. To this end, this subsection shows two additional experimental results on the 3-round small TWINE-[4] with 100 secret keys for Experiment 1 and 10000 secret keys for Experiment 2, respectively. The following explains why we chose the 3-round small TWINE-[4] for confirming the accuracy: If we choose a target with a probability of 1 or 2^{-15}, it appears difficult to see how the number of secret keys affects the accuracy. As shown in Table 3, the average success probabilities of ciphertext prediction and plaintext recovery by the proposed attacks in the 3-round small TWINE-[4] are approximately $2^{-10.46}$ and $2^{-9.72}$, respectively. We choose the 3-round small TWINE-[4] as the best target for additional experiments because these probabilities possibly vary significantly if the number of keys affects the accuracy.

3.3.1 Experimental Procedure.

We explain the following two additional experiments, i.e., Experiment 1' and Experiment 2'.

Experiment 1': We use the same procedures as in Experiment 1 to optimize the hyperparameters for the 3-round small TWINE-[4]. Unlike Experiment 1, we use plaintext/ciphertext pairs generated by 100 secret keys rather than 20 secret keys in this experiment. In the hyperparameter optimization, we examine the impact of the number of secret keys used in Experiment 1' on the experimental results.

Experiment 2': We obtain the average success probabilities of ciphertext prediction/plaintext recovery for the 3-round small TWINE-[4] in the same procedures of Experiment 2 using the hyperparameters optimized by Experiment 1 (see Table 3). Unlike Experiment 2, we use the plaintext/ciphertext pairs generated by 10000 secret keys rather than 100 secret keys. In the ciphertext prediction/plaintext recovery, we explore the influence of the number of secret keys used in Experiment 2' on the experimental results.

3.3.2 Experimental Results.

Table 4 shows a comparison of the experimental results in Experiment 1 and Experiment 1' for the 3-round small TWINE-[4]. From the table, in the hyperparameter optimization for ciphertext

Table 4: Comparison of the experimental results in Experiment 1 and Experiment 1' for the 3-round small TWINE-[4].

Category of Attack	# keys	# trials	# nodes of hidden layer	# layers of hidden layer	Initial learning rate	Optimizer	Succ. prob.	Ref.
Ciphertext Prediction	20	100	300	2	0.001	RMSprop	$2^{-11.42}$	Experiment 1
	100	40	300	7	0.001	RMSprop	$2^{-11.26}$	Experiment 1'
Plaintext Recovery	20	100	400	2	0.001	RMSprop	$2^{-7.80}$	Experiment 1
	100	40	100	4	0.001	RMSprop	$2^{-12.82}$	Experiment 1'

Table 5: Comparison of experimental results in Experiment 2 and Experiment 2' for the 3-round small TWINE-[4]. We use the optimized hyperparameters obtained in Experiment 1 (see Table 3).

Attack	# keys	Succ. prob.	Ref.
Ciphertext Prediction	100	$2^{-10.46}$	Experiment 2
	10000	$2^{-10.64}$	Experiment 2'
Plaintext Recovery	100	$2^{-9.72}$	Experiment 2
	10000	$2^{-9.22}$	Experiment 2'

prediction, the highest average success probabilities obtained from Experiment 1 and Experiment 1' are nearly equal, such as $2^{-11.42}$ and $2^{-11.26}$. Conversely, in the hyperparameter optimization for the plaintext recovery, the highest average success probability obtained from Experiment 1 is much higher than that obtained from Experiment 1', such as $2^{-7.80}$ and $2^{-12.82}$. As per these experimental results, optimizing the hyperparameters with a small number of secret keys is sufficient to obtain hyperparameters with the best average success probability; therefore, we consider that the hyperparameter optimization presented in Sect. 3.1 has led to reliable results.

Table 5 shows a comparison of experimental results in Experiment 2 and Experiment 2' for the 3-round small TWINE-[4]. We can see from the table that in both ciphertext prediction and plaintext recoveries, the average success probabilities obtained from Experiment 2 and Experiment 2' are nearly equal , such as $2^{-10.46}$ and $2^{-10.64}$ in the ciphertext prediction and $2^{-9.72}$ and $2^{-9.22}$ in the plaintext recovery. According to these experimental results, the additional experiments with a small number of secret keys are sufficient to obtain the best average success probability; therefore, we consider that the ciphertext prediction/plaintext recovery presented in Sects. 3.1 and 3.2 has led to reliable results.

4 Extended Whitebox Analysis on Small PRESENT-[4]

As shown in Table 3, the average success probability of ciphertext prediction by the proposed attack on the 4-round small PRESENT-[4] is approximately 2^9 times greater than that of plaintext recovery. However, the security of the encryption and decryption is thought to be equivalent in terms of the linear/differential probabilities on small PRESENT-[4]; thus, the experimental result of the proposed attacks on the 4-round small PRESENT-[4] seems contrary to intuition. We speculate that this can be a deep learning specific characteristic.

In this section, we redesign the 4-round small PRESENT-[4] by swapping or replacing the internal components, e.g., S-box and bit permutation, and execute Experiment 1 and Experiment 2 against the new designs of the 4-round small PRESENT-[4] to reveal the relationship between the designs of block ciphers and average success probability of the proposed attacks.

Table 6: Average success probabilities when swapping or replacing components on the 4-round small PRESENT-[4]. We use 2^{15} training data and 2^{15} testing data. CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

	Category of Attack	# nodes of hidden layer	# layers of hidden layer	Initial learning rate	Optimizer	Succ. prob.
Original small PRESENT-[4]	CP	300	4	0.01	Adam	$2^{-5.63}$
	PR	300	1	0.01	Adam	$2^{-14.50}$
Replacing the components (Enc: sLayer-inv $\rightarrow$ pLayer) (Dec: pLayer $\rightarrow$ sLayer)	CP	200	4	0.01	Adam	$2^{-3.75}$
	PR	500	1	0.001	Adam	$2^{-12.13}$
Swapping the components (Enc: pLayer $\rightarrow$ sLayer) (Dec: sLayer-inv $\rightarrow$ pLayer)	CP	500	1	0.001	Adam	$2^{-12.21}$
	PR	400	7	0.001	Adam	$2^{-13.74}$

4.1 Experimental Procedure

We discuss two types of experiments to investigate the average success probabilities of ciphertext prediction and plaintext recovery by the proposed attacks under the conditions that (1) the substitution layer (sLayer) and its inverse function (sLayer-inv) are replaced, and (2) the order of the sLayer and permutation layer (pLayer) is swapped in the encryption and decryption algorithms. The target toy block ciphers are the 4-round small PRESENT-[4] and the 2-round small AES-[4], and small TWINE-[4] is excluded from the target of these experiments. This is because the Feistel block ciphers generally use the same components for both encryption and decryption algorithms. The order of the sLayer and pLayer is the same in both the encryption and decryption algorithms, and sLayer-inv is not used in neither the encryption nor decryption algorithms. Rather than the experiments described in this section, we should compare the maximum number of rounds that the proposed attacks can be successful against small TWINE-[4] (a type-II generalized Feistel cipher) to that on the other types of the Feistel block ciphers, such as classical, unbalanced, alternating, type-I and type-III generalized Feistel ciphers. This will be our future study.

4.2 Experimental Results

Table 6 shows experimental results for new designs of the 4-round small PRESENT-[4]. The average success probability of ciphertext prediction is greater than that of the original small PRESENT-[4] when the sLayer is replaced with the sLayer-inv, as shown in the table. However, when the order of the sLayer and pLayer is swapped, the average success probability of ciphertext prediction is less than that of the original small PRESENT-[4]. We believe that swapping the component order affects the average success probabilities of the proposed attacks because the difference in these average success probabilities is relatively large. Given that swapping or replacing components does not affect linear/differential probabilities, we expect that our results can be an important stepping stone for designing deep learning-resistant symmetric-key ciphers.

Nevertheless, the average success probability of plaintext recovery for both cases is greater than that of the original small PRESENT-[4]; this result tends to differ from ciphertext prediction. Because the probabilities are nearly 2^{-15}, the results require more detailed analyses to increase reliability, which we leave as a future work.

In the experimental results of the 2-round small AES-[4], all average success probabilities for ciphertext prediction/plaintext recovery by the proposed attacks are less than 2^{-15}. Therefore, these results do not show whether swapping or replacing the components has any effect on the average success probabilities of the proposed attacks in the 2-round small AES-[4].

5 Conclusion

In this study, we presented deep learning-based output prediction attacks on three block ciphers with a block size of 64 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 such as linear/differential attacks:

- For PRESENT, the maximum number of rounds that the proposed attack can be successful is at least equal to that of classical linear/differential attacks.
- For AES-like and TWINE-like ciphers, we conjecture that the maximum number of rounds that the proposed attacks can be successful also becomes equal to that of classical linear/differential attacks when the amount of training data is increased more.

In addition, we redesigned the 4-round small PRESENT-[4] by swapping or replacing the internal components, and we used the whitebox analysis technique to examine the relationship between the new target cipher designs and the success probability of the proposed attacks. Consequently, we clarified that swapping or replacing the internal components did not affect success probabilities of the classical linear/differential attacks, whereas it affects the average success probabilities of the proposed deep learning-based attacks; thus, we have obtained a deep learning specific characteristic. The obtained results are expected to be a foundation for designing deep learning-resistant symmetric-key ciphers.

Acknowledgments

This work was supported in part by the JSPS KAKENHI Grant Number 19K11971.

References

[1] Khaled M. Alallayah, Alaa H. Alhamami, Waiel AbdElwahed, and Mohamed Amin. Applying Neural Networks for Simplified Data Encryption Standard (SDES) Cipher System Cryptanalysis. *Int. Arab J. Inf. Technol.*, 9(2):163–169, 2012.

[2] Mohammed M. Alani. Neuro-Cryptanalysis of DES and Triple-DES. In *ICONIP*, pages 637–646, 2012.

[3] Riyad Alshammari and A. Nur Zincir-Heywood. Machine learning based encrypted traffic classification: Identifying SSH and Skype. In *IEEE CISDA*, pages 1–8, 2009.

[4] Seunggeun Baek and Kwangjo Kim. Recent Advances of Neural Attacks against Block Ciphers. *SCIS*, 2020. available at `https://caislab.kaist.ac.kr/publication/paper_files/2020/scis2020_SG.pdf`.

[5] Abbas Ghaemi Bafghi, Reza Safabakhsh, and Babak Sadeghiyan. Finding the differential characteristics of block ciphers with neural networks. *Information Sciences*, 178(15):3118 – 3132, 2008. Nature Inspired Problem-Solving.

[6] Anubhab Baksi, Jakub Breier, Yi Chen, and Xiaoyang Dong. Machine learning assisted differential distinguishers for lightweight ciphers. In *DATE*, pages 176–181, 2021.

[7] Zhenzhen Bao, Jian Guo, Meicheng Liu, Li Ma, and Yi Tu. Conditional Differential-Neural Cryptanalysis. *IACR Cryptol. ePrint Arch.*, 2021:719, 2021.

[8] Adrien Benamira, David Gérault, Thomas Peyrin, and Quan Quan Tan. A Deeper Look at Machine Learning-Based Cryptanalysis. In *EUROCRYPT*, pages 805–835, 2021.

[9] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *CHES*, pages 450–466, 2007.

[10] Yi Chen and Hongbo Yu. Neural Aided Statistical Attack for Cryptanalysis. *IACR Cryptol. ePrint Arch.*, 2020:1620, 2020.

[11] Yi Chen and Hongbo Yu. A New Neural Distinguisher Model Considering Derived Features from Multiple Ciphertext Pairs. *IACR Cryptol. ePrint Arch.*, 2021:310, 2021.

[12] Yi Chen and Hongbo Yu. Bridging Machine Learning and Cryptanalysis via EDLCT. *IACR Cryptol. ePrint Arch.*, 2021:705, 2021.

[13] Yi Chen and Hongbo Yu. Improved Neural Aided Statistical Attack for Cryptanalysis. *IACR Cryptol. ePrint Arch.*, 2021:311, 2021.

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

[15] Riccardo Focardi and Flaminia L. Luccio. Neural Cryptanalysis of Classical Ciphers. In *ICTCS*, pages 104–115, 2018.

[16] SK Pal Girish Mishra, SVSSNVG Krishna Murthy. Neural Network Based Analysis of Lightweight Block Cipher PRESENT. In Harmony Search and Nature Inspired Optimization Algorithms, 2019.

[17] Aron Gohr. Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning. In *CRYPTO*, pages 150–179, 2019.

[18] Aidan N. Gomez, Sicong Huang, Ivan Zhang, Bryan M. Li, Muhammad Osama, and Lukasz Kaiser. Unsupervised Cipher Cracking Using Discrete GANs. *CoRR*, abs/1801.04883, 2018.

[19] Sam Greydanus. Learning the Enigma with Recurrent Neural Networks. *CoRR*, abs/1708.07576, 2017.

[20] Sepp Hochreiter and Jurgen Schmidhuber. LONG SHORT-TERM MEMORY. In *Neural Computation Volume 9 Issue 8*, pages 1735–1780, 1997.

[21] Botao Hou, Yongqiang Li, Haoyue Zhao, and Bin Wu. Linear Attack on Round-Reduced DES Using Deep Learning. In *ESORICS*, pages 131–145, 2020.

[22] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Cryptanalysis of Round-Reduced SIMON32 Based on Deep Learning. *IACR Cryptol. ePrint Arch.*, 2021:362, 2021.

[23] Zezhou Hou, Jiongjiong Ren, and Shaozhen Chen. Improve Neural Distinguisher for Cryptanalysis. *IACR Cryptol. ePrint Arch.*, 2021:1017, 2021.

[24] Xinyi Hu and Yaqun Zhao. Research on Plaintext Restoration of AES Based on Neural Network. *Security and Communication Networks*, 2018:6868506:1–6868506:9, 2018.

[25] Mohamed Fadl Idris, Je Sen Teh, Jasy Liew Suet Yan, and Wei-Zhu Yeoh. A Deep Learning Approach for Active S-Box Prediction of Lightweight Generalized Feistel Block Ciphers. *IEEE Access*, 9:104205–104216, 2021.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015.

[27] Gregor Leander. Small Scale Variants Of The Block Cipher PRESENT. *IACR Cryptol. ePrint Arch.*, 2010:143, 2010.

[28] Ting Lee, Je Sen Teh, Jasy Liew, Suet Yan, Norziana Jamil, and Wei-Zhu Yeoh. A Machine Learning Approach to Predicting Block Cipher Security. In *CRYPTOLOGY*, 2020.

[29] Ting Rong Lee, Je Sen Teh, Norziana Jamil, Jasy Liew Suet Yan, and Jiageng Chen. Lightweight Block Cipher Security Evaluation Based on Machine Learning Classifiers and Active S-Boxes. *IEEE Access*, 9:134052–134064, 2021.

[30] Yu Liu, Jianshu Chen, and Li Deng. Unsupervised Sequence Classification using Sequential Output Statistics. In *NIPS*, pages 3550–3559, 2017.

[31] Itsik Mantin and Adi Shamir. A Practical Attack on Broadcast RC4. In *Fast Software Encryption*, pages 152–164, 2001.

[32] Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In *CHES*, pages 342–357, 2011.

[33] Jaewoo So. Deep Learning-Based Cryptanalysis of Lightweight Block Ciphers. *Security and Communication Networks*, 2020:3701067, 2020.

[34] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*, pages 3104–3112, 2014.

[35] C. Tan and Q. Ji. An approach to identifying cryptographic algorithm from ciphertext. In *ICCSN*, pages 19–23, 2016.

[36] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[37] Gao Wang and Gaoli Wang. Improved Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Analysis. In *ICICS 2021*, pages 21–38, 2021.

[38] Ya Xiao, Qingying Hao, and Danfeng Daphne Yao. Neural Cryptanalysis: Metrics, Methodology, and Applications in CPS Ciphers. In *IEEE DSC*, pages 1–8, 2019.

[39] Tarun Yadav and Manoj Kumar. Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Cryptanalysis. In *LATINCRYPT 2021*, pages 191–212, 2021.

A Our Target Ciphers

In this section, we introduce two SPN block ciphers (PRESENT [9] and AES-like cipher), one Feistel block cipher (TWINE-like cipher), and their toy ciphers (small PRESENT-[n] [27], small AES-[n], and small TWINE-[n]).

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] [27] has been proposed. We show the round function of small PRESENT-[n] in Fig. A.1. Since the block size is $4n$, 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 the round of full diffusion. The S-box has 4-bit input and output. We provide the correspondence table in Table A.1 that maps $\mathbb{F}_2^4 \to \mathbb{F}_2^4$. The pLayer is described as 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 $n = 16$. $P(i)$ 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.

AES-like and small AES-[n]: We design AES-like cipher with a 64-bit block size, called AES-like for short. To analyze AES-like, we design its toy model called small AES-[n]. The round function of small AES is shown in Fig. A.1. As with the case of PRESENT, small AES-[16] is equivalent to AES-like since the block size is $4n$. The S-box and key scheduling are the same as those of PRESENT. The maximum distance separable (MDS) matrix (over $GF(2^4)$ defined by the irreducible polynomial $x^4 + x + 1$) is the same as that of Piccolo [32], which is expressed as follows.

$$M = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

When a 16-bit input $X_{(16)}$ is given, the output is computed as ${}^t(y_{0(4)}, y_{1(4)}, y_{2(4)}, y_{3(4)}) \leftarrow M \cdot {}^t(x_{0(4)}, x_{1(4)}, x_{2(4)}, x_{3(4)})$.

TWINE-like and small TWINE-[n]: We design TWINE-like cipher with a 64-bit block size, called TWINE-like for short. To analyze TWINE-like, we design its toy model called small TWINE-[n]. For our design, we adopt the type-II generalized Feistel structure with n branches and similar F function as TWINE, which comprises round key operation and 4-bit S-box, as shown in Fig. A.2.

As with the case of PRESENT, small TWINE-[16] is equivalent to TWINE-like since the block size is $4n$. The S-box and key scheduling are the same as those of PRESENT. The pLayer is described as round permutation RP, which is defined as follows:

$$RP : (y_0, y_1, \ldots, y_{n-2}, y_{n-1}) \leftarrow (x_1, x_2, \ldots, x_{n-1}, x_0).$$

Table A.1: S-box (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

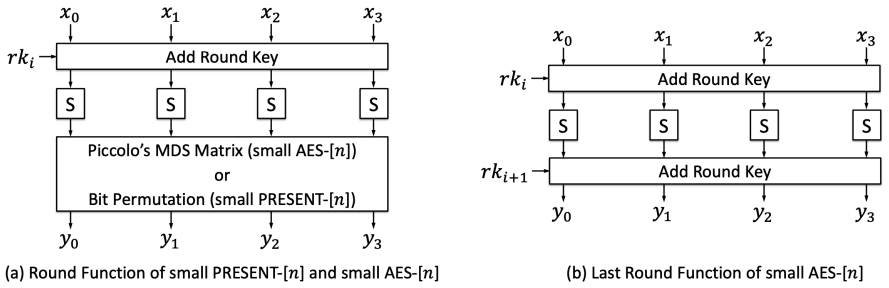

Figure A.1: (a) Round Functions of small PRESENT-[n] and small AES-[n], (b) Last Round Function of small AES-[n].

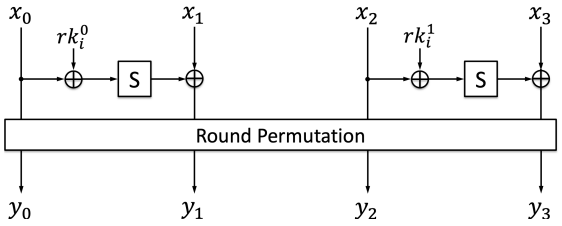

Figure A.2: Round Function of small TWINE-[n]

Two sub-round keys, rk_i^s for $s \in \{0, 1, \ldots, \frac{n}{2} - 1\}$, are used in each round, which are generated from the round key rk_i as follows:

$$rk_i^s = (rk_i \gg (4n - (4s + 4))) \ \& \ \mathsf{0xF},$$

where $\gg$ and & are bitwise right shift operation and bitwise AND operation, respectively.

B Related Works

Regarding the whitebox analysis, Danziger et al. presented deep learning-based attacks that predict key bits of 2-round DES from a plaintext/ciphertext set, and analyze the relationship between these attacks and the differential probability [14]. They compared variants employing several types of S-boxes with different properties for differential attacks, and they 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 ideal function. It is unclear how much the property of internal components affects the security of the whole construction. In addition to improve Gohr's deep learning-based attack [17], Benamira et al. [8] and Chen et al. [12] improved the success probability of traditional distinguishers using characteristics that are expected to be reacted by Gohr's attack. Their work confirms whether characteristics explored by Gohr can be employed in the traditional distinguishing attacks and they did not identify any deep learning specific characteristic. However, we calculated the ability of traditional distinguisher and our deep

Table B.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 [38]	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 [33]	Simplified DES (8 bits)	Feistel	No	KR/LD	8 (8)	No
So20 [33]	Speck32/64 (32 bits)	Feistel	No	KR/LD	22 (22)	No
So20 [33]	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
BGMLT21 [7]	Speck32/64 (32 bits)	Feistel	No	KR/DD	13 (22)	Yes
BGMLT21 [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 [39]	Speck32/64 (32 bits)	Feistel	No	DD	9 (22)	Yes
YK21 [39]	Simon32/64 (32 bits)	Feistel	No	DD	12 (32)	Yes
YK21 [39]	GIFT 64 (64 bits)	SPN	No	DD	8 (28)	Yes
WW21 [37]	Speck32/64 (32 bits)	Feistel	No	DD	12 (22)	Yes
WW21 [37]	Speck48/72 (48 bits)	Feistel	No	DD	15 (22)	Yes
WW21 [37]	Speck64/96 (64 bits)	Feistel	No	DD	18 (26)	Yes
This paper	**PRESENT (64 bits)**	SPN	**Yes**	**OP**	**4** (31)	**Yes**
This paper	**AES-like (64 bits)**	SPN	**Yes**	**OP**	**1** (N/A[2])	**Yes**
This paper	**TWINE-like (64 bits)**	Feistel	**Yes**	**OP**	**3** (N/A[2])	**Yes**

[1] Gohr described in his paper [17] 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* are different from our defined blackbox setting. For this reason, we consider his proposed model as a non-blackbox setting.

[2] Because 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, we described the number of full rounds of these modified versions as 'N/A'.

learning-based attack and compared them to investigate a relationship between them. Then, we identified a deep learning specific characteristic of small-PRESENT. To summarize, to the best of our knowledge, our results are the first ones that perform the whitebox analysis.

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 1741 ($\simeq 2^{10.76}$) 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, we exclude these results in Table B.1. Mishra et al. reported that they mounted output prediction attacks on full-round PRESENT; however, it did not work well [16]. In addition, certain results have yielded classical ciphers such as Caesar cipher, Vigenere, and Enigma ciphers [15, 18, 19, 30].

Other machine learning-based analyses have also been reported, e.g., [28,29]. Tan et al. demonstrated that deep learning can be used to distinguish ciphertexts encrypted by AES, Blowfish, DES, 3-DES, and RC5, respectively [35], for detecting the encryption algorithm that the malware utilizes. Alshammari et al. attempted to classify encrypted Skype and SSH traffic [3].

C Experimental Results Using the CNN

To confirm that the use of the LSTM induces better experimental results than that of the CNN, we conducted experiments using the CNN in the same procedure described in Sect. 3.1.1. In our experiments, we optimize activation functions in addition to the hyperparameters shown in Table 1. The following is the search range for activation functions: Tanh, Sigmoid, and ReLU. For developing CNN models by Keras, e.g., `model.add(Conv1D(...))`, we specify only `filters`, `kernel_size`, `activation`, and `input_shape` as its arguments[5].

Table C.1 shows experimental results using the CNN. Consequently, we clarify the following facts by comparing the experimental results using the LSTM and CNN based on Tables 3 and C.1:

- For small PRESENT-[4], the maximum number of rounds that the proposed attacks using the LSTM and CNN can be successful is 4 and 3, respectively.
- For small AES-[4], the maximum number of rounds that the proposed attacks using the LSTM and CNN can be successful is 1 for each case. In addition, the average success probabilities of ciphertext prediction (plaintext recovery) by the proposed attacks against the 1-round small AES-[4] using the LSTM and CNN are 1 (1) and $2^{-11.88}$ ($2^{-11.83}$), respectively.
- For small TWINE-[4], the maximum number of rounds that the proposed attacks using the LSTM and CNN can be successful is 3 and 1, respectively.

To summarize the foregoing facts, we conclude that the use of the LSTM induces better experimental results of all the target block ciphers compared to the use of the CNN.

D Maximum Differential Probabilities of Small PRESENT-[4], Small AES-[4], and Small TWINE-[4]

Table D.1 shows the maximum differential probabilities of small PRESENT-[4], small AES-[4], and small TWINE-[4].

Table C.1: Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against three toy block ciphers with a block size of 16 bits. We employ the CNN and use 2^{15} training data as well as 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	Activation	Succ. prob.
small PRESENT-[4]	1	CP	100	5	0.001	Adam	Tanh	1
		PR	300	3	0.0001	RMSprop	Tanh	1
	2	CP	200	1	0.01	SGD	ReLU	$2^{-11.72}$
		PR	400	3	0.001	Adam	Sigmoid	$2^{-11.69}$
	3	CP	200	2	0.01	RMSprop	ReLU	$2^{-14.58}$
		PR	300	2	0.0001	RMSprop	Sigmoid	$2^{-14.57}$
	4	CP	300	7	0.0001	Adam	Tanh	$2^{-15.02}$
		PR	200	1	0.01	SGD	ReLU	$2^{-15.20}$
small AES-[4]	1	CP	100	4	0.0001	Adam	ReLU	$2^{-11.88}$
		PR	400	5	0.0001	Adam	Tanh	$2^{-11.83}$
	2	CP	100	4	0.01	SGD	ReLU	$2^{-15.76}$
		PR	100	7	0.001	RMSprop	Sigmoid	$2^{-15.00}$
small TWINE-[4]	1	CP	300	2	0.001	RMSprop	Sigmoid	$2^{-8.01}$
		PR	500	5	0.01	RMSprop	ReLU	$2^{-8.03}$
	2	CP	100	4	0.0001	SGD	Tanh	$2^{-15.86}$
		PR	300	2	0.0001	RMSprop	Tanh	$2^{-15.62}$

E More Detailed Results in Sect. 3.1.3

Table E.1 shows the minimum amount of training data required for successful ciphertext prediction/plaintext recovery against three toy block ciphers. In addition, Table E.2 details the experimental results shown in Table E.1. If the predicted probability is greater than 2^{-15}, we consider the proposed attacks to be successful[6].

[5] `https://keras.io/ja/layers/convolutional/`

[6] This assumption is strictly incorrect, but we use it for simple discussion.

Table D.1: Maximum differential probabilities of small PRESENT-[4], small AES-[4], and small TWINE-[4]

Round	Maximum differential probability		
	small PRESENT-[4]	small AES-[4]	small TWINE-[4]
1	2^{-2}	2^{-2}	2^{0}
2	2^{-4}	2^{-9}	2^{-2}
3	2^{-7}	2^{-11}	2^{-4}
4	2^{-8}	2^{-11}	2^{-6}
5	2^{-14}	2^{-11}	2^{-7}
6	2^{-15}	2^{-11}	2^{-9}
7	2^{-15}	2^{-11}	2^{-9}
8	2^{-15}	2^{-12}	2^{-11}
9	–	–	2^{-11}
10	–	–	2^{-11}

Table E.1: Minimum amount of training data required for successful ciphertext prediction/plaintext recovery using the proposed attacks against three toy block ciphers with a block size of 16 bits. We use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	# training data	Succ. prob.
small PRESENT-[4]	1	CP	2^{3}	$2^{-14.76}$
		PR	2^{3}	$2^{-13.40}$
	2	CP	2^{4}	$2^{-14.57}$
		PR	2^{4}	$2^{-14.56}$
	3	CP	2^{11}	$2^{-12.11}$
		PR	2^{11}	$2^{-14.55}$
	4	CP	2^{15}	$2^{-5.63}$
		PR	2^{15}	$2^{-14.50}$
small AES-[4]	1	CP	2^{9}	$2^{-13.32}$
		PR	2^{8}	$2^{-14.82}$
small TWINE-[4]	1	CP	2^{4}	$2^{-13.58}$
		PR	2^{3}	$2^{-14.93}$
	2	CP	2^{11}	$2^{-11.73}$
		PR	2^{11}	$2^{-13.18}$
	3	CP	2^{14}	$2^{-13.54}$
		PR	2^{14}	$2^{-13.02}$

Table E.2: Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against three toy block ciphers with a block size of 16 bits. We vary the amount of training data in the range of from 2^2 to 2^{14} and use all remaining plaintexts or ciphertexts as testing data. Moreover, we use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	Success probability for each amount of training data												
			2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}
small PRESENT-[4]	1	CP	$2^{-16.57}$	$2^{-14.76}$	$2^{-14.46}$	$2^{-13.83}$	$2^{-13.78}$	$2^{-13.99}$	$2^{-12.90}$	$2^{-9.76}$	$2^{-6.08}$	$2^{-0.76}$	1	1	1
		PR	$2^{-15.44}$	$2^{-13.40}$	$2^{-12.28}$	$2^{-11.79}$	$2^{-11.65}$	$2^{-11.62}$	$2^{-10.56}$	$2^{-8.70}$	$2^{-5.65}$	$2^{-1.53}$	$2^{-0.01}$	1	1
	2	CP	–	$2^{-15.63}$	$2^{-14.57}$	$2^{-14.81}$	$2^{-14.60}$	$2^{-14.94}$	$2^{-14.83}$	$2^{-12.80}$	$2^{-7.96}$	$2^{-2.49}$	$2^{-0.01}$	1	1
		PR	–	$2^{-15.08}$	$2^{-14.56}$	$2^{-14.46}$	$2^{-14.46}$	$2^{-14.51}$	$2^{-14.32}$	$2^{-14.38}$	$2^{-13.81}$	$2^{-10.44}$	$2^{-2.46}$	1	1
	3	CP	–	–	–	–	–	–	–	–	$2^{-15.02}$	$2^{-12.11}$	$2^{-10.34}$	$2^{-7.80}$	$2^{-0.01}$
		PR	–	–	–	–	–	–	–	–	$2^{-15.54}$	$2^{-14.55}$	$2^{-14.43}$	$2^{-13.89}$	$2^{-12.86}$
	4	CP	–	–	–	–	–	–	–	–	–	–	–	$2^{-15.16}$	$2^{-15.01}$
		PR	–	–	–	–	–	–	–	–	–	–	–	$2^{-16.09}$	$2^{-15.42}$
small AES-[4]	1	CP	–	–	–	–	–	$2^{-15.04}$	$2^{-15.24}$	$2^{-13.32}$	$2^{-9.22}$	$2^{-4.42}$	$2^{-0.29}$	$2^{-0.01}$	1
		PR	–	–	–	–	–	$2^{-15.17}$	$2^{-14.82}$	$2^{-13.67}$	$2^{-10.47}$	$2^{-7.58}$	$2^{-4.35}$	$2^{-0.05}$	$2^{-0.01}$
small TWINE-[4]	1	CP	$2^{-15.54}$	$2^{-15.09}$	$2^{-13.58}$	$2^{-13.48}$	$2^{-13.19}$	$2^{-13.24}$	$2^{-12.99}$	$2^{-11.05}$	$2^{-8.12}$	$2^{-4.97}$	$2^{-1.64}$	$2^{-0.01}$	1
		PR	$2^{-16.99}$	$2^{-14.93}$	$2^{-13.28}$	$2^{-12.51}$	$2^{-11.87}$	$2^{-11.77}$	$2^{-12.06}$	$2^{-10.19}$	$2^{-7.28}$	$2^{-3.75}$	$2^{-1.93}$	1	$2^{-0.02}$
	2	CP	–	–	–	–	–	–	–	–	$2^{-15.86}$	$2^{-11.73}$	$2^{-7.62}$	$2^{-4.55}$	$2^{-0.58}$
		PR	–	–	–	–	–	–	–	–	$2^{-15.63}$	$2^{-13.18}$	$2^{-9.72}$	$2^{-5.03}$	$2^{-1.06}$
	3	CP	–	–	–	–	–	–	–	–	–	–	–	$2^{-15.47}$	$2^{-13.54}$
		PR	–	–	–	–	–	–	–	–	–	–	–	$2^{-15.40}$	$2^{-13.02}$

F More Detailed Results in Sect. 3.2

Tables F.1 and F.2 show the minimum amount of training data required for successful ciphertext prediction/plaintext recovery against three block ciphers with block sizes of 32 and 64 bits, respectively. In addition, Tables F.3 and F.4 detail the experimental results shown in Tables F.1 and F.2, respectively.

We vary the amount of training data in the range of from 2^8 to 2^{17} or from 2^{10} to 2^{19} and use 2^{16} of the remaining plaintexts or ciphertexts as testing data against three toy block ciphers with block sizes of 32 or 64 bits, respectively; thus, if the predicted probability is greater than the threshold derived by equation $(2^{4n} - 2^x)^{-1}$ shown in Sect. 2.3, we consider the proposed attacks to be successful for both cases. In this case, those thresholds are $(2^{32} - 2^8)^{-1}$ to $(2^{32} - 2^{17})^{-1}$ or from $(2^{64} - 2^{10})^{-1}$ to $(2^{64} - 2^{19})^{-1}$.

Table F.1: Minimum amount of training data required for successful ciphertext prediction/plaintext recovery using the proposed attacks against three block ciphers with a block size of 32 bits. We use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	# training data	Succ. prob.
small PRESENT-[8]	1	CP	2^{10}	$2^{-19.47}$
		PR	2^{9}	$2^{-22.64}$
	2	CP	2^{11}	$2^{-21.64}$
		PR	2^{14}	$2^{-1.20}$
	3	CP	2^{15}	$2^{-6.34}$
		PR	2^{17}	$2^{-2.24}$
	4	CP	N/A	N/A
		PR	N/A	N/A
small AES-[8]	1	CP	2^{12}	$2^{-17.78}$
		PR	2^{11}	$2^{-22.64}$
small TWINE-[8]	1	CP	2^{11}	$2^{-20.32}$
		PR	2^{10}	$2^{-21.64}$
	2	CP	2^{14}	$2^{-20.32}$
		PR	2^{14}	$2^{-14.49}$
	3	CP	2^{16}	$2^{-22.64}$
		PR	2^{17}	$2^{-19.18}$

Table F.2: Minimum amount of training data required for successful ciphertext prediction/plaintext recovery using the proposed attacks against three block ciphers with a block size of 64 bits. We use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	# training data	Succ. prob.
small PRESENT-[16] (PRESENT)	1	CP	2^{13}	$2^{-18.00}$
		PR	2^{14}	$2^{-17.68}$
	2	CP	2^{14}	$2^{-6.64}$
		PR	2^{16}	$2^{-2.27}$
	3	CP	2^{17}	$2^{-1.30}$
		PR	2^{19}	$2^{-3.14}$
	4	CP	N/A	N/A
		PR	N/A	N/A
small AES-[16] (AES-like)	1	CP	2^{15}	$2^{-0.24}$
		PR	2^{15}	$2^{-0.06}$
small TWINE-[16] (TWINE-like)	1	CP	2^{15}	$2^{-0.30}$
		PR	2^{14}	$2^{-15.91}$
	2	CP	2^{19}	$2^{-12.50}$
		PR	2^{19}	$2^{-7.05}$
	3	CP	N/A	N/A
		PR	N/A	N/A

Table F.3: Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against three block ciphers with a block size of 32 bits. We vary the amount of training data in the range of from 2^8 to 2^{17} and use 2^{16} of the remaining plaintexts or ciphertexts as testing data. Moreover, we use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	Success probability for each amount of training data									
			2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}
small PRESENT-[8]	1	CP	0	0	$2^{-19.47}$	$2^{-13.47}$	$2^{-0.51}$	1	1	1	1	1
		PR	0	$2^{-22.64}$	$2^{-20.64}$	$2^{-13.76}$	$2^{-7.61}$	$2^{-1.80}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$
	2	CP	0	0	0	$2^{-21.64}$	$2^{-17.35}$	$2^{-6.22}$	$2^{-0.57}$	$2^{-0.54}$	$2^{-0.05}$	$2^{-0.02}$
		PR	0	0	0	0	0	0	$2^{-1.20}$	$2^{-0.06}$	$2^{-0.13}$	$2^{-0.05}$
	3	CP	0	0	0	0	0	0	0	$2^{-6.34}$	$2^{-3.32}$	$2^{-2.35}$
		PR	0	0	0	0	0	0	0	0	0	$2^{-2.24}$
	4	CP	0	0	0	0	0	0	0	0	0	0
		PR	0	0	0	0	0	0	0	0	0	0
small AES-[8]	1	CP	0	0	0	0	$2^{-17.78}$	$2^{-8.25}$	$2^{-0.12}$	$2^{-0.01}$	$2^{-0.01}$	1
		PR	0	0	0	$2^{-22.64}$	$2^{-20.64}$	$2^{-13.43}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$
small TWINE-[8]	1	CP	0	0	0	$2^{-20.32}$	$2^{-17.47}$	$2^{-4.41}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$
		PR	0	0	$2^{-21.64}$	$2^{-16.45}$	$2^{-12.62}$	$2^{-3.86}$	$2^{-1.69}$	$2^{-0.76}$	$2^{-0.76}$	$2^{-0.27}$
	2	CP	0	0	0	0	0	0	$2^{-20.32}$	$2^{-4.97}$	$2^{-4.84}$	$2^{-2.84}$
		PR	0	0	0	0	0	0	$2^{-14.49}$	$2^{-16.62}$	$2^{-8.78}$	$2^{-12.68}$
	3	CP	0	0	0	0	0	0	0	0	$2^{-22.64}$	$2^{-15.71}$
		PR	0	0	0	0	0	0	0	0	0	$2^{-19.18}$

Table F.4: Average success probabilities of ciphertext prediction/plaintext recovery using the proposed attacks against three block ciphers with a block size of 64 bits. We vary the amount of training data in the range of from 2^{10} to 2^{19} and use 2^{16} of the remaining plaintexts or ciphertexts as testing data. Moreover, we use the optimized hyperparameters obtained in Experiment 1 (see Table 3). CP:=Ciphertext Prediction and PR:=Plaintext Recovery.

Cipher	Round	Attack	Success probability for each amount of training data									
			2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}
small PRESENT-[16] (PRESENT)	1	CP	0	0	0	$2^{-18.00}$	$2^{-0.01}$	1	1	–	–	–
		PR	0	0	0	0	$2^{-17.68}$	$2^{-8.04}$	$2^{-0.37}$	$2^{-0.01}$	$2^{-0.01}$	–
	2	CP	0	0	0	0	$2^{-6.64}$	$2^{-2.64}$	$2^{-2.31}$	$2^{-1.68}$	$2^{-0.60}$	–
		PR	0	0	0	0	0	0	$2^{-2.27}$	$2^{-0.39}$	$2^{-0.08}$	–
	3	CP	0	0	0	0	0	0	0	$2^{-1.30}$	$2^{-1.68}$	$2^{-0.74}$
		PR	0	0	0	0	0	0	0	0	0	$2^{-3.14}$
	4	CP	0	0	0	0	0	0	0	0	0	0
		PR	0	0	0	0	0	0	0	0	0	0
small AES-[16] (AES-like)	1	CP	0	0	0	0	0	$2^{-0.24}$	$2^{-0.06}$	$2^{-0.01}$	$2^{-0.01}$	–
		PR	0	0	0	0	0	$2^{-0.06}$	$2^{-0.01}$	$2^{-0.01}$	$2^{-0.01}$	–
small TWINE-[16] (TWINE-like)	1	CP	0	0	0	0	0	$2^{-0.30}$	$2^{-0.04}$	$2^{-0.01}$	$2^{-0.01}$	–
		PR	0	0	0	0	$2^{-15.91}$	$2^{-5.72}$	$2^{-5.16}$	$2^{-3.99}$	$2^{-2.45}$	$2^{-1.16}$
	2	CP	0	0	0	0	0	0	0	0	0	$2^{-12.50}$
		PR	0	0	0	0	0	0	0	0	0	$2^{-7.05}$
	3	CP	0	0	0	0	0	0	0	0	0	0
		PR	0	0	0	0	0	0	0	0	0	0