SCIS 2020 2020 Symposium on
Cryptography and Information Security
Kochi, Japan, Jan. 28 – 31, 2020
The Institute of Electronics,
Information and Communication Engineers

Recent Advances of Neural Attacks against Block Ciphers

Seunggeun Baek * Kwangjo Kim †

Abstract: Neural cryptanalysis is the utilization of deep learning to attack cryptographic primitives. As computing power increases, deploying neural cryptanalysis becomes a more feasible option to attack more complex ciphers. After reviewing all recent neural cryptanalysis publications on the security of block ciphers including the detailed outcome of the attacks, we find that the types of neural cryptanalysis on block ciphers can be classified into key recovery, cipher emulation, and identification attacks. We evaluate whether the publications have used correct methodologies to analyze the attack results or not, discuss limitations of current neural cryptanalysis results, and suggest future direction of development of this field.

Keywords: Deep Learning, Neural Cryptanalysis, Block Cipher.

1 Introduction

Deep learning (DL) models are capable of efficiently approximating any unknown functions. Researchers have extended applications of DL to every field from customer behavior prediction to video generation. Now the frontier has reached to the most complex types of functions, namely cryptographic primitives. Attacking cryptosystems using deep learning is called neural cryptanalysis.

It has been a long time since the possibility of utilizing neural networks for design and analysis of cryptosystems has first proposed [29]. Such systems and analysis methods had to remain on theoretical interest before the available computing power reaches a certain level to perform actual experiments. Now the resource becomes plentiful, and theoretical advances lead to more effective network structure and optimization methods.

However, it seems that neural cryptanalysis is still in the early stage due to the hardness of the tasks targeting complex cryptographic primitives designed to have strict requirements such as indistinguishability. Also, the knowledge of the previous attacks has scattered and not been systemized effectively. Therefore, we survey the previous approaches and applications of neural cryptanalysis on block ciphers. We also raise disputes on the results of some previous publications, about whether they have proper models and analysis methods.

* Graduate School of Information Security, KAIST. 291, Daehak-ro, Yuseong-gu, Daejeon, South Korea 34141. baek449@kaist.ac.kr

† School of Computing, KAIST. 291, Daehak-ro, Yuseong-gu, Daejeon, South Korea 34141. kkj@kaist.ac.kr

2 Background

2.1 Block Cipher

Block ciphers are symmetric cryptosystems that encrypt plaintexts or decrypts ciphertexts by the unit of blocks. Common designs of block ciphers include the Feistel network and the substitution-permutation network (SPN).

Practical ciphers are general-purpose block ciphers having decent security level. Data Encryption Standard (DES) is one of the first practical ciphers. Currently, Advanced Encryption Standard (AES) is in the most widespread use. Other than AES, plenty of practical block ciphers exist such as MARS, RC6, Serpent, and Twofish.

Toy ciphers are block ciphers that having small key and block size and simple structures. Simplified Data Encryption Standard (SDES) [30] is one of the well-known toy ciphers. Toy ciphers are used for demonstrating cryptanalysis techniques as proofs of concepts. In the neural cryptanalysis area, a lot of papers we surveyed have targeted toy ciphers, especially SDES.

Lightweight ciphers are block ciphers designed for small devices which have strict performance and memory requirements. Lightweight ciphers utilize simple operations such as arithmetic addition, shifts (including rotation), and bitwise binary operations. Lightweight ciphers are denoted as LW in the tables of this paper.

2.2 Cryptanalysis of Block Cipher

The two well-known cryptanalysis methods are differential cryptanalysis (DC) and linear cryptanalysis (LC). The two attacks are notable as they broke the security of DES. Other attempts have been proposed, such as a method using multivariate quadratic equations [11].

2.3 Deep Learning

Deep Learning is a group of techniques to solve learning problems using a neural network containing various hidden layers. Dense layers and convolutional layers are frequently used. Other types of connections can be used; cascaded layers have connections from neurons of all the layers from the input layer to the previous layer. By training, the parameters (weights and the biases of the neurons) are optimized to minimize the given loss function.

Neural networks are known to be capable of approximating any given function if enough training data and latent space were given. However, it is also known that in general boolean functions with constant depth threshold circuits, the required resources would be beyond polynomial [6]. Shown by a known provable security of a public-key cryptosystem, this result making cryptanalysis using neural network a challenging topic.

2.4 Neural Cryptanalysis

• **Definition** Neural cryptanalysis is defined as any attempt to break a cryptosystem using deep learning.

• **Attacks on Block Ciphers** For block ciphers, several attack scenarios of neural cryptanalysis will be discussed in Sections 4, 5, and 6. Most attacks took the black-box view of block ciphers, assuming the adversary knows the entire specifications of the block cipher algorithms, but does not know the secret key and corresponding round keys.

• **Relationship with Neural Cryptography** There exists the field of neural cryptography which utilizes deep learning models as encryption and decryption oracles. Some schemes of neural cryptography have been proposed for encrypting images. However, emulating these oracles of the schemes are equivalent to a model extraction attack. From the perspective of deep learning security, model extraction attacks already have been widely researched and deployed in order to steal paid models.

2.5 Neuro-aided Cryptanalysis

• **Definition** As a subset of neural cryptanalysis, neuro-aided cryptanalysis is defined as any attempt to break a cryptosystem using classical methods, while deep learning models are utilized to increase the effectiveness of the attacks.

• **Neuro-aided Side-channel Attack** The most developed area of neuro-aided cryptanalysis is neuro-aided side-channel attack [7, 16, 31]. The adversary obtains the physical properties such as level of power signal on the target hardware implementation of a cryptosystem. Deep learning is used for analyzing metrics such as timing and power amplitude information obtained from the hardware more effectively.

• **Possible Applications on Block Ciphers** Scenarios such as finding weaknesses of S-box design, or finding strong differential properties of a block cipher may fall in the category of neuro-aided cryptanalysis.

3 Scope of the Survey

3.1 Classification by Attacks

Even if the target cryptosystem is same, different attack scenarios exist based on different assumptions, which leads to different use cases of DL models on the attacks. For instance, regarding block ciphers, key recovery attacks aim to recover keys used in the cipher (Section 4.1), while cipher emulation attacks (Section 5.1) aim to simulate encryption or decryption oracles to recover plaintexts from ciphertexts without knowing the key. Identification attacks (Section 6.1) determine which encryption algorithm is used for the ciphertext.

The configurations of how deep learning models are applied to cryptanalysis widely vary among the attacks. For example, DL could be applied to end-to-end key recovery while plaintext and ciphertext pairs are given as training data, or DL could be incorporated into classical cryptanalysis techniques such as differential cryptanalysis. The details of each attack setting will be described in the corresponding section.

3.2 Comparison Metrics

For each attack, the previous publications are compared by following comparison metrics. A comparison table is provided for each attack, while the properties not specified in the publications are denoted as question marks.

Target Block Cipher The publications are classified by the target block ciphers used in the attack. The structures of the block ciphers were compared, also including whether the block ciphers are toy, lightweight (specified as LW), or practical ciphers. Number of rounds indicate the round count of the block cipher whose authors tried to break.

Neural Network Properties For the general comparison of neural networks, multiple aspects and parameters such as layer types, number of hidden layers, number of neurons per layer, and activation function are specified in the tables.

Training Methods Amount of training data, loss functions, and average number of epoch are also provided to describe the training environment and parameters.

3.3 Disclaimer

This survey does not cover neuro-aided side-channel attacks of block ciphers. This survey does not cover neural cryptanalysis of stream ciphers [15], random number generators [32], physically unclonable functions [28], public key cryptosystems [27], neural cryptosystems [24, 1], chaos-based cryptosystems [19], historical

Table 1: Comparison on key recovery attacks using deep learning

Publications		AW04 [4]	AAAA12 [2]	DH14 [12]	Goh19 [17]	Wab19 [33]
Attack Method		KR/F	KR	KR	KR/F	KR/F
Target Block Cipher	Name	HypCipher	SDES	SDES	Speck32/64	Hey02 [20]
	Internal Structure	Feistel/Toy	Feistel/Toy	Feistel/Toy	ARX/LW	SPN/Toy
	Block Size (bit)	16	8	8	32	16
Attacked Round (Full Round)		4 (4)	2 (2)	2 (2)	11 (22)	4 (4)
Neural Network Properties	Layer Type	Dense	Dense	Dense	CNN/Residual	Dense
	# of Hidden Layers	2	32	1	$\leq$10	2
	# of Neurons per Layer	16	32	?	?	256
	Activation Function	Sigmoid	?	?	ReLU	ReLU
Training Methods	Loss Function†	SSE	SSE	?	MSE	Varying
	Training Data (tuples)	$\leq$53	1,024	102,400	10^7	$\leq$8,000
	Avg. # of Epoch	?	7,869	?	200	200
Result		Success	Success	Success	Success	Success

† MSE: Mean Squared Error, SSE: Sum Squared Error [(SSE) = (# of output neurons) $\times$ (MSE)]

ciphers such as substitution ciphers and ENIGMA [18], and any cryptosystems other than block ciphers.

4 Key Recovery Attacks

4.1 Description

Key recovery attacks are attempts to guess the key of the cipher in non-negligible advantage. Even if an adversary obtains some partial information of the key, such as a round key, the information becomes an important clue toward full key recovery, the total break of the cipher.

• **Key Recovery Attack (KR)** Given a plaintext and ciphertext pair (p, c) satisfying $c = Enc(k, p)$, the attacker makes attempt to find k. Triples of a random plaintext, a random key, and the corresponding ciphertext (k_i, p_i, c_i) such that $c_i = Enc(k_i, p_i)$ are given as training data.

• **Key Recovery Attack on Fixed Key (KR/F)** The objective of the attack is same as normal key recovery attacks, but only pairs of random plaintexts and the corresponding ciphertexts (p_i, c_i) such that $c_i = Enc(k, p_i)$ are given as training data. A single deep learning model cannot be an end-to-end solution since the key is not given in the training data. Therefore, the attack is usually integrated with an analysis method that requires the knowledge of the algorithm. A common approach is to guess the key and examine the model's performance for each possible key.

4.2 Outcome

Albassal et al. [4] produced one of the pioneering works in the neural cryptanalysis field. To perform the key recovery attack on a fixed key for n-round cipher, an attacker may guess the final round key r and train the neural network for $(n-1)$-round cipher. If r is wrong, the result would be uniformly distributed for the test data, by the wrong key randomization hypothesis [5]. If r is correct, the error rate would be significantly lower than the wrong guesses. Wabbersen [33] had similar but slightly different approaches and network structures against another SPN-structured toy cipher on the Master's thesis.

Alallayah et al. [2] distinguished key recovery attacks from other emulation attacks, and performed both attacks on SDES. Danziger et al. [12] did a similar task using smaller layers and more training data. The authors used the neural cryptanalysis results to claim that a possible differential weakness on the S-box for SDES had been effectively patched.

Gohr [17] provided significant contribution to extend key recovery attack toward real-world lightweight cipher named Speck32/64. He provided a systematic approach by first training a real-or-random distinguisher on chosen plaintext assumption based on some known property from differential cryptanalysis and utilized the model for key recovery. The author further claimed that the wrong key randomization hypothesis is not ideally applicable for lightweight ciphers, and proposed some methods to collect some bias (called "wrong-key response profile") from the hypothesis and additionally took advantage of the bias for key recovery.

4.3 Comparison

In Table 1, the five notable results on key recovery attacks using deep learning are compared.

Most of the key recovery attempts have been targeted on toy ciphers. The authors of [4] provided their own toy cipher called HypCipher as the target. HypCipher is a Feistel-structured toy cipher borrowing one of the AES S-boxes, supporting 8-bit key, 16-bit plaintext, and 16-bit ciphertext. As toy ciphers have only few rounds, the attacks targeting the toy ciphers were successful on full-round ciphers.

On the other hand, Gohr used Speck32/64, which was the only non-toy target block cipher among the key recovery attacks. Note that Speck32/64 is a lightweight block cipher having 22 rounds and 32-bit block size. He took a bottom-up approach to start the attack from lower rounds and extend the attack up to 11 rounds, which is a half of the total rounds of Speck32/64.

Table 2: Comparison on cipher emulation attacks using deep learning

Publications		AAAA12 [2]		MMP19 [26]	JM19 [22]	XHY19 [34]
Attack Method		PR	EE	PR/B	PR/B	PR
Target Block Cipher	Name	SDES	SDES	PRESENT	FeW	DES
	Internal Structure	Feistel/Toy	Feistel/Toy	SPN/LW	Feistel/LW	Feistel/Practical
	Block Size (bit)	8	8	64	64	64
Attacked Round (Full Round)		2 (2)	2 (2)	31 (31)	32 (32)	3 (16)
Neural Network Properties	Layer Type	Dense	Dense	Dense	Dense	Dense†
	# of Hidden Layers	32	32	1	2	1
	# of Neurons per Layer	32	32	?	?	1,000
	Activation Function	?	?	Sigmoid	Sigmoid	Varying
Training Methods	Loss Function	MSE	MSE	MSE	MSE	MSE
	Training Data (tuples)	1,024	1,024	10,000	10,000	65,536
	Avg. # of Epoch	1,640	2,861	?	?	350
Result		Success	Success	Failure	Failure	Success (round≤2)

† The authors tried two other network layer types having weaker results compared to dense network.

4.4 Our Claim

Directly training a neural network to predict key from plaintext and ciphertext pairs has been reported to be effective on 2-round SDES which still holds linear relationship between plaintexts and ciphertexts such as follows:

$$\forall x \in \{0,1\}^4, \forall p \in \{0,1\}^8, p' = p \oplus IP^{-1}(x|0000),$$
$$IP(SDES(p) \oplus SDES(p')) = ????|x \quad (1)$$

We claim that the approach is only effective for breaking toy ciphers, because of the complexity of lightweight and practical block ciphers.

The round key guessing approach requires that the model should be built for each round key in the round key space. Speck32/64 uses 16-bit round key, while DES and AES use 48-bit and 32-bit round key, respectively. Therefore, the authors would not be able to extend their approach toward practical ciphers having large key space.

5 Cipher Emulation Attacks

5.1 Description

Cipher emulation attacks are attempts that trying to mimic either the encryption or decryption oracle of a target cipher.

• **Plaintext Restoration Attack (PR)** Given a ciphertext c, the attacker tries to guess bits of the corresponding plaintext p such that $c = Enc_k(p)$ with non-negliglble advantage. Random plaintext and corresponding ciphertext pairs (p_i, c_i) such that $c_i = Enc(k, p_i)$ are given as training data. If an attacker assumes the chosen plaintext attack, the oracle would have been given to the attacker.

• **Bitwise Plaintext Restoration Attack (PR/B)** Plaintext restoration attack can be also deployed by individual bits of plaintext. Pairs of $(p_i[k], c_i)$ is given as training data instead of (p_i, c_i), where $p_i[k]$ is the k^{th} bit of the plaintext p_i.

• **Encryption Emulation Attack (EE)** Given a plaintext p, the attacker tries to guess bits of the corresponding ciphertext c such that $c = Enc(k, p)$ with non-negliglble advantage. Random plaintext and corresponding ciphertext pairs (p_i, c_i) such that $c_i = Enc(k, p_i)$ are given as training data.

5.2 Outcome

Alallayah et al. [2] provided cipher emulation attack results of 12-bit SDES. The attack was performed in both directions: plaintext restoration and encryption emulation.

Mishra et al. [26] attempted to restore plaintext from ciphertext encrypted by a lightweight block cipher PRESENT having 31 rounds with 64-bit block. The authors broke down the entire plaintext restoration problem into smaller problems. Instead of building one large network having 64 output neurons, the authors tried to build 64 binary classifiers predicting each bit on the restored plaintext. As the authors claimed that the models did not produce any meaningful predictions of plaintext, testing reduced-round PRESENT or other successful attacks would be future work. Jain et al. [22] obtained a similar result attacking another lightweight cipher named FeW.

Xiao et al. [34] targeted DES up to two rounds. They tried different types of networks (deep and thin, shallow and fat, and cascade) and different activation functions for plaintext restoration. The authors found out that the shallow and fat network works more properly compared to the other networks, and activation functions do not affect the results. The authors failed to apply the attack on three round DES.

5.3 Comparison

In Table 2, the five notable results from four publications on cipher emulation attacks using deep learning are compared.

Like the key recovery attacks, cipher emulation attacks are successful on toy ciphers, but not yet on full-round lightweight or practical ciphers. Xiao et al. [34]

Table 3: Disputable plaintext restoration attack results on full-round practical ciphers

Publications		Ala12 [3]		HZ18 [21]	
Attack Method		PR	PR	PR	PR
Target Block Cipher	Name	DES	3DES	AES-128	AES-256
	Internal Structure	Feistel/Practical	Feistel/Practical	SPN/Practical	SPN/Practical
	Mode of Operation	ECB	ECB	ECB/CBC	ECB/CBC
	Block Size (bit)	64	64	128	128
Neural Network Properties	Layer Type	Dense	Dense	Cascaded	Cascaded
	# of Hidden Layers	4	4 to 5	4	4
	# of Neurons per Layer	$\leq$512	$\leq$1,024	$\leq$256	$\leq$256
	Activation Function	Sigmoid	Sigmoid	Sigmoid	Sigmoid
Training Methods	Loss Function	MSE	MSE	MSE	MSE
	Training Data (tuples)	2,048	4,096	$\approx$ 1741	$\approx$ 1741
	Avg. # of Epoch	352	239	45	41
Total Error†		0.1110	0.1658	0.1734	0.2052

† Measured on total data including both training and test data

Table 4: Estimation of total error under the overfitting assumption ($\alpha = 0.7$)

Publications	Target Block Cipher	ϵ	Total Error	E
Ala12 [3]	DES (ECB)	0.0317	0.1110	0.1722
	3DES (ECB)	0.0410	0.1658	0.1787
HZ18 [21]	AES-128 (ECB)	0.0358	0.1768	0.1751
	AES-128 (CBC)	0.0627	0.2095	0.1939
	AES-256 (ECB)	0.0198	0.1699	0.1639
	AES-256 (CBC)	0.0580	0.1909	0.1906

succeeded in the attack on 2-round DES but not 3-round DES. However, 2-round DES is definitely not enough to make the avalanche effect takes place. The avalanche effect is a key requirement of block ciphers describing diffusion of the one-bit plaintext difference over the entire bits of ciphertext as rounds go by.

It is a common belief that deeper neural networks are more effective to express complex functions than shallow and fat networks. Xiao et al. [34] examined that shallow and fat networks are better to break 2-round DES, while Alallayah et al. [2] sticked to deep and thin networks to make the attacks to the toy cipher successful.

5.4 Our Claim

Besides the publications mentioned above, two claims [3, 21] on plaintext restoration attack of the practical ciphers exist. Alani [3] claimed that only a simple fully connected network having 4 to 5 fully connected layers were enough to recover plaintext of DES and 3DES. Hu and Zhao [21] followed the same approach to attack AES. However, Xiao et al. [34] reported the Alani's work is not reproducible, while Lagerhjelm concluded the same in his Master's thesis [23]. After trying to reproduce the Alani [3] and Hu and Zhao [21]'s results, we claim that these results are consequences of overfitting, not a success on neural cryptanalysis. Table 3 lists the disputable results briefly.

• **Metrics** We found out that the analysis methods of the publications were possibly misleading the readers. Error values should be measured for test data only, to properly show whether the trained models are indeed successful plaintext restoration attacks. However, the metrics of "outside error"[3] and "total error"[21] are shown as the main results, which are measured by the full dataset containing the training dataset.

• **Overfitting Hypothesis** Suppose that the disputed models overfit to the training data and cannot predict any corresponding plaintext from a ciphertext excluding the training data, with advantage compared to a random prediction. Such models would have test error 0.5 from the baseline of randomly predicting plaintext bit regardless of the ciphertext. With a training data portion α and train error ϵ, the estimated total error E can be calculated by

$$E \approx \epsilon\alpha + 0.5(1 - \alpha) \tag{2}$$

On the MATLAB default, α is equal to 0.7, specifying 15% validation data and another 15% test data.

• **Finding Parameters** Alani [3] gave explicit values of ϵ of the 10 experiments for each cipher type, while not mentioning the α value used in the work. Hu and Zhao [21] did not provide the values; instead, we could reverse calculate ϵ and α from the byte error distribution table to fit in the distribution

$$\alpha B(8, \epsilon) + (1 - \alpha)B(8, 0.5), \tag{3}$$

where $B(n, p)$ stands for the binomial distribution with n trials and p probability.

The training data portion α best describing the error distribution table is approximately 0.68; the default value of 0.7 is good for α. Though the authors specified $\alpha = 0.85$, another 15% validation data were highly

Table 5: Cipher Identification Attacks using Deep Learning

Publications		CV07 [8]	CVSK07 [9]	DST13 [13]
Attack Method		CSI	CSI	PTI / CSI
Target Block Cipher	Names	Enhanced RC6, SEAL	Enhanced RC6, Serpent	MARS, RC6, AES, Serpent, Twofish
	Internal Structure	Feistel, Stream	Feistel, SPN	Feistel or SPN
	Block Size (bit)	128	128	128
Attacked Round (Full Round)		20 (20)	20 (20), 32 (32)	10 to 32 (10 to 32)
Neural Network Properties	Layer Type	Varying	Dense	Self-organizing Map
	# of Hidden Layers	2	2	0
	# of Neurons per Layer	≤10	≤25	100 / 400
Training Methods	Loss Function	?	?	Cosine Angle
	Training Data (MB)†	<2.0	<20.5	<1.5 / <1.2
	# of Epoch	≤1,000	≤5,000	10
Result		Unknown (Train Acc. ≤93%)	Failure	Success

† Amount of data was measured by megabytes due to the data collection methods.

likely to be included in the training data as default. Corresponding ϵ values are displayed in Table 4.

• **Verifying the Hypothesis** The similarity of the estimated total error values and actual total error values supports the overfitting hypothesis except for Alani's DES experiment. Still, to convince readers about the validity of the cryptanalysis attempt, the settings of the DES experiment should be verified by the following.

i. The detailed test environment, such as α, should be specified for a reproduction.

ii. The test error (excluding train and validation data) which is less than 0.5 should be given for any meaningful result. Using the total error may mislead the readers when overfitting occurs.

iii. The data preparation method should be explained in detail, ensuring the readers that the training dataset is disjoint from the validation or test datasets.

In [3, 21], the results only implies that their fully connected neural networks can overfit the training data produced by practical ciphers. The results definitely do not indicate that the plaintext restoration attack on any of the practical ciphers was successful unless the authors provide more crucial information to verify the results.

To train any deep learning models successfully, the loss landscape of the model should have a smooth gradient toward the global minimum. However, in the models of the papers, the loss landscape has a basin area representing parameters resulting 0.5 activation level for each output node regardless of the input (MSE/bit = 0.25). This limits the use of the training algorithm on finding the global minimum for end-to-end attack on full-round lightweight or practical ciphers.

6 Identification Attacks

6.1 Description

• **Plaintext Type Identification Attack (PTI)** Let $P' \subset P$ is the domain of the plaintext type identification attack. Given a partition $\{P_x\}$ of P' and a ciphertext $c = Enc(k, p)$ where $p \in P'$, the attack aims to find j such that $p \in P_j$. Random plaintext and corresponding ciphertext pairs (p_i, c_i) such that $p_i \in P'$ and $c_i = Enc(k, p_i)$ are given as training data.

• **Cipher System Identification Attack (CSI)** Let $P' \subset P$ is the plaintext domain of the cipher system identification attack. Let $\{Enc_x\}$ is a set of block cipher encryption oracles with the same block size. Given a ciphertext c satisfying $c = Enc_j(k, p)$ where $p \in P'$, the attack attempts to find j. Triples of a random plaintext, an oracle index, and the corresponding ciphertext (p_i, j_i, c_i) such that $c_i = Enc_{j_i}(k, p_i)$ where $p_i \in P'$ are given as training data.

6.2 Outcome

Chandra et al. [8] tried to identify whether a block cipher (Enhanced RC6) or a stream cipher (SEAL) was used for encrypting the plaintext. The authors did not conduct test data evaluation despite getting high training accuracy. On their subsequent work [9], the authors conducted a larger experiment involving two block ciphers (Enhanced RC6 and Serpent) and two stream ciphers (LILI-128 and RABBIT). This time, the authors performed test data evaluation for each pair of ciphers and successfully identified RABBIT ciphertexts from the other ciphertexts in the accuracy far higher than 50%. However, identifying the two types of block ciphers was failed.

de Souza et al. [13] experimented on both plaintext type identification and cipher system identification attacks. The authors used words from the Bible in 8 different languages (Portuguese, Spanish, French, German, Danish, Dutch, Greek, and Hebrew) as training data, encrypted in the five AES finalist ciphers. Unlike other publications, the authors used unsupervised learning, since clustering can be implemented using a self-organizing map of neurons. Instead of identifying each ciphertext block, the training data was divided into 6-to-8-kilobyte 'collections' and its encryption was converted to a vector.

6.3 Comparison

In Table 5, the four notable results from three publications on identification attacks using deep learning are compared. The work of de Souza et al. [13] is unique in the aspect of utilizing unsupervised training algorithm for the task. Neural network structures such as self-organizing map had to be deployed for the unsupervised learning tasks. Since the attack used block collections instead of using individual blocks, duplicates in ciphertext blocks would occur and can be leveraged for detection of same block cipher and plaintext.

6.4 Our Claim

Technically not being a deep learning model as no hidden layers are present, the neural network clustering approach in [13] utilized ECB mode of the ciphers, encrypting the same plaintext blocks into the same ciphertext blocks. Therefore, if two ciphertext collections have at least one common ciphertext block, a model can safely conclude that they are from the same cipher algorithm and the same plaintext class except negligible probability. The cosine angle between vectors of two collections becomes less than 90 degrees if and only if this is the case. The results are the attacks on ECB mode of operation, rather than cryptanalysis on block ciphers themselves. This is similar to the other previous results using other machine learning algorithms outside deep learning, such as a decision tree [25] or a support vector machine [14, 10].

7 Conclusion and Future Work

We reviewed recent publications using deep learning to perform cryptanalysis on block ciphers, namely neural cryptanalysis. We enumerated different attacks on the field of neural cryptanalysis. We found out most of the successful reports on neural cryptanalysis target toy ciphers or reduced-round ciphers. The result indicates that no effective methods exist to break full round lightweight or practical block ciphers until now. On some of the attack results on practical block ciphers, we raised disputes on their metrics and methodology to declare success of cryptanalysis.

As the future work, new sophisticated applications of deep learning should be developed in order to break full-round lightweight and practical ciphers. Cryptanalysis using deep learning can be applied to cryptosystems other than block ciphers. For example, in public key cryptosystems (including post-quantum ones), deep learning may provide techniques to more efficiently solve the underlying problems.

Finally, quantum computing becomes a breakthrough technology toward more efficient computation. Quantum computing may enhance the training of existing deep learning models using Grover search or quantum annealing, as the training procedures are equivalent to solving optimization problem. Other quantum machine learning techniques will be developed in the future, which can be applied to faster cryptanalysis of block ciphers and other cryptosystems.

Acknowledgement

This work was partly supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00555, Towards Provable-secure Multiparty Authenticated Key Exchange Protocol based on Lattices in a Quantum World).

References

[1] Musheer Ahmad, Mohammad Zaiyan Alam, Subia Ansari, Dragan Lambić, and Hamed D AlSharari. Cryptanalysis of an image encryption algorithm based on pwlcm and inertial delayed neural network. *Journal of Intelligent & Fuzzy Systems*, 34(3):1323–1332, 2018.

[2] 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.

[3] Mohammed M Alani. Neuro-cryptanalysis of des and triple-des. In *International Conference on Neural Information Processing*, pages 637–646. Springer, 2012.

[4] Ayman MB Albassal and A-MA Wahdan. Neural network based cryptanalysis of a feistel type block cipher. In *International Conference on Electrical, Electronic and Computer Engineering, 2004. ICEEC'04.*, pages 231–237. IEEE, 2004.

[5] Martin R Albrecht and Gregor Leander. An all-in-one approach to differential cryptanalysis for small block ciphers. In *International Conference on Selected Areas in Cryptography*, pages 1–15. Springer, 2012.

[6] Dana Angluin and Michael Kharitonov. When won' t membership queries help? *Journal of Computer and System Sciences*, 50(2):336–355, 1995.

[7] Timo Bartkewitz. Leakage prototype learning for profiled differential side-channel cryptanalysis. *IEEE Transactions on Computers*, 65(6):1761–1774, 2015.

[8] B Chandra and P Paul Varghese. Applications of cascade correlation neural networks for cipher system identification. *World Academy of Science, Engineering and Technology*, 26:312–314, 2007.

[9] B Chandra, P Paul Varghese, Pramod K Saxena, and Shri Kant. Neural networks for identification of crypto systems. In *IICAI*, pages 402–411, 2007.

[10] Jung-Wei Chou, Shou-De Lin, and Chen-Mou Cheng. On the effectiveness of using state-of-the-art machine learning techniques to launch cryptographic distinguishing attacks. In *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence*, pages 105–110. ACM, 2012.

[11] Nicolas T Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 267–287. Springer, 2002.

[12] Moisés Danziger and Marco Aurélio Amaral Henriques. Improved cryptanalysis combining differential and artificial neural network schemes. In *2014 International Telecommunications Symposium (ITS)*, pages 1–5. IEEE, 2014.

[13] William AR De Souza and Allan Tomlinson. A distinguishing attack with a neural network. In *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 154–161. IEEE, 2013.

[14] Aroor Dinesh Dileep and Chellu Chandra Sekhar. Identification of block ciphers using support vector machines. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 2696–2701. IEEE, 2006.

[15] James G Dunham, Ming-Tan Sun, and Judy CR Tseng. Classifying file type of stream ciphers in depth using neural networks. In *The 3rd ACS/IEEE International Conference onComputer Systems and Applications, 2005.*, page 97. IEEE, 2005.

[16] Richard Gilmore, Neil Hanley, and Maire O'Neill. Neural network based attack on a masked implementation of aes. In *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 106–111. IEEE, 2015.

[17] Aron Gohr. Improving attacks on speck32/64 using deep learning. *IACR Cryptology ePrint Archive*, 2019:37, 2019.

[18] Sam Greydanus. Learning the enigma with recurrent neural networks. *arXiv preprint arXiv:1708.07576*, 2017.

[19] Chen He, Kan Ming, Yongwei Wang, and Z Jane Wang. A deep learning based attack for the chaos-based image encryption. *arXiv preprint arXiv:1907.12245*, 2019.

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

[21] Xinyi Hu and Yaqun Zhao. Research on plaintext restoration of aes based on neural network. *Security and Communication Networks*, 2018, 2018.

[22] Aayush Jain and Girish Mishra. Analysis of lightweight block cipher few on the basis of neural network. In *Harmony Search and Nature Inspired Optimization Algorithms*, pages 1041–1047. Springer, 2019.

[23] Linus Lagerhjelm. Extracting information from encrypted data using deep neural networks. Master's thesis, Umeå University, 2018.

[24] Chengqing Li, Shujun Li, Gonzalo Alvarez, Guanrong Chen, and Kwok-Tung Lo. Cryptanalysis of two chaotic encryption schemes based on circular bit shift and xor operations. *Physics Letters A*, 369(1-2):23–30, 2007.

[25] R Manjula and R Anitha. Identification of encryption algorithm using decision tree. In *International Conference on Computer Science and Information Technology*, pages 237–246. Springer, 2011.

[26] Girish Mishra, SVSSNVG Krishna Murthy, and SK Pal. Neural network based analysis of lightweight block cipher present. In *Harmony Search and Nature Inspired Optimization Algorithms*, pages 969–978. Springer, 2019.

[27] Eliya Nachmani, Elad Marciano, Loren Lugosch, Warren J Gross, David Burshtein, and Yair Be'ery. Deep learning methods for improved decoding of linear codes. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):119–131, 2018.

[28] Anugayathiri Pugazhenthi, Nima Karimian, and Fatemeh Tehranipoor. Dla-puf: deep learning attacks on hardware security primitives. In *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2019*, volume 11009, page 110090B. International Society for Optics and Photonics, 2019.

[29] Ronald L Rivest. Cryptography and machine learning. In *International Conference on the Theory and Application of Cryptology*, pages 427–439. Springer, 1991.

[30] Edward F Schaefer. A simplified data encryption standard algorithm. *Cryptologia*, 20(1):77–84, 1996.

[31] Benjamin Timon. Non-profiled deep learning-based side-channel attacks. *IACR Cryptology ePrint Archive*, 2018:196, 2018.

[32] Nhan Duy Truong, Jing Yan Haw, Syed Muhamad Assad, Ping Koy Lam, and Omid Kavehei. Machine learning cryptanalysis of a quantum random number generator. *IEEE Transactions on Information Forensics and Security*, 14(2):403–414, 2018.

[33] Jan Wabbersen. Cryptanalysis of block ciphers using feedforward neural networks. Master's thesis, Georg-August-Universität Göttingen, 2019.

[34] Ya Xiao, Qingying Hao, and Danfeng Yao. Neural cryptanalysis: Metrics, methodology, and applications in cps ciphers. *arXiv preprint arXiv:1911.04020*, 2019.