Bridging Machine Learning and Cryptanalysis via EDLCT

Yi Chen and Hongbo Yu

Department of Computer Science and Technology, Tsinghua University, P.R. China
chenyi19@mails.tsinghua.edu.cn
yuhongbo@mail.tsinghua.edu.cn

Abstract. Machine learning aided cryptanalysis is an interesting but challenging research topic. At CRYPTO'19, Gohr proposed a Neural Distinguisher (ND) based on a plaintext difference. The ND takes a ciphertext pair as input and outputs its class (a real or random ciphertext pair). At EUROCRYPTO'20, Benamira et al proposed a deeper analysis of how two specific NDs against Speck32/64 work. However, there are still three research gaps that researchers are eager to fill in. (1) what features related to a ciphertext pair are learned by the ND? (2) how to explain various phenomena related to NDs? (3) what else can machine learning do in conventional cryptanalysis?
In this paper, we filled in the three *research gaps*: (1) we first propose the Extended Differential-Linear Connectivity Table (EDLCT) which is a generic tool describing a cipher. Features corresponding to the EDLCT are designed to describe a ciphertext pair. Based on these features, various machine learning-based distinguishers including the ND are built. To explore various NDs from the EDLCT view, we propose a Feature Set Sensitivity Test (FSST) to identify which features may have a significant influence on NDs. Features identified by FSST share the same characteristic related to the cipher's round function. Surrogate models of NDs are also built based on identified features. Experiments on Speck32/64 and DES confirm that features corresponding to the EDLCT are learned by NDs. (2) We explain phenomena related to NDs via EDLCT. (3) We show how to use machine learning to search differential-linear propagations $\Delta \rightarrow \lambda$ with a high correlation, which is a tough task in the differential-linear attack. Applications in Chaskey and DES demonstrate the advantages of machine learning.
Furthermore, we provide some optional inputs to improve NDs.

Keywords: Machine learning · Distinguishing attack · EDLCT · Neural distinguisher · Differential-Linear attack.

1 Introduction

1.1 Priliminaries and Motivations

Supervised learning. Machine learning is a family of algorithms that can automatically learn hidden knowledge from collected samples for performing

some tasks [17]. It can be divided into supervised learning [20], unsupervised learning [2], and reinforcement learning [13] basically.

Supervised learning is widely used in solving classification and regression problems. In this field, each sample is represented by a feature set $\mathcal{X}_i, i \in [1, n]$ and labeled with a label Y related to the final task. Supervised learning aims at training a model $\mathcal{M}$ over numerous labeled samples. Given a sample $X = [x_1, \cdots, x_n]$ where each x_i is an instance drawn from $\mathcal{X}_i, i \in [1, n]$, $\mathcal{M}$ will output a prediction label.

Model $\mathcal{M}$ can be explainable models (eg. logistic regression [15]) or unexplainable ones (eg. neural networks [12]). Explainable models usually make a prediction based on input features $\mathcal{X}_i, i \in [1, n]$. The prediction of unexplainable tools usually contains two stages: generate new features from input features, and make a prediction based on these new features.

The performance of the model on the final task depends on two factors: first, whether features for prediction-making are related to Y. Second, whether M has learned the accurate relation between these features and Y.

In cryptography, many issues can be modeled as classification or regression problems. For example, regarding encryption as a regression task, Greydanus successfully simulated a simplified version of Enigma by recurrent neural networks [11]. Modeling signal processing as a classification problem, researchers have applied supervised learning to side-channel cryptanalysis [3, 14].

Distinguishing attack. Let $E : \{0,1\}^m \to \{0,1\}^m$ be an encryption function. The distinguishing attack aims at judging whether E is a pseudo-random permutation.

To solve this problem, a typical technique in conventional cryptanalysis is the differential attack [7]. Here we choose a plaintext difference ΔP and generate a plaintext pair $(P_0, P_1)|P_0 \oplus P_1 = \Delta P$. If there exists a ciphertext difference $\Delta C = C_0 \oplus C_1$ that makes $Pr(\Delta P \xrightarrow{E} \Delta C) > 2^{-m}$ hold, then E is not a pseudo-random permutation.

The mechanism behind the differential attack is the same as supervised learning. The judging about E is a binary classification problem. The difference propogation probability $Pr(\Delta P \xrightarrow{E} \Delta C)$ is the feature. If it exceeds a threshold 2^{-m}, the prediction is that E is not a pseudo-random permutation.

Thus, the distinguishing attack can be done by supervised learning. According to the type of model $\mathcal{M}$, it can be achieved from two directions. The first direction is to design features for explainable models. This is still a *research gap* since it is hard to design related features. The second direction is to use unexplainable models to explore related features.

Neural distinguisher. At CRYPTO'19, Gohr made the first step in the second direction by proposing a ND based on a plaintext difference constraint [10].

The ND needs to distinguish two classes of ciphertext pairs (C_0, C_1): real pairs (the label is $Y = 1$) corresponding to plaintext pairs with a difference ΔP, random pairs (the label is $Y = 0$) corresponding to plaintext pairs with

a random difference. We input a ciphertext pair $X = C_0||C_1$, and the ND will predict its label. If the ND achieves a distinguishing accuracy higher than 0.5, the cipher E is not a pseudo-random permutation.

Adopting a deep residual network as $\mathcal{M}$, Gohr built four NDs against round reduced Speck32/64. Surprisingly, these NDs achieve higher accuracy than pure differential distinguishers. Except for the ciphertext difference, NDs learn some unknown features that result in advantages in accuracy. Due to the black-box nature of neural networks, Gohr didn't figure out what the unknown features are, which is an important open problem.

CNN and its feature learning mechanism. The above open problem related to NDs needs to be explored based on the feature learning mechanism of the Convolutional Neural Network (CNN) since the deep residual network adopted by Gohr is a CNN.

Although there are many tricks for designing various CNNs now, CNNs share the same feature learning mechanism. Fig. 1 shows a classic four-layer CNN.

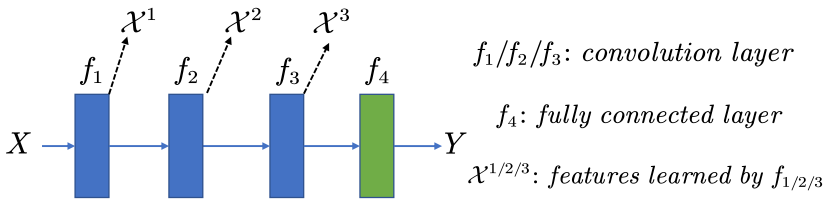

Fig. 1. A classic four-layer CNN.

The learning on features is accomplished by convolution layers. The function of the $(i+1)$-th convolution layer is to learn new features $\mathcal{X}^{i+1}$ from $\mathcal{X}^i$. For deep residual networks, $\mathcal{X}^{i+1}$ is learned from features (eg. $\mathcal{X}^i, \mathcal{X}^{i-1}, \cdots$) learned by multiple convolution layers.

In the machine learning community, there are three important concepts: *low-level* features (eg. $\mathcal{X}^1$) learned by a shallow layer, *middle-level* features (eg. $\mathcal{X}^2$) learned by a middle layer, and *high-level* features (eg. $\mathcal{X}^3$) learned by a deep layer [19]. *Low-level* features are related to the input. *High-level* features are abstract features related to the prediction label, while *middle-level* features play the role of a bridge.

In the background of NDs, *low-level* features could be the ciphertext difference $C_0 \oplus C_1$. *Middle-level* features could be features related to cipher E, such as the difference propagation probability $p = Pr(C_0 \oplus C_1 = \Delta C|\Delta P)$ where ΔC is a specific value. *High-level* features could be the posterior probability $Pr(Y = 1|p)$.

High-level features can be used to make an accurate prediction. But they can not directly provide any insights related to E since they are related to the prediction label instead of E. The work introduced by Benamira et al [6] affirms this contradiction.

Surrogate model. To understand how NDs proposed by Gohr work, Benamira et al [6] built surrogate models of NDs.

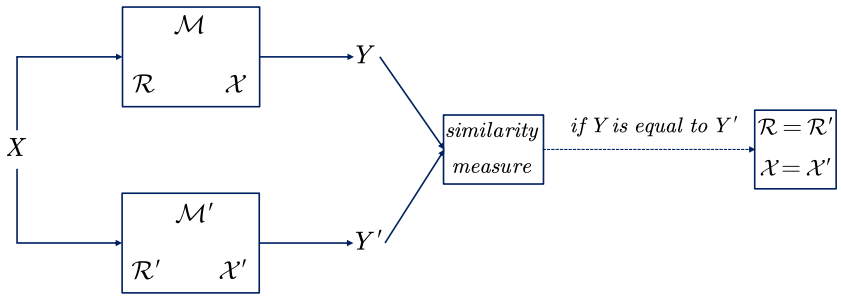

Fig. 2. The mechanism of the surrogate model. $\mathcal{M}/\mathcal{M}^{'}$: Target model / Surrogate model. $\mathcal{X}/\mathcal{X}^{'}$: Features used for decision-making. $\mathcal{R}/\mathcal{R}^{'}$: Decision rules. X : The raw input. $Y/Y^{'}$: The prediction of $\mathcal{M}/\mathcal{M}^{'}$.

Fig.2 summarizes the mechanism of the surrogate model [18]. If a surrogate model $\mathcal{M}^{'}$ can always make a prediction $Y^{'}$ that is equivalent to the prediction Y given by the target model $\mathcal{M}$, it means that the decision rules $\mathcal{R}, \mathcal{R}^{'}$ of two models are the same. Furthermore, the features $\mathcal{X}, \mathcal{X}^{'}$ used for decision-making are treated as the same features.

In general, it is difficult to build a perfect surrogate model. Thus, the matching ratio between the prediction given by $\mathcal{M}^{'}$ and the prediction given by $\mathcal{M}$ is used to judge whether a surrogate model is efficient. The higher the matching ratio is, the better the surrogate model is.

To obtain good surrogate models of NDs, Benamira et al [6] proposed a concept named Masked Output Distribution Table (M-ODT). With the help of M-ODT, a ciphertext pair $X = C_0||C_1$ is finally represented by multiple posterior probabilities $Pr(Y = 1|f(X))$ where $f(X)$ is a transformation on X. By training a classifier on these probabilities, they obtained good surrogate models of 5/6-round ND against Speck32/64. However, researchers are still not able to obtain more useful insights since the posterior probabilities are *high-level* features.

To obtain some insights related to cipher E, we should focus on *middle-level* or even *low-level* features.

Our targets. According to the feature learning mechanism of CNNs, Gohr's NDs should have learned a class of features describing a ciphertext pair. Moreover, this class of features should be related to an unknown tool describing the cipher.

In this paper, our core target is finding such a class of features as well as the unknown tool. The requirement is that the tool and the class of features could be used to explain phenomena related to NDs. Besides, a reasonable explanation about how NDs work could be provided.

The ND proposed by Gohr is a generic distinguisher that can be applied to other ciphers [9]. Thus, we are curious about the following issues: 1) how does

the nature of a cipher E affect this class of features? 2) Can we obtain some interesting insights, such as how to improve cryptanalysis techniques?

1.2 Contributions

The ND proposed by Gohr is built on a plaintext difference. Benamira et al [6] also showed NDs may learn linear or differential-linear characteristics. Thus, we first analyze the Differential-Linear Connectivity Table (DLCT) [1], and find that the DLCT can not describe the cipher completely.

Inspired by this research, we propose the Extended Differential-Linear Connectivity Table (EDLCT). To describe a ciphertext pair, we design a class of features corresponding to the EDLCT. Various machine learning-based distinguishers are built with this class of features. The EDLCT not only fills in the research gap between supervised learning and distinguishing attack, but also provides a new way to explore NDs.

The EDLCT is the desired tool describing the cipher. The features corresponding to the EDLCT are learned by NDs. To confirm our opinion, we explore various NDs from the EDLCT view.

The exploration is carried out in three areas. First, a Feature Set Sensitivity Test (FSST) is proposed to identify which features corresponding to the EDLCT have a significant influence on the accuracy of NDs. Second, we analyze the relationship between identified features and the round function of the cipher. Third, surrogate models are built with identified features.

By performing experiments on NDs against two ciphers (Speck32/64 and DES), we find that the features having a significant influence share the same characteristic related to the round function. Although the features corresponding to the EDLCT are not directly related to the final prediction, it's still possible to build good surrogate models using these features.

We investigate the phenomena related to NDs that researchers have discovered so far. These phenomena related to NDs are all explained via EDLCT.

Furthermore, we obtain two *useful insights* from the EDLCT view. First, we show how to use machine learning to search differential-linear propagation $\Delta \to \lambda$ with a high correlation, which is a tough task in the differential-linear attack. Applying this technique to Chaskey and DES, we find some better differential-linear propagations. This technique also proves that the differential-linear distinguisher against 8-round DES [1] is the optimal one. Second, we can improve NDs by changing the input describing a ciphertext pair, NDs can be accelerated without reducing the accuracy.

Outline. We introduce the EDLCT in Section 2. How to build various machine learning-based distinguishers via EDLCT is shown in Section 3. From the EDLCT view, we explore various NDs in Section 4 and explain phenomena related to NDs in Section 5. In Section 6, we show how to search differential-linear propagations via machine learning. In Section 7, we present a method to improve NDs.

2 The Extended Differential-Linear Connectivity Table

To better introduce our work, we adopt the definition of correlation [5]

$$\boldsymbol{Cor}_{x\in S}[f(x)] := \frac{1}{|S|}\sum_{x\in S}(-1)^{f(x)}, \tag{1}$$

where S is a sample set. The DLCT is first introduced to improve the conventional differential-linear attack [1].

Definition of the DLCT [1]. Let $E : \{0,1\}^m \to \{0,1\}^m$ be an encryption function. The DLCT of E is an $2^m \times 2^m$ table whose rows correspond to input differences to E and whose columns correspond to bit masks of outputs of E. Formally, for $\Delta \in \{0,1\}^m$ and $\lambda \in \{0,1\}^m$, the DLCT entry (Δ, λ) is

$$DLCT_E(\Delta,\lambda) \triangleq |\{P\,|\lambda \cdot E(P) = \lambda \cdot E(P \oplus \Delta)\}| - 2^{m-1} \tag{2}$$

When E is a pseudo-random permutation, $DLCT_E(\Delta,\lambda)$ should always be 0 for each (Δ,λ). For convenience, we say DLCT is balanced in (Δ,λ) if $DLCT_E(\Delta,\lambda) = 0$. If the DLCT of E could be constructed, the distinguishing attack on E can be directly performed. But this is an unpractical target so far.

The DLCT entry (Δ,λ) can be described with the correlation. We say that the differential-linear propagation $\Delta \xrightarrow{E} \lambda$ is satisfied with the correlation

$$\boldsymbol{Cor}_{P\in\mathbb{F}_2^m}[(\lambda \cdot E(P)) \oplus (\lambda \cdot E(P \oplus \Delta))] = \frac{DLCT_E(\Delta,\lambda) \times 2}{2^m}. \tag{3}$$

Definition of the EDLCT. Let $E : \{0,1\}^m \to \{0,1\}^m$ be an encryption function. The EDLCT of E is an $2^m \times 2^m \times 2^m$ table. Formally, for $\Delta \in \{0,1\}^m$, $\lambda_0 \in \{0,1\}^m$ and $\lambda_1 \in \{0,1\}^m$, the EDLCT entry $(\Delta, \lambda_0, \lambda_1)$ is

$$EDLCT_E(\Delta,\lambda_0,\lambda_1) \triangleq |\{P\,|\lambda_0 \cdot E(P) = \lambda_1 \cdot E(P \oplus \Delta)\}| - 2^{m-1} \tag{4}$$

Similarly, we say EDLCT is balanced in $(\Delta,\lambda_0,\lambda_1)$ if $EDLCT_E(\Delta,\lambda_0,\lambda_1) = 0$. Besides, we also say that the differential-linear propagation $\Delta \xrightarrow{E} (\lambda_0,\lambda_1)$ is satisfied with the correlation

$$\boldsymbol{Cor}_{P\in\mathbb{F}_2^m}[(\lambda_0 \cdot E(P)) \oplus (\lambda_1 \cdot E(P \oplus \Delta))] = \frac{EDLCT_E(\Delta,\lambda_0,\lambda_1) \times 2}{2^m}. \tag{5}$$

It is clear that the DLCT only covers entries of the EDLCT with $\lambda_0 = \lambda_1$.

Compared with the DLCT, the EDLCT can describe the encryption function E more completely. When E is not a pseudo-random permutation, the entries $(\Delta,\lambda_0,\lambda_1)|\lambda_0 \neq \lambda_1$ that are not covered by the DLCT may also not be 0. Table 1 shows the partial EDLCT of the fourth S-box $S4$ of DES. The difference constraint is $\Delta = 0x10$. As expected, the EDLCT is not balanced in many entries $(\Delta,\lambda_0,\lambda_1)$ where $\lambda_0 \neq \lambda_1$.

The EDLCT is balanced in $(\Delta,\lambda_0,\lambda_1)$ where $\lambda_0 = 0$ or $\lambda_1 = 0$. Besides, the EDLCT is symmetric since the entry $(\Delta,\lambda_0,\lambda_1)$ is always equal to $(\Delta,\lambda_1,\lambda_0)$.

Table 1. The partial $EDLCT$ of $S4$ of DES. $\Delta = 0x10$

$\lambda_0 \backslash \lambda_1$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	-8	0	0	-8	4	4	8	-12	0	8	-4	4	4	4	-4
2	0	0	-8	0	-12	-4	-4	4	8	-4	-4	8	0	4	-4	0
3	0	0	0	-16	0	4	4	0	-4	-4	4	4	-8	4	-4	0
4	0	-8	-12	0	-8	-4	4	4	0	4	-4	4	0	8	-4	0
5	0	4	-4	4	-4	0	-4	-4	-8	-4	8	0	4	0	0	-8
6	0	4	-4	4	4	-4	8	4	4	-16	4	4	-4	-4	4	-8
7	0	8	4	0	4	-4	4	0	4	0	0	0	4	16	-4	-4
8	0	-12	8	-4	0	-8	4	4	-8	0	-4	-4	0	4	8	-4
9	0	0	-4	-4	4	-4	-16	0	0	8	4	4	4	-4	0	-8
A	0	8	-4	4	-4	8	4	0	-4	4	0	0	4	0	4	8
B	0	-4	8	4	4	0	4	0	-4	4	0	0	4	-4	-16	0
C	0	4	0	-8	0	4	-4	4	0	4	4	4	-16	4	0	0
D	0	4	4	4	8	0	-4	16	4	-4	0	-4	4	0	0	0
E	0	4	-4	-4	-4	0	4	-4	8	0	4	-16	0	0	0	-4
F	0	-4	0	0	0	-8	-8	-4	-4	-8	8	0	0	0	-4	16

3 Bridge Supervised Learning and Distinguishing Attack via EDLCT

The EDLCT provides a class of features that can bridge the gap between supervised learning and distinguishing attack.

3.1 Features Corresponding to the EDLCT

Features describing the ciphertext pair. Given two linear masks $\lambda_0, \lambda_1 \in \mathbb{F}_2^m$, a unique feature describing a ciphertext pair $(C_0, C_1), C_0, C_1 \in \mathbb{F}_2^m$ is

$$\mathcal{X}(\lambda_0, \lambda_1, C_0, C_1) = (C_0 \cdot \lambda_0) \oplus (C_1 \cdot \lambda_1).$$

Traversing the linear mask pair, we can obtain 2^{2m} features describing a ciphertext pair. For convenience, we denote this class of features as

$$\mathcal{X}(\lambda_0, \lambda_1) = (C_0 \cdot \lambda_0) \oplus (C_1 \cdot \lambda_1) \in \{0, 1\}. \tag{6}$$

Relations to the EDLCT. From the machine learning perspective, each EDLCT entry $(\Delta, \lambda_0, \lambda_1)$ is a *middle-level* feature describing the encryption function E.

Based on the 2^{2m} features $\mathcal{X}(\lambda_0, \lambda_1)$, each EDLCT entry $(\Delta, \lambda_0, \lambda_1)$ can be represented as

$$EDLCT_E(\Delta, \lambda_0, \lambda_1) = (Pr(\mathcal{X}(\lambda_0, \lambda_1) = 0|\Delta) - 0.5) \times 2^m, \tag{7}$$

where $C_0 = E(P)$ and $C_1 = E(P \oplus \Delta)$. In other words, $\mathcal{X}(\lambda_0, \lambda_1)$ is a class of features corresponding to the EDLCT. Then we obtain two important properties.

Property 1. We say the features $\mathcal{X}(\lambda_0, 0), \mathcal{X}(0, \lambda_1)$ are trivial features. Since the EDLCT is balanced in $(\Delta, \lambda_0, \lambda_1)$ where $\lambda_0 = 0$ or $\lambda_1 = 0$, trivial features obey the uniform distribution.

Property 2. Non-trivial features $\mathcal{X}(\lambda_0, \lambda_1)|\lambda_0 \neq 0, \lambda_1 \neq 0$ can be generated from two trivial features since $\mathcal{X}(\lambda_0, \lambda_1) = \mathcal{X}(\lambda_0, 0) \oplus \mathcal{X}(0, \lambda_1)$. But non-trivial features could obey the non-uniform distribution.

Property 1 and *Property 2* explain some phenomena related to NDs. Besides, we know the following transformations also hold

$$\begin{aligned}\rho &= Pr(\mathcal{X}(\lambda_0, \lambda_1) = 0|\Delta) - Pr(\mathcal{X}(\lambda_0, \lambda_1) = 1|\Delta) \\ &= \frac{|\{P|(C_0 \cdot \lambda_0) \oplus (C_1 \cdot \lambda_1) = 0\}| - |\{P|(C_0 \cdot \lambda_0) \oplus (C_1 \cdot \lambda_1) = 1\}|}{2^m} \\ &= \boldsymbol{Cor}_{P \in \mathbb{F}_2^m}[(\lambda_0 \cdot C_0) \oplus (\lambda_1 \cdot C_1)].\end{aligned}$$

Thus, the larger the correlation is, the larger the value of ρ is. This leads to a useful conclusion adopted in this article. **Conclusion 1** is verified in Section 3.3.

Conclusion 1 *Consider a machine learning-based distinguisher built on a plaintext difference Δ. If a feature $\mathcal{X}(\lambda_0, \lambda_1)$ has a significant influence on this distinguisher, then $\Delta \xrightarrow{E} (\lambda_0, \lambda_1)$ is satisfied with a high correlation.*

Relation to the ciphertext difference. NDs can learn the ciphertext difference. Next, we check whether the ciphertext difference can be generated from the class of features corresponding to the $EDLCT$.

Consider a ciphertext pair (C_0, C_1) where $C_0, C_1 \in \mathbb{F}_2^m$, the ciphertext difference $\Delta C = C_0 \oplus C_1$ can be described as a set of conditions

$$\mathcal{X}(\lambda_0^i, \lambda_1^i) = \Delta C[i], i \in [0, m-1] \tag{8}$$

where $\lambda_0^i = \lambda_1^i = 1 \ll i$.

Thus, once the class of features $\mathcal{X}(\lambda_0, \lambda_1)$ is learned by NDs, the ciphertext difference can be learned.

3.2 Build Machine Learning-based Distinguishers via EDLCT

Now, the distinguishing attack can be tackled in both directions introduced in Section 1. Specifically, machine learning-based distinguishers can be built using the class of features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT.

Build distinguishers with explainable models. Explainable models $\mathcal{M}$ share the same working mechanism. Therefore, the method of building distinguishers is generic.

The whole process is divided into two stages: we first select n features $\mathcal{X}(\lambda_0^i, \lambda_1^i)$, $i \in [1, n]$ corresponding to the EDLCT, train the model $\mathcal{M}$. In the second stage, we will generate two datasets (one for training, one for testing) according to the selected features. Algorithm 1 summarizes the dataset generation.

How to combine these features to obtain the desired output is determined by the explainable model $\mathcal{M}$ itself. In this paper, we introduce our work by taking the Logistic Regression (LR) as an example of explainable models.

Algorithm 1 Generate the training/test dataset dataset via EDLCT.

Require: the cipher, E; the difference constraint, Δ; the number of samples, N; n features corresponding to the EDLCT, $\mathcal{X}(\lambda_0^j, \lambda_1^j), j \in [1, n]$.

Ensure: a dataset consisting of N samples.

1: Randomly generate N plaintexts P_0^i and N sample labels $Y_i, i \in [1, N]$;
2: **for** $i = 1$ to N **do**
3: **if** $Y_i = 1$ **then**
4: $P_1^i = P_0^i \oplus \Delta$;
5: **else**
6: Randomly generate a plaintext $P_1^i \neq P_0^i \oplus \Delta$;
7: **end if**
8: $C_0^i = E(P_0^i)$, $C_1^i = E(P_1^i)$;
9: Compute the value x_j^i of each feature $\mathcal{X}(\lambda_0^j, \lambda_1^j)$:

$$x_j^i = (\lambda_0^j \cdot C_0^i) \oplus (\lambda_1^j \cdot C_1^i), j \in [1, n];$$

10: Save $(X_i = [x_1^i, \cdots, x_n^i], Y_i)$ as the i-th sample;
11: **end for**
12: Return $(X_i, Y_i), i \in [1, N]$ as the dataset.

The LR can be expressed as

$$z = \frac{1}{1 + e^{-W^T \times X}} = \frac{1}{1 + e^{-(w_0 + w_1 \times x_1 + \cdots + w_n \times x_n)}}, \quad 0 \leqslant z \leqslant 1, \tag{9}$$

where X is the feature vector consisting of n feature values, and w_i is the weight of the feature value $x_i, i \in [1, n]$. When we build distinguishers based on the LR, the predicted label of the input sample is 1 if $z > 0.5$.

The reasons to choose LR as an example are: (1) it's very fast to train the LR and (2) the feature weights $W^T = [w_0, \cdots, w_n]$ can directly tell which features have a higher influence on the prediction.

Build distinguisher with unexplainable models. Actually, it is a tough task to artificially select features for building distinguishers. Unexplainable models such as neural networks provide an alternative way to build distinguishers via EDLCT.

The whole process is divided into two stages: (1) we first design a representation X for a ciphertext pair, (2) train the model $\mathcal{M}$. The requirement is that any feature $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT can be derived from X. For example, $X = C_0||C_1$ is an applicable representation, which is adopted by Gohr's NDs as the input.

Here we provide an explanation about how Gohr's NDs work. The NDs have the ability of generating new features from some features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT. By selecting different features $\mathcal{X}(\lambda_0, \lambda_1)$, more new features are generated for the final prediction.

Since the EDLCT is a generic tool describing ciphers, we have the following conjecture

Conjecture 1. When building NDs for different ciphers, which features $\mathcal{X}(\lambda_0, \lambda_1)$ have a higher influence on NDs is determined by the unique nature of the cipher.

Conjecture 1 is challenged by exploring NDs against two round reduced ciphers (DES, Speck32/64) in Section 4.

3.3 Property of Machine Learning-based Distinguishers

For machine learning-based distinguishers, the most important target is to achieve a high distinguishing accuracy.

Machine learning-based distinguishers share the same property:

Property 3. If more related features are provided, the machine learning-based distinguisher is more likely to achieve better distinguishing accuracy.

To better explain *Property 3*, we have trained a series of distinguishers (Algorithm 1, $N = 10^7$) by taking the fourth S-box $S4$ of DES as a toy encryption function E. Table 2 summarizes these distinguishers.

Table 2. Summary of machine learning-based distinguishers against $S4$ of DES. acc: distinguishing accuracy. Feature weights only retain 2 decimal places. $\Delta = 0x10$.

ID	$\mathcal{M}$	X / $\mathcal{X}(\lambda_0, \lambda_1)$	$W^T = [w_1, \cdots, w_n, w_0]$	acc
1	LR	$\mathcal{X}(0x3, 0x3)$	$[1.1, -0.69]$	0.625
2	LR	$\mathcal{X}(0x3, 0x3), \mathcal{X}(0xc, 0xc)$	$[0.96, 0.96, -1.21]$	0.688
3	LR	all the 15 $\mathcal{X}(\lambda, \lambda)$, $\lambda \in [0x1, 0xf]$	[3.29, 3.29, 3.5, 3.29, 0, -0.2, 0, 3.29, -0.2, 0, 0, 3.5, 0, 0, -0.7, -12.48]	0.718
4	LR	all the 256 $\mathcal{X}(\lambda_0, \lambda_1)$, $\lambda_0, \lambda_1 \in \mathbb{F}_2^4$	-	0.925
5	NN	$X = C_0 \| C_1$	-	0.925

By adding more features $\mathcal{X}(\lambda_0, \lambda_1)$, the distinguisher can achieve better accuracy. The accuracy comparison of the first four distinguishers (ID = 1,2,3,4) fully reflects this property. In fact, the idea of adding more features has been verified in [6,9]. By adopting batches of ciphertexts instead of pairs, the accuracy of NDs is improved since the input contains more features.

Because the encryption function E is only an S-box of DES, $N = 10^7$ is sufficient to cover all possible samples including the two classes of samples. Thus the distinguishing accuracy can be regarded as the upper bound under corresponding settings.

For example, since $EDLCT_E(0x10, 0x3, 0x3) = -16 < 0$ (see Table 1), if we predict that the label of $(C_0, C_1)|C_0 \cdot 0x3 \neq C_1 \cdot 0x3$ is 1, the ideal distinguishing accuracy is

$$\frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{48}{64} = 0.625$$

It's clear that the first distinguisher (ID = 1) can achieve the ideal distinguishing accuracy. Besides, LR also captures the same decision rule since $w_1 = 1.1 > |w_0| = 0.69$.

Besides, there is another interesting phenomenon in Table 2. Let $\mathcal{M}$ be the deep residual network proposed by Gohr and $X = C_0||C_1$. The final distinguishing accuracy is 0.925. If we adopt LR as the model $\mathcal{M}$ and exploit all the 256 features $\mathcal{X}(\lambda_0, \lambda_1)$, the accuracy is also 0.925. This phenomenon implies that the neural network has learned all the features corresponding to EDLCT.

As introduced, the feature weights W^T of LR can tell which features have a higher influence. Taking the third distinguisher (ID=3) as an example, we know that the feature $\mathcal{X}(0x3, 0x3)$ has a higher influence than $\mathcal{X}(0x1, 0x1)$ because $w_3 = 3.5 > w_1 = 3.29$. Table 1 also shows that

$$|EDLCT_E(\Delta, 0x3, 0x3)| = 16 > |EDLCT_E(\Delta, 0x1, 0x1)| = 8.$$

As long as the two feature weights are higher than 0 or lower than 0 simultaneously, we can apply the above rule.

The above rule also verifies **Conclusion 1** presented in Section 3.1.

4 Explore Neural Distinguishers from the EDLCT View

The EDLCT provides an explanation about how NDs work. In this section, we will explore various NDs from the EDLCT view.

Three areas are mainly explored. First, we identify which features $\mathcal{X}(\lambda_0, \lambda_1)$ have a significant influence on NDs. Second, we explore how the nature of a cipher affects the influence of these features. Third, we build surrogate models based on identified features.

4.1 Feature Set Sensitivity Test

To efficiently identify features that may have a significant influence on NDs, we propose a scheme denoted as *Feature Set Sensitivity Test* (FSST).

The idea of FSST. Suppose that the decision-making rules of a ND are related to a feature $\mathcal{X}(\lambda_0, \lambda_1)$, the distinguishing accuracy of the ND will be affected if we randomize this feature. If the feature is not exploited by the ND, the distinguishing accuracy will not be affected.

Given a feature set Ω, we randomize all the features belonging to Ω simultaneously. If all the features $\mathcal{X}(\lambda_0, \lambda_1) \in \Omega$ are not exploited, the distinguishing accuracy will not be affected. Then we can filter a set of features at a time.

To select a set of features that can be randomized simultaneously, we can describe features corresponding to the EDLCT in a new form. Let $X = C_0||C_1$, and $\lambda = \lambda_0||\lambda_1$. It follows that

$$\mathcal{X}(\lambda_0, \lambda_1) = (C_0 \cdot \lambda_0) \oplus (C_1 \cdot \lambda_1) = X \cdot \lambda = \left(\underset{i \in S}{\oplus} X[i]\right) \oplus \left(\underset{j \notin S}{\oplus} X[j]\right) \quad (10)$$

where S is a bit index subset that can divide ciphertext bits selected by λ into two parts. Specifically, $\left(\underset{i \in S}{\oplus} X[i]\right)$ is the unfree part related to S and $\left(\underset{j \notin S}{\oplus} X[j]\right)$ is the free part.

Once S is determined, we obtain a feature set Ω. All the features belonging to Ω have the same unfree part. For $E : \{0,1\}^m \to \{0,1\}^m$ and a fixed Δ, if S contains n bit indexes, there are 2^{2m-n} features belonging to Ω.

For example, let $S = \{m-1, 2m-1\}$. The feature set Ω contains the following 2^{2m-2} features

$$\Omega = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0, \lambda_1 \in \mathbb{F}_2^m; \lambda_0[m-1] = 1; \lambda_1[m-1] = 1\}.$$

After we obtain a feature set Ω, randomizing Ω is equivalent to randomizing the unfree part related to S.

Implementation of FSST. Suppose the distinguishing accuracy of a ND is acc_1. After randomizing a feature set Ω, we denote the new accuracy of the ND as acc_2. Then $acc_1 - acc_2$ is defined as the *Feature Set Sensitivity* (FSS) that is used to measure the influence of the feature set on the ND.

To estimate the FSS of Ω, the accuracy change is solely caused by Ω. Actually, this is not easy when S is a large set. To explain how to implement FSST, we focus on $|S| = 1$ and $|S| = 2$ at this initial stage.

When $|S| = 1$, only one feature set Ω is involved. Let $S = \{i\}$. The feature $\mathcal{X}(\lambda_0, \lambda_1)$ is described as

$$\mathcal{X}(\lambda_0, \lambda_1) = X[i] \oplus \left(\bigoplus_{j \notin S} X[j] \right). \tag{11}$$

By XOR $X[i]$ with a random binary mask η, all the features belonging to Ω are randomized since

$$Pr(\eta = 0) = \frac{1}{2} \Rightarrow Pr\left(\eta \oplus X[i] \oplus \left(\bigoplus_{j \notin S} X[j] \right) = 0 \right) = \frac{1}{2}.$$

Features that do not belong to Ω are not randomized by the operation above.

Thus, when $|S| = 1$, the FSS of Ω is estimated as follows:

a) Generate N ciphertext pairs (C_0, C_1): half corresponding to plaintext pairs with a difference Δ, half corresponding to random plaintext pairs.
b) Test the distinguishing accuracy acc_1 of the ND over (C_0, C_1).
c) Generate N random binary masks η, and get the masked ciphertext pairs

$$C_{1,0} || C_{1,1} \leftarrow (C_0 || C_1) \oplus (\eta \ll i)$$

where $S = \{i\}$.
d) Test the distinguishing accuracy acc_2 of the ND over $(C_{1,0}, C_{1,1})$.
e) Return $acc_1 - acc_2$ as the FSS.

When $|S| = 2$, three features sets $\Omega_1, \Omega_2, \Omega_3$ are involved. For convenience, let $S = \{i_1, i_2\}$. It yields that

$$\Omega_1 = \left\{ \mathcal{X}(\lambda_0, \lambda_1) | \mathcal{X}(\lambda_0, \lambda_1) = X[i_1] \oplus \left(\bigoplus_{j \notin \{i_1\}} X[j] \right) \right\}, \tag{12}$$

$$\Omega_2 = \left\{ \mathcal{X}(\lambda_0, \lambda_1) | \mathcal{X}(\lambda_0, \lambda_1) = X[i_2] \oplus \left(\underset{j \notin \{i_2\}}{\oplus} X[j] \right) \right\}, \tag{13}$$

$$\Omega_3 = \left\{ \mathcal{X}(\lambda_0, \lambda_1) | \mathcal{X}(\lambda_0, \lambda_1) = X[i_1] \oplus X[i_2] \oplus \left(\underset{j \notin S}{\oplus} X[j] \right) \right\}. \tag{14}$$

The third set Ω_3 is the target feature set to be randomized. Additionally, it is clear that $|\Omega_1| > |\Omega_3|$ and $|\Omega_2| > |\Omega_3|$ both hold.

To test the influence of the target feature set on the distinguishing accuracy, we will conduct the following steps:

1. Randomize Ω_1, Ω_2 by XOR $X[i_1], X[i_2]$ with one mask
$$X[i_1] \leftarrow X[i_1] \oplus \eta_1, \quad X[i_2] \leftarrow X[i_2] \oplus \eta_1.$$
2. Randomize $\Omega_1, \Omega_2, \Omega_3$ by XOR $X[i_1], X[i_2]$ with two independent masks
$$X[i_1] \leftarrow X[i_1] \oplus \eta_2, \quad X[i_2] \leftarrow X[i_2] \oplus \eta_3.$$

Then the distinguishing accuracy difference in the two stages is caused only by Ω_3.

Hence, when $|S| = 2$, the FSS of Ω is estimated as follows:

a) Generate N ciphertext pairs (C_0, C_1): half corresponding to plaintext pairs with a difference Δ, half corresponding to random plaintext pairs.
b) Generate N random binary masks η_1, and get the masked ciphertext pairs
$$C_{1,0}||C_{1,1} \leftarrow (C_0||C_1) \oplus (\eta_1 \ll i_1) \oplus (\eta_1 \ll i_2).$$
where $S = \{i_1, i_2\}$.
c) Test the distinguishing accuracy acc_1 of the ND over $(C_{1,0}, C_{1,1})$.
d) Generate N random binary mask pairs (η_2, η_3), and get the masked ciphertext pairs
$$C_{2,0}||C_{2,1} \leftarrow (C_0||C_1) \oplus (\eta_2 \ll i_1) \oplus (\eta_3 \ll i_2)$$
e) Test the distinguishing accuracy acc_2 of the ND over $(C_{2,0}, C_{2,1})$.
f) Return $acc_1 - acc_2$ as the FSS.

If $|S| \geqslant 3$, there are more feature sets to be considered. Then the estimation of the FSS is more complex. But these two cases ($|S| = 1, |S| = 2$) are sufficient to explore NDs.

Select features using FSST. If a feature $\mathcal{X}(\lambda_0, \lambda_1)$ has a high influence on the distinguishing accuracy of the ND, each feature set Ω that contains this feature will obtain a high FSS. Then it is worth performing FSST under two settings ($|S| = 1, |S| = 2$).

Let $X = C_0||C_1 \in \mathbb{F}_2^{2m}$ and $\lambda = \lambda_0||\lambda_1 \in \mathbb{F}_2^{2m}$. First, we traverse $S = \{i\}, i \in [0, 2m-1]$ and perform FSST. If $S = \{i\}$ leads to a high FSS, $\lambda[i] = 1$ will hold for the feature $\mathcal{X}(\lambda_0, \lambda_1)$. Second, we traverse each possible $S = \{i_1, i_2\}$ and perform FSST. If a high FSS is obtained, $\lambda[i_1] = 1$ and $\lambda[i_2] = 1$ will hold simultaneously for the feature $\mathcal{X}(\lambda_0, \lambda_1)$.

These clues derived from FSST can be used to select features that may have a significant influence on NDs.

4.2 Explore Neural Distinguishers against Speck32/64

Brief introduction of Speck32/64. Speck32/64 is an ARX cipher [4]. Fig. 3 shows the round function of Speck32/64.

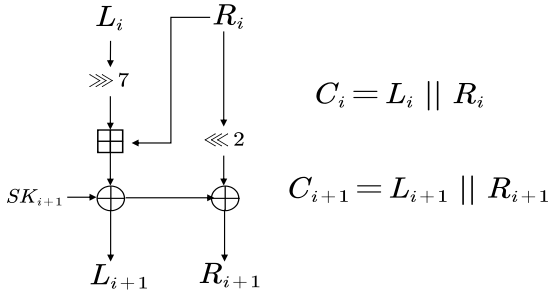

Fig. 3. The round function of Speck32/64. SK_{i+1} : the round key. $L_{i+1}||R_{i+1}$: the state of the $(i+1)$-th round.

Phenomena found by FSST. Let the plaintext difference be $\Delta = 0x40/0x0$, Gohr built NDs against Speck32/64 reduced to 5/6/7/8 rounds respectively. Table 3 summarizes the FSS estimations of the 5-round ND.

Table 3. The FSS of the 5-round ND against Speck32/64.

subset Type	**FSS**	$i/(i_1, i_2)$	number of subsets
$S = \{i\}$	$\geqslant 0.1$	$C_0 : 2 \sim 5, 10 \sim 12, 17 \sim 21, 26 \sim 28$ $C_1 : 2 \sim 5, 10 \sim 12, 17 \sim 21, 26 \sim 28$	30
	$[0.01, 0.1)$	C_0 : 9,13,14,16,24,25,29,30 C_1 : 9,13,14,16,24,25,29,30	16
	$(0, 0.01)$	others	$64 - 30 - 16 = 18$
$S = \{i_1, i_2\}$	$\geqslant 0.1$	$\{(i_1, i_2) \mid i_1 = i, i_2 = i + 16, i \in \mathcal{B}\}$, $\{(i_1, i_2) \mid i_1 = i, i_2 = i + 32, i \in \mathcal{B}\}$, $\{(i_1, i_2) \mid i_1 = i, i_2 = i + 48, i \in \mathcal{B}\}$, $\{(i_1, i_2) \mid i_1 = i + 16, i_2 = i + 32, i \in \mathcal{B}\}$, $\{(i_1, i_2) \mid i_1 = i + 16, i_2 = i + 48, i \in \mathcal{B}\}$, $\{(i_1, i_2) \mid i_1 = i + 32, i_2 = i + 48, i \in \mathcal{B}\}$, $\mathcal{B} = \{2, 3, 4, 5, 10, 11, 12\}$	42

There are some interesting phenomena shown in Table 3. First, after traversing S where $|S| = 1$, we find that if $S = \{i\}, i \in [0, 31]$ (the i-th bit of C_1) leads to a high FSS, $S = \{i + 32\}$ (the i-th bit of C_0) also leads to a high FSS. Second, if $S = \{i\}, i \in [0, 15]$ or $i \in [32, 47]$ leads to a high FSS, $S = \{i + 16\}$ also leads to a high FSS.

Based on the FSS estimations above, we further perform FSST by setting $|S| = 2$. Let $\mathcal{B} = \{2 \sim 5, 10 \sim 12\}$. For each $i \in \mathcal{B}$, we find any $S = \{i_1, i_2\}, i_1, i_2 \in \{i, i + 16, i + 32, i + 48\}$ can lead to a *similar* high FSS.

These three phenomena can be summarized as

Phenomenon 1 *Let* $X = C_0||C_1$. *If* $S = \{i\}, i \in [0, 15]$ *leads to a high FSS,* $S = \{j\}, j \in \mathcal{B}_1$ *and* $S = \{i_1, i_2\}, i_1, i_2 \in \mathcal{B}_2$ *can also lead to a* similar *high FSS where* $\mathcal{B}_1 = \{i + 16, i + 32, i + 48\}, \mathcal{B}_2 = \{i, i + 16, i + 32, i + 48\}$.

Relation with the round function. Phenomenon 1 is related to the round function of Speck32/64 (Fig.3).

Let $X = C_0||C_1$. It is clear that $X[i] = C_1[i], X[i + 32] = C_0[i], i \in [0, 31]$. The similarity between $C_1[i]$ and $C_0[i]$ is that they are both related to $SK[i]$ where $SK[i]$ is the i-th bit of the last round key. Besides, Fig.3 tells another two truths. First, both $C_0[i]$ and $C_0[i + 16], i \in [0, 15]$ are related to $SK[i]$. Second, both $C_1[i]$ and $C_1[i + 16], i \in [0, 15]$ are related to $SK[i]$.

In other words, there is a fact resulted from the round function of Speck32/64.

Fact 1 *Let* $X = C_0||C_1$. *Four ciphertext bits* $X[i]$, $X[i+16]$, $X[i+32]$, $X[i+48]$ *are related to the same bit* $SK[i], i \in [0, 15]$ *of the last round key.*

Fact 1 results in **Phenomenon 1**. To further confirm this causality, we can perform FSST on more NDs. If this causality holds, **Phenomenon 1** will always occur.

After performing experiments on the 6/7/8 round ND, we find **Phenomenon 1** always occurs. The only change is that the number of bit index subsets ($S, |S| = 1$) leading to a high FSS decreases. Taking the 6-round ND as an example, Table 4 shows the FSS estimations.

Table 4. The FSS of the 6-round ND against Speck32/64.

subset Type	**FSS**	$i/(i_1, i_2)$	number of subsets
$S = \{i\}$	$\geqslant 0.1$	$C_0 : 3, 4, 5, 11, 12, 19 \sim 21, 26 \sim 28$ $C_1 : 3, 4, 5, 11, 12, 19 \sim 21, 26 \sim 28$	22
	$[0.01, 0.1)$	$C_0 : 2, 10, 13, 14, 16 \sim 18, 25, 29, 30$ $C_1 : 2, 10, 13, 14, 16 \sim 18, 25, 29, 30$	22
	$(0, 0.01)$	others	$64 - 20 - 20 = 24$
$S = \{i_1, i_2\}$	$\geqslant 0.1$	$\{(i_1, i_2)\|i_1 = i, i_2 = i + 16, i \in \mathcal{B}\}$, $\{(i_1, i_2)\|i_1 = i, i_2 = i + 32, i \in \mathcal{B}\}$, $\{(i_1, i_2)\|i_1 = i, i_2 = i + 48, i \in \mathcal{B}\}$, $\{(i_1, i_2)\|i_1 = i + 16, i_2 = i + 32, i \in \mathcal{B}\}$, $\{(i_1, i_2)\|i_1 = i + 16, i_2 = i + 48, i \in \mathcal{B}\}$, $\{(i_1, i_2)\|i_1 = i + 32, i_2 = i + 48, i \in \mathcal{B}\}$, $\mathcal{B} = \{3, 4, 5, 11, 12\}$	30

These experiments prove the causality between **Fact 1** and **Phenomenon 1**. This causality further proves that NDs proposed by Gohr successfully capture the characteristic related to the round function of Speck32/64.

Surrogate models. Although the class of features $\mathcal{X}(\lambda_0, \lambda_1)$ is not directly related to the prediction, it's still possible to build a good surrogate model when there are sufficient features that have a significant influence on NDs.

Based on the FSST results, many clues about features $\mathcal{X}(\lambda_0, \lambda_1)$ having a significant influence on NDs can be obtained. Taking Table 3 as an example, we obtain that:

1. Because $S = \{i\}, i \notin \mathcal{B}_1$ where

$$\mathcal{B}_1 = \{3 \sim 5, 11, 12, 19 \sim 21, 26 \sim 28, 35 \sim 37, 43, 44, 51 \sim 53\}$$

   does not lead to a high FSS, the features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda = \lambda_0||\lambda_1, \lambda[i] = 1, i \notin \mathcal{B}_1$ can be discarded.
2. Because $S = \{i_1, i_2\}, i_1, i_2 \in \{i, i+16, i+32, i+48\}$ can lead to a similar high FSS when $i \in \mathcal{B}_2 = \{2, 3, 4, 5, 10, 11, 12\}$, features $\mathcal{X}(\lambda_0, \lambda_1)$ that satisfy at leaset one following condition can be saved

$$\lambda = \lambda_0||\lambda_1, \lambda[i] = \lambda[i+16] = \lambda[i+32] = \lambda[i+48] = 1, i \in \mathcal{B}_2.$$

Thus, we choose the following 128 features $\mathcal{X}(\lambda_0, \lambda_1)$ to build the surrogate model of the 5-round ND

$$\begin{aligned}\mathcal{X}^1 = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = \lambda_0[i+16], i \in [0, \cdots, 15]; \\ \lambda_0[i] = 0, i \notin \{2, 3, 4, 5, 10, 11, 12\}\}.\end{aligned} \tag{15}$$

Using the same reasoning method, we choose the following 32 features $\mathcal{X}(\lambda_0, \lambda_1)$ to build the surrogate model of the 6-round ND

$$\begin{aligned}\mathcal{X}^2 = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = \lambda_0[i+16], i \in [0, \cdots, 15]; \\ \lambda_0[i] = 0, i \notin \{3, 4, 5, 11, 12\}\}.\end{aligned} \tag{16}$$

By comparison, the folloing two feature sets are also considered

$$\begin{aligned}\mathcal{X}^3 = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = \lambda_0[i+16], i \in [0, \cdots, 15]; \\ \lambda_0[i] = 0, i \notin \{2, 3, 4, 5, 10, 11, 12, 9\}\},\end{aligned} \tag{17}$$

$$\begin{aligned}\mathcal{X}^4 = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = \lambda_0[i+16], i \in [0, \cdots, 15]; \\ \lambda_0[i] = 0, i \notin \{3, 4, 5, 11, 12, 1, 2, 10\}\}.\end{aligned} \tag{18}$$

By adopting LR as the model $\mathcal{M}$, we build four surrogate models. Table 5 summarizes the quantitative studies for the prediction between surrogate models and NDs.

Table 5. A comparison between surrogate models and NDs against Speck32/64.

Nr	Model	Number of features	Accuracy	Prediction Matching Ratio
5	ND_5	-	0.926	-
	LR + $\mathcal{X}^1$	128	0.845	0.881
	LR + $\mathcal{X}^3$	256	0.847	0.885
6	ND_6	-	0.783	-
	LR + $\mathcal{X}^2$	32	0.634	0.678
	LR + $\mathcal{X}^4$	256	0.667	0.725

It is surprising that surrogate models (LR +$\mathcal{X}^1$, LR +$\mathcal{X}^3$) achieve a prediction matching ratio higher than 0.88. After all, this class of features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT is not directly related to the final decision-making of NDs. It is difficult to build a good surrogate model by directly adopting these

features $\mathcal{X}(\lambda_0, \lambda_1)$ for a decision-making process. Thus, the results shown in Table 5 can further prove that features $\mathcal{X}(\lambda_0, \lambda_1)$ are learned by NDs.

With the number of encryption rounds increasing, the number of features $\mathcal{X}(\lambda_0, \lambda_1)$ having a significant influence decreases. In other words, all the features corresponding to EDLCT tend to make an equal contribution. This is why the fourth surrogate model (LR $+\mathcal{X}^4$) can achieve a significant improvement over the third surrogate model (LR $+\mathcal{X}^4$).

If the target is to build a surrogate model with a higher prediction matching ratio, it's better to design high-level features approximating the features adopted by NDs for the final decision-making. This target has been achieved by Benamira et al [6]. They built surrogate models that can achieve a prediction matching ratio higher than 0.9.

4.3 Explore Neural Distinguishers against DES

In *Conjecture 1* (Section 3.2), we mention that the ND proposed by Gohr is generic. Besides, the unique nature of the cipher can be captured by the ND. To challenge *Conjecture 1*, we perform experiments on DES.

Phenomena found by FSST. Let $\Delta = 0x200008/0x400$ be the plaintext difference and $N = 10^7$ (the number of training samples). We successfully build NDs against DES reduced to 5 and 6 rounds respectively. Table 6 summarizes the FSS estimations of the 5-round ND.

Table 6. The FSS of the 5-round ND against DES. {35,45,53,60}: the bit index set corresponding to $S6$ in the 4-th round. {38,44,54,63}: the bit index set corresponding to $S4$ in the 4-th round.

subset Type	**FSS**	$i/(i_1, i_2)$	number of subsets
$S = \{i\}$	$\geqslant 0.01$	C_0 : 35,36,38,44,46,53,54,60,63 C_1 : 35,36,38,44,46,53,54,60,63	18
	$[0.005, 0.01)$	C_0 : 32,39,42,45,51,52,61,62 C_1 : 32,39,42,45,51,52,61,62	16
	< 0.005	others	$128 - 18 - 16 = 94$
$S = \{i_1, i_2\}$	$\geqslant 0.01$	$\{(i_1, i_2) \mid i_1, i_2 \in \mathcal{B}$, $\mathcal{B} = \{38, 44, 54, 63, 102, 108, 118, 127\}\}$	28
	$\geqslant 0.005$	$\{(i_1, i_2) \mid i_1, i_2 \in \mathcal{B}$, $\mathcal{B} \in \{35, 45, 53, 60, 99, 109, 117, 124\}\}$	28

There are some interesting phenomena as shown in Table 6. First, if $S = \{i\}, i \in [0, 63]$ (the i-th bit of C_1) leads to a high FSS, then $S = \{i + 64\}$ (the i-th bit of C_0) also leads to a high FSS. Second, $S = \{i\}, i \in \mathcal{B}$ where $\mathcal{B}$ contains the bit indexes related to the same S-box are likely to lead to a *unique* high FSS simultaneously.

Based on the FSS estimations under the setting $|S| = 1$, we perform the FSST for $|S| = 2$. Let $\mathcal{B}$ contain the bit indexes related to the same S-box such as $S4/S6$ in the 4-th round. We find that all the $S = \{i_1, i_2\}, i_1, i_2 \in \mathcal{B}$ can lead to a *unique* high FSS.

These three phenomena can be summarized as

Phenomenon 2 *Let j_1, j_2, j_3, j_4 be the 4 bit indexes corresponding to the output of an S-box. All the $S = \{i\}, i \in \mathcal{B}$ and $S = \{i_1, i_2\}, i_1, i_2 \in \mathcal{B}$ where $\mathcal{B} = \{j_1, j_2, j_3, j_4, j_1 + 64, j_2 + 64, j_3 + 64, j_4 + 64\}$ may lead to a unique high FSS simultaneously.*

Relation with the round function. Phenomenon 2 is also related to the round function of DES.

Let $X = C_0||C_1$. $X[i] = C_1[i], X[i + 64] = C_0[i], i \in [0, 63]$ are related to the same round key bits. Besides, the input of each S-box is related to 6 round key bits, and the output of each S-box is controlled by the input.

Thus, there is a fact resulted from the round function of DES.

Fact 2 *Let $X = C_0||C_1$. Eight ciphertext bits $X[j_1]$, $X[j_2]$, $X[j_3]$, $X[j_4]$, $X[j_1 + 64]$, $X[j_2 + 64]$, $X[j_3 + 64]$, $X[j_4 + 64]$ where j_1, j_2, j_3, j_4 correspond to the same S-box are related to the same 6 bits of a round key.*

Similarly, in order to prove the causality between **Fact 2** and **Phenomenon 2**, we perform the FSST on the 6-round ND. Table 7 summarizes the estimation results of FSS.

Table 7. The FSS of the 6-round ND against DES. {34,40,48,58}: the output of $S3$ in the 5-th round. {33,41,47,55}: the output of $S1$ in the 5-th round.

subset Type	**FSS**	$i/(i_1, i_2)$	number of subsets
$S = \{i\}$	$\geqslant 0.01$	C_0 : 33,40,41,47,48,55 C_1 : 33,40,41,47,48,55	12
	$[0.005, 0.01)$	C_0 : 34,37,39,43,49,56,59,61 C_1 : 34,37,39,43,49,56,59,61	16
	$(0, 0.005)$	others	$128 - 12 - 16 = 100$
$S = \{i_1, i_2\}$	$\geqslant 0.01$	$\{(i_1, i_2)\|i_1, i_2 \in \mathcal{B}\}$, $\mathcal{B} = \{33, 41, 47, 55, 97, 105, 111, 119\}$	28
	$\geqslant 0.005$	$\{(i_1, i_2)\|i_1, i_2 \in \mathcal{B}\}$, $\mathcal{B} = \{34, 40, 48, 58, 98, 104, 114, 122\}$	28

Phenomenon 2 also occurs. This can prove the causality between **Fact 2** and **Phenomenon 2**. Besides, this causality proves that NDs also successfully capture the characteristic related to the round function of DES.

Surrogate models. Based on our analysis above, it suffices for us to focus on the ciphertext bits related to the same S-box.

To obatin the surrogate model of the 5-round ND, we choose the following 32 features $\mathcal{X}(\lambda_0, \lambda_1)$

$$\begin{aligned}\mathcal{X}^5 = &\{\mathcal{X}(\lambda_0, \lambda_1)|\lambda_0 = \lambda_1; \lambda_0[i] = 0, i \notin \{38, 44, 54, 63\}\} \\ &+\{\mathcal{X}(\lambda_0, \lambda_1)|\lambda_0 = \lambda_1; \lambda_0[i] = 0, i \notin \{35, 45, 53, 60\}\}.\end{aligned} \tag{19}$$

To obatin the surrogate model of the 6-round ND, we choose the following 32 features $\mathcal{X}(\lambda_0, \lambda_1)$

$$\begin{aligned}\mathcal{X}^6 = \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = 0, i \notin \{33, 41, 47, 55\}\} \\ +\{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1; \lambda_0[i] = 0, i \notin \{34, 40, 48, 58\}\}.\end{aligned} \tag{20}$$

By adopting LR as the model $\mathcal{M}$, we build two surrogate models. Table 8 summarizes the quantitative studies for the prediction between surrogate models and NDs.

Table 8. A comparison between surrogate models and NDs against DES.

Nr	Model	Number of features	Accuracy	Prediction Matching Ratio
5	ND_5	-	0.627	0.958
	LR + $\mathcal{X}^5$	32	0.623	
6	ND_6	-	0.551	0.815
	LR + $\mathcal{X}^6$	32	0.543	

Compared with the 5-round ND against DES, the surrogate model yields a ratio of 0.958 identical predictions. When **Nr** increases to 6, the prediction matching ratio reduces to 0.815. These results prove that features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT are learned by NDs. When **Nr** increases, these features tend to have an equal influence on the ND.

5 Explain Phenomena Related to Neural Distinguishers

Researchers have found some other phenomena related to NDs. In fact, all these phenomena are related to the EDLCT.

5.1 The Phenomenon Found at CRYPTO'19 [10]

Based on the *real difference experiment* [10], Gohr found the following phenomenon

Phenomenon 3 *Consider a ND built on a plaintext difference $\Delta \in \mathbb{F}_2^m$. Let $K \in \mathbb{F}_2^m$ be the random mask. The ND can distinguish real ciphertext pairs (C_0, C_1) from masked real ciphertext pairs $(C_0 \oplus K, C_1 \oplus K)$ without retraining.*

Table 9 shows the results of Gohr's *real difference experiment*. Since $C_0 \oplus C_1 = (C_0 \oplus K) \oplus (C_1 \oplus K)$, masked real ciphertext pairs have the same difference distribution as real ciphertext pairs. Based on **Phenomenon 3**, Gohr concluded NDs could capture some unknown features except for the ciphertext difference.

From the EDLCT perspective, **Phenomenon 3** can be explained. Real ciphertext pairs (C_0, C_1) provide two kinds of features corresponding to the EDLCT

$$\{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1\}, \quad \{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 \neq \lambda_1\}.$$

Table 9. Accuracies of NDs against Speck32/64 in the *real difference experiment* [10].

Nr	Distinguisher	Accuracy
5	ND_5	0.707
6	ND_6	0.606
7	ND_7	0.551
8	ND_8	0.507

Masked real ciphertext pairs $(C_0 \oplus K, C_1 \oplus K)$ only provide one kind of features

$$\{\mathcal{X}(\lambda_0, \lambda_1) | \lambda_0 = \lambda_1\}.$$

As long as NDs exploit features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda_0 \neq \lambda_1$, **Phenomenon 3** will occur. We prove this explanation with the following *experiment*:

1. Denote the accuracy of a ND as acc_1 and let $X = C_0||C_1, X \in \mathbb{F}_2^{2m}$.
2. Randomize $X[i] = C_1[i], X[i+m] = C_0[i], i \in [0, m-1]$ by letting

$$X \leftarrow X \oplus (\eta \ll i) \oplus (\eta \ll (i+m)) \tag{21}$$

   where η is a random binary mask.
3. Denote the accuracy of the ND over masked ciphertext pairs as acc_2.
4. Return $acc_1 - acc_2$ as the accuracy change.

If the condition $acc_1 - acc_2 \neq 0$ holds for a certain bit index $i \in [0, m-1]$, the ND must exploit some features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda_0[i] \neq \lambda_1[i]$. Table 10 shows partial testing results of NDs against Speck32/64.

Table 10. Partial experiment results of Gohr's NDs against Speck32/64. Only three decimals are kept.

Bit index	Accuracy change			
	ND_8	ND_7	ND_6	ND_5
18	0	0.001	0.018	0.071
19	0.001	0.003	0.023	0.119
20	0.002	0.004	0.016	0.028

The results in Table 10 prove our explanation for **Phenomenon 3** is correct. NDs against Speck32/64 do exploit some features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda_0 \neq \lambda_1$.

5.2 The Phenomena Found at EUROCRYPTO'20 [6]

When analyzing ND_5, ND_6 against Speck32/64, Benamira et al [6] found some interesting phenomena that can be summarized as

Phenomenon 4 *NDs may learn short but strong differential, linear or differential-linear characteristics. Consider a ND against h-round Speck32/64. The ND relies strongly on the differences at round* $h-1$ *and even strongly at round* $h-2$ *sometimes.*

Phenomenon 4 can be explained via EDLCT. First, each EDLCT entry is indeed a type of differential-linear characteristic. Second, we divide E into $E_2 \circ E_1$. Suppose E_1 covers $h-1$ rounds and E_2 covers 1 round. Then we get the characteristic $\Delta \xrightarrow{E_1} \Delta_{h-1} \xrightarrow{E_2} (\lambda_0, \lambda_1)$ where Δ_{h-1} is the difference at round $h-1$.

As we have presented, if $\Delta \xrightarrow{E} (\lambda_0, \lambda_1)$ is satisfied with a high correlation $\boldsymbol{Cor}$, the feature $\mathcal{X}(\lambda_0, \lambda_1)$ has a significant influence on NDs. Let p denote the probability $Pr(\Delta \xrightarrow{E_1} \Delta_{h-1})$. Besides, we assume $\Delta_{h-1} \xrightarrow{E_2} (\lambda_0, \lambda_1)$ is satisfied with the correlation r. Then $\boldsymbol{Cor} \approx p \times r$ holds according to the work in [5].

Because E_2 covers 1 round, r is very likely to be close to 1. Then $\boldsymbol{Cor} \approx p$ holds and the ND will rely strongly on Δ_{h-1}. Similarly, if some characteristics $\Delta_{h-2} \rightarrow (\lambda_0, \lambda_1)$ are satisfied with a very high correlation, the ND will rely on Δ_{h-2} too.

In fact, Benamira et al did an interesting experiment that can prove the features $\mathcal{X}(\lambda_0, \lambda_1)$ are learned by NDs. They extracted the output of the first convolution layer of Gohr's NDs. They found that the output is equivalent to twenty-four non-linear Boolean expressions on the input X [6]. Many Boolean expressions are only related to partial ciphertext bits.

Denote the 2D input of the first convolution layer as $[C_{0l}, C_{0r}, C_{1l}, C_{1r}]$. Taking one Boolean expression $Z = C_{0l} \wedge \overline{C_{1l}} \wedge C_{1r} \in \mathbb{F}_2^{16}$ (presented in Table 14 in [6]) as an example, we know that $Z[0]$ is related to $C_{0l}[0], C_{1l}[0], C_{1r}[0]$. Let $X = C_{0l}||C_{0r}||C_{1l}||C_{1r}$, then the three bits actually corrspond to three features $\mathcal{X}(0x10000, 0x0)$, $\mathcal{X}(0x0, 0x10000)$, and $\mathcal{X}(0x0, 0x1)$ respectively.

This experiment presented by Benamira et al proves our explanation about how NDs work. But the performance of NDs is surprising since the first convolution layer simultaneously achieves two targets: select some features $\mathcal{X}(\lambda_0, \lambda_1)$, generate new features.

Property 2 (Section 3.1) shows non-trivial features are also useful, and NDs indeed exploit this property. Thus, we think that the guess of Benamira et al is right. Neural networks indeed find the easiest way to achieve the best accuracy.

5.3 The Phenomenon Found by Chen et al [8]

Chen et al [8] found another phenomenon related to NDs. It can be described as

Phenomenon 5 *Consider a ND taking the ciphertext pair $X = C_0||C_1$ as the input. If the input X is transformed into $(C_0 \oplus K)||C_1$ or $C_0||(C_1 \oplus K)$ where $K \in \mathbb{F}_2^m$ is a random mask. ND can't distinguish masked real ciphertext pairs from masked random ciphertext pairs anymore.*

We perform experiments on NDs against two ciphers (Speck32/64, DES) again. Results prove that **Phenomenon 5** always occurs. Actually this phenomenon is normal.

Taking this operation $X \leftarrow (C_0 \oplus K)||C_1$ as an example, we know the features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda_0 \neq 0$ are all randomized. Furthermore, the features $\mathcal{X}(\lambda_0, \lambda_1)$ where $\lambda_0 = 0$ are trivial. Thus, all the features $\mathcal{X}(\lambda_0, \lambda_1)$ obey the uniform

distribution. In other words, the masked pair $(C_0 \oplus K, C_1)$ is equivalent to a random ciphertext pair now. As a result, **Phenomenon 5** occurs.

All the phenomena found by previous researchers can be well explained from the EDLCT view. This further strongly proves that the features corresponding to the EDLCT are learned by NDs.

6 Improved Differential-Linear Propagation Search

Our work and **Conclusion 1** provide a method to improve the differential-linear attack.

6.1 Differential-Linear Attack

In the latest scheme of differential-linear attack, the whole cipher E is divided into three parts E_1, E_2, E_3 as shown in Fig.4. DLCT covers the middle part E_2.

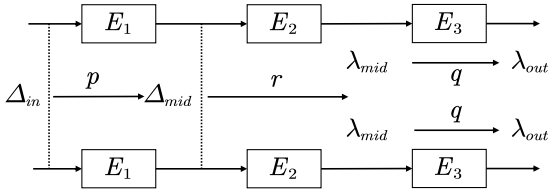

Fig. 4. The scheme of differential-linear attack. DLCT covers the middle part E_2.

Assume that the differential-linear propagation $\Delta_{mid} \xrightarrow{E_2} \lambda_{mid}$ is satisfied with the correlation

$$\boldsymbol{Cor}_{x \in S}[(\lambda_{mid} \cdot x) \oplus (\lambda_{mid} \oplus E_2(x \oplus \Delta_{mid}))] = r. \tag{22}$$

This correlation r is usually experimentally evaluated.

In particular, we assume that the differential $\Delta_{in} \xrightarrow{E_1} \Delta_{mid}$ holds with probability

$$\boldsymbol{Pr}_{x \in \mathbb{F}_2^m}[E_1(x) \oplus E_1(x \oplus \Delta_{in}) = \Delta_{mid}] = p. \tag{23}$$

We further assume that the linear approximation $\lambda_{mid} \xrightarrow{E_3} \lambda_{out}$ is satisfied with the correlation

$$\boldsymbol{Cor}_{x \in \mathbb{F}_2^m}[(\lambda_{mid} \cdot x) \oplus (\lambda_{out} \cdot E_3(x))] = q. \tag{24}$$

Then the total correlation of $\Delta_{in} \xrightarrow{E} \lambda_{out}$ can be estimated as prq^2 [5].

Since the EDLCT can describe E more completely, the scheme shown in Fig.4 is a special case if we replace the DLCT with the EDLCT. However, compared with the middle part E_2, E_3 has a more significant impact on the total correlation. Thus, it may still be a better choice by covering E_2 with the DLCT.

6.2 Search Differential-Linear Propagations via Machine Learning

In the differential-linear attack, both the optimal difference propagation $\Delta_{in} \xrightarrow{E_1} \Delta_{mid}$ and the optimal linear approximation $\lambda_{mid} \xrightarrow{E_3} \lambda_{out}$ can be obtained by some automated search techniques. However, the searching of differential-linear propagations $\Delta_{mid} \xrightarrow{E_2} \lambda_{mid}$ still needs to be solved.

For convenience, we denote a differential-linear propagation as a simple form $\Delta \xrightarrow{E} \lambda$. We propose a machine learning-based method for searching differential-linear propagations with a high correlation.

We first build a ND on Δ. If a feature λ, λ has a significant influence on the ND, we know that $\Delta \xrightarrow{E} \lambda$ is satisfied with a high correlation, which is derived from **Conclusion 1**.

Thus, searching differential-linear propagations with a high correlation is equivalent to identifying features $\mathcal{X}(\lambda, \lambda)$ that have a significant influence on NDs. This generic method is summarized in Algorithm 2.

Algorithm 2 Search differential-linear propagations via machine learning

Require: the cipher, $E : \{0,1\}^m \to \{0,1\}^m$; the plaintext difference, Δ.

Ensure: Differential-linear propagations $\Delta \xrightarrow{E} \lambda$ with a high correlation.

1: Based on Δ, randomly generate N ciphertext pairs: half real ciphertext pairs, half random ciphertext pairs.
2: Train a ND over the N ciphertext pairs.
3: Perform the FSST on the ND.
4: Save all the bit index sets S that lead to a high FSS.
5: Select features $\mathcal{X}(\lambda, \lambda)$ that may lead to a high FSS.
6: Adopt LR as the model $\mathcal{M}$, and build another distinguisher using selected features.
7: For features that have a feature weight $w > 0$, save features $\mathcal{X}(\lambda, \lambda)$ that the feature weight has a high absolute value $|w|$.
8: For features that have a feature weight $w < 0$, save features $\mathcal{X}(\lambda, \lambda)$ that the feature weight has a high absolute value $|w|$.
9: Verify the correlation of $\Delta \xrightarrow{E} \lambda$ corresponding to saved features experimentally.
10: Return $\Delta \xrightarrow{E} \lambda$ having a high correlation.

There may be many features that are selected by the FSST. Steps 3,4,5 can be used to further filter features that do not have a high correlation.

6.3 Application to Chaskey

Brief introduction of Chaskey. Chaskey is a lightweight MAC algorithm whose underlying primitive is an ARX-based permutation in an Even-Mansour construction, i.e., Chaskey-EM [16]. The permutation operates on four 32-bit words and employs 12 rounds of the form as shown in Fig.5.

Denote the difference of a, b as

$$\Delta a = \Delta a_0 || \Delta a_1 || \Delta a_2 || \Delta a_3, \quad \Delta b = \Delta b_0 || \Delta b_1 || \Delta b_2 || \Delta b_3.$$

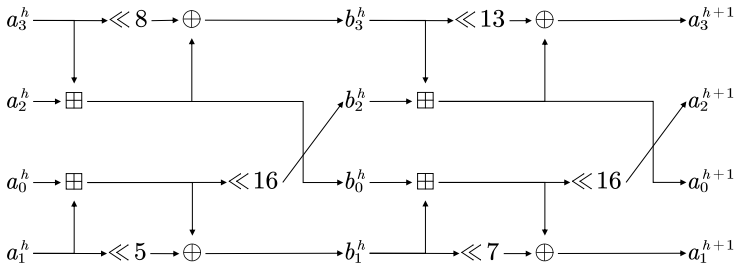

Fig. 5. The round function of Chaskey.

Similarly, the linear mask of four 32-bit words is also denoted in the same order. For convenience, we also adopt a simple form for describing an integer consisting of four 32-bit words. For example, let $X = x_0||x_1||x_2||x_3$ where $x_i \in \mathbb{F}_2^{32}, i \in [0,3]$. If the following 4 bits satisfy $x_0[j_0] = x_1[j_1] = x_2[j_2] = x_3[j_3] = 1$, then X can be described as

$$X = [j_0]|||[j_1]|||[j_2]|||[j_3], \quad j_0, j_1, j_2, j_3 \in [0, 31].$$

Differential-linear propagations observed via rules. Based on the scheme of differential-linear attack shown in Fig.4, Christof et al [5] built a 6-round distinguisher by dividing this cipher into three parts, i.e, E_1 covering 1.5 rounds, E_2 covering 4 rounds, and E_3 covering 0.5 rounds. The DLCT covers the middle part E_2.

Since it is infeasible to verify *all* possible differential-linear propagations $\Delta b^1 \xrightarrow{E_2} \lambda$, Christof et al [5] only verify the case where the input difference of Hamming weight is 1 and linear masks have the form $[i]$ or $[i, i+1]$, i.e., 1-bit or consecutive 2-bit linear masks.

As a result, when there is a non-zeros difference *only* in the 31st bit (msb) of b_2^1, i.e.

$$\Delta b^1 = []|||[]|||[31]|||[],$$

Christof et al [5] observed the following two differential-linear propagations

$$\begin{aligned} \Delta b^1 = []|||[]|||[31]|||[] &\xrightarrow{E_2} \lambda = [20]|||[]|||[]|||[], \\ \Delta b^1 = []|||[]|||[31]|||[] &\xrightarrow{E_2} \lambda = [20, 19]|||[]|||[]|||[]. \end{aligned} \tag{25}$$

The corresponding correlations are both approximately $2^{-5.1}$.

Differential-linear propagations searched via machine learning. Based on algorithm 2, we are ready to search differential-linear propagations without any restrictions on the linear masks.

We first attempted to build a 4-round ND based on $\Delta b^1 = []|||[]|||[31]|||[]$, however, it was not successful. After analyzing the round function of Chaskey, we observe the optimal differential convering 1.5 rounds

$$\Delta b^1 = []|||[]|||[31]|||[] \rightarrow \Delta a^3 = [31, 15, 7]|||[6]|||[]|||[31, 20, 15, 12, 7].$$

The transition probability is 2^{-2}.

Taking Δa^3 as the difference constraint, we successfully build a ND covering 2.5 rounds. After running algorithm 2, we first notice that the following bit index sets can lead to a FSS higher than 0.05

$$\{S = \{i, i+128\} | i \in \{28, 30, 116, 118, 123, 124, 126\}\}. \tag{26}$$

Then a distinguisher is built using the following features

$$\{\mathcal{X}(\lambda, \lambda) | \lambda[i] = 0, i \notin \{28, 30, 116, 118, 123, 124, 126\}\}. \tag{27}$$

The feature weights W^T extracted from LR show that there are 7 features having a significant influence

$$\{\mathcal{X}(\lambda, \lambda) | \lambda \in \{\lambda^1, \lambda^2, \lambda^3, \lambda^4, \lambda^5, \lambda^6, \lambda^7\}\}$$

where

$$\begin{gathered}
\lambda^1 = [20]||[]||[]||[] \\
\lambda^2 = [20, 19]||[]||[]||[] \\
\lambda^3 = [28]||[]||[]||[28] \\
\lambda^4 = [28, 20]||[]||[]||[28] \\
\lambda^5 = [27, 20]||[]||[]||[27] \\
\lambda^6 = [28, 27]||[]||[]||[28, 27] \\
\lambda^7 = [28, 27, 20]||[]||[]||[28, 27].
\end{gathered} \tag{28}$$

Finally, we experimentally verify differential-linear propagations corresponding to the 7 linear masks. Table 11 summarizes the experimental correlations.

Table 11. 4-round differential-linear propagations searched via machine learning. E_2 covers 4 rounds $b^1 \to b^5$ of Chaskey.

differential-linear propagation	Correlation
$\Delta b^1 \xrightarrow{E_2} \lambda^1$	$2^{-5.1}$
$\Delta b^1 \xrightarrow{E_2} \lambda^2$	$2^{-5.1}$
$\Delta b^1 \xrightarrow{E_2} \lambda^3$	$2^{-3.4}$
$\Delta b^1 \xrightarrow{E_2} \lambda^4$	$2^{-2.7}$
$\Delta b^1 \xrightarrow{E_2} \lambda^5$	$2^{-5.1}$
$\Delta b^1 \xrightarrow{E_2} \lambda^6$	$2^{-3.8}$
$\Delta b^1 \xrightarrow{E_2} \lambda^7$	$2^{-2.7}$

Except for the first two differential-linear propagations observed by Christof et al [5], Algorithm 2 helps us find five more differential-linear propagations.

6.4 Application to DES

Differential-linear propagations searched by traversing differentials. Based on the scheme of differential-linear attack shown in Fig.4, Bar-On [1] build

a 7-round distinguisher by dividing 7-round DES into three parts, i.e., E_1 covers 2 rounds, E_2 covers 3 rounds, and E_3 covers 2 rounds. The DLCT covers the middle part E_2.

The differential with a probability of $\frac{16}{64}$ presented in [1] is

$$\Delta_{in} = 0x200008||0x0400 \xrightarrow{E_1} \Delta_{mid} = 0x0||0x60000000.$$

The differential-linear propagation is

$$\Delta_{mid} = 0x0||0x60000000 \xrightarrow{E_2} \lambda_{mid} = 0x0||0x808202.$$

And the corresponding correlation is about 0.52. The linear approximation is

$$\lambda_{mid} = 0x0||0x808202 \xrightarrow{E_3} \lambda_{out} = 0x80000000||0x808202.$$

Remark 1. There is a minor mistake in [1]. The value of Δ_{mid} should be $0x0||0x0400$ instead of $0x0||0x60000000$. $\Delta_{in} \xrightarrow{E_1} \Delta_{mid} = 0x0||0x0400$ is satisfied with a probability of $\frac{16}{64}$. $\Delta_{mid} = 0x0||0x0400 \xrightarrow{E_3} \lambda_{out}$ is satisfied with a correlation 0.52 (the bias is 0.26).

The 3-round differential-linear propagation presented in [1] is obtained by looking at all 3-round differentials starting with the input difference $\Delta_{mid} = 0x0||0x0400$.

If we remove Δ_{mid}, we can derive the equivalent 5-round / 6-round differential-linear propagation adopted in [1]. Specifically, the 5-round differntial-linear propagation is

$$\Delta_{in} = 0x200008||0x0400 \xrightarrow{E_2 \circ E_1} \lambda_{mid} = 0x0||0x808202. \tag{29}$$

By extending 1 round, the 6-round differential linear propagation is

$$\Delta_{in} = 0x200008||0x0400 \xrightarrow{E_2^{'} \circ E_1} \lambda_{mid}^{'} = 0x808202||0x0 \tag{30}$$

where $E_2^{'}$ covers 4 rounds. The corresponding correlations are both about $0.13 = \frac{1}{4} \times 0.52$, which are also experimentally verified.

Differential-linear propagations searched via machine learning. Based on the plaintext difference $\Delta_{in} = 0x200008||0x0400$, we successfully build NDs against DES reduced to 5 and 6 rounds respectively.

Thus, we directly find 5-round or 6-round differential-linear propagations via machine learning. Furthermore, there is no need to check all the 5-round / 6-round differentials. In Section 4.3, the FSST has been performed on two NDs. Based on the results shown in Table 6 and Table 7, we continue running algorithm 2.

Finally, we identify more differential-linear propagations with a high correlation. Table 12 summarizes differential-linear propagations searched via machine learning.

Table 12. Differential-linear propagations searched via machine learning. $E_2 \circ E_1$ covers 5 rounds of DES. $E_2^{'} \circ E_1$ covers 6 rounds. $\Delta_{in} = 0x200008||0x0400$.

rounds	differential-linear propagation	Correlation
5	$\{\Delta_{in} \xrightarrow{E_2 \circ E_1} \lambda_{mid} \vert \lambda_{mid} \neq 0; \lambda_{mid}[i] = 0, i \notin \{63, 54, 44, 38\}\}$	> 0.13
5	$\{\Delta_{in} \xrightarrow{E_2 \circ E_1} \lambda_{mid} \vert \lambda_{mid} \neq 0; \lambda_{mid}[i] = 0, i \notin \{60, 53, 45, 35\}\}$	> 0.13
6	$\Delta_{in} \xrightarrow{E_2^{'} \circ E_1} \lambda_{mid}^{'} = 0x808202||0x0$	0.13

Taking 0.13 as the correlation threshold, we find 30 more 5-round differential-linear propagations that are satisfied with a correlation higher than 0.13. We further verify that only one 6-round differential-linear propagation is satisfied with a correlation of 0.13. In fact, it is the equivalent 6-round differential-linear propagation adopted in [1].

The result above proves the 7-round differential-linear distinguisher presented in [1] is the *optimal one*, which is one of our extra findings. All the above results including the extra finding fully prove that the advantage of searching differential-linear propagations via machine learning.

7 Improve the Neural Distinguisher

We prove that features corresponding to the EDLCT are learned by NDs. This finding provided an insight (*Conjecture 2*) for accelerating NDs.

Conjecture 2. As long as most features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT can be derived from the input X of NDs, the concrete form of X will not affect the distinguishing accuracy.

Optional Inputs. Actually, most features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT can be derived from many optional inputs. To challenge *Conjecture 2*, we propose a generic method for generating optional inputs.

We divide a ciphertext C into two parts $C = C_l||C_r$ with the same length. A ciphertext pair can be described as $(C_0, C_1) = (C_{0l}||C_{0r}, C_{1l}||C_{1r})$. Then there are more available types of representation that can be used as the input of NDs. Table 13 summarizes some possible forms of the input X.

Table 13. Some available forms of the input X of NDs

X^1	$(C_{0l} \oplus C_{1l})\vert\vert(C_{0r} \oplus C_{1r})\vert\vert(C_{0l} \oplus C_{0r})$
X^2	$(C_{0l} \oplus C_{1l})\vert\vert(C_{0r} \oplus C_{1r})\vert\vert(C_{1l} \oplus C_{1r})$
X^3	$(C_{0l} \oplus C_{0r})\vert\vert(C_{1l} \oplus C_{1r})\vert\vert(C_{0l} \oplus C_{1l})$
X^4	$(C_{0l} \oplus C_{0r})\vert\vert(C_{1l} \oplus C_{1r})\vert\vert(C_{0r} \oplus C_{1r})$
X^5	$(C_{0l} \oplus C_{0r})\vert\vert(C_{1l} \oplus C_{1r})\vert\vert(C_{0l} \oplus C_{1r})$
X^6	$(C_{0l} \oplus C_{0r})\vert\vert(C_{1l} \oplus C_{1r})\vert\vert(C_{0r} \oplus C_{1l})$
X^7	$(C_{0l} \oplus C_{1r})\vert\vert(C_{0r} \oplus C_{1l})\vert\vert(C_{0l} \oplus C_{0r})$
X^8	$(C_{0l} \oplus C_{1r})\vert\vert(C_{0r} \oplus C_{1l})\vert\vert(C_{1l} \oplus C_{1r})$

Taking the first possible form X^1 as an example, we prove that most features $\mathcal{X}(\lambda_0, \lambda_1)$ corresponding to the EDLCT can be derived from X^1.

Proof. *Denote* $\lambda_0^{'}, \lambda_1^{'}, \lambda_2^{'}, \lambda_3^{'}$ *as four independent linear masks. First, all the features* $\mathcal{X}(\lambda_0, \lambda_1)$ *where* $\lambda_0 = \lambda_1$ *can be obtained as follows:*

$$\begin{aligned}
&[\lambda_0^{'} \cdot (C_{0l} \oplus C_{1l})] \oplus [\lambda_1^{'} \cdot (C_{0r} \oplus C_{1r})] \\
&= [(\lambda_0^{'}||\lambda_1^{'}) \cdot (C_{0l}||C_{0r})] \oplus [(\lambda_0^{'}||\lambda_1^{'}) \cdot (C_{1l}||C_{1r})] \\
&\Rightarrow \lambda_0 = \lambda_1 = \lambda_0^{'}||\lambda_1^{'}.
\end{aligned}$$

Since $\lambda_0^{'}, \lambda_1^{'}$ *are two independent linear masks, it follows that all the features* $\mathcal{X}(\lambda_0, \lambda_1)$ *where* $\lambda_0 = \lambda_1$ *can be obtained.*

Most features $\mathcal{X}(\lambda_0, \lambda_1)$ *where* $\lambda_0 \neq \lambda_1$ *can be obtained as follows:*

$$\begin{aligned}
&(C_{0l} \oplus C_{1l}) \oplus (C_{0r} \oplus C_{1r}) \oplus (C_{0l} \oplus C_{0r}) = C_{1l} \oplus C_{1r} \\
&\Rightarrow [\lambda_0^{'} \cdot (C_{0l} \oplus C_{1l})] \oplus [\lambda_1^{'} \cdot (C_{0r} \oplus C_{1r})] \oplus [\lambda_2^{'} \cdot (C_{0l} \oplus C_{0r})] \oplus [\lambda_3^{'} \cdot (C_{1l} \oplus C_{1r})] \\
&= [(\lambda_0^{'} \oplus \lambda_2^{'})||(\lambda_1^{'} \oplus \lambda_2^{'})] \cdot C_0 \oplus [(\lambda_0^{'} \oplus \lambda_3^{'})||(\lambda_1^{'} \oplus \lambda_3^{'})] \cdot C_1 \\
&\Rightarrow \lambda_0 = (\lambda_0^{'} \oplus \lambda_2^{'})||(\lambda_1^{'} \oplus \lambda_2^{'}), \quad \lambda_1 = (\lambda_0^{'} \oplus \lambda_3^{'})||(\lambda_1^{'} \oplus \lambda_3^{'}).
\end{aligned}$$

If we first fix λ_0 *by fixing* $\lambda_0^{'}, \lambda_1^{'}$, *and* $\lambda_2^{'}$, *then* λ_0 *can be any possible value and* λ_1 *will only face a weak restriction. Generally speaking, most features* $\mathcal{X}(\lambda_0, \lambda_1)$ *corresponding to EDLCT can be obtained.*

Using a similar reasoning method, we arrive at the same conclusion when another form $X^i, i \in [2, 8]$ is considered. From the EDLCT perspective, both the available forms of the input X in Table 13 and $X = C_{0l}||C_{0r}||C_{1l}||C_{1r}$ can provide many of the same features. Thus, we reduce the input length of NDs. This can accelerate NDs and the distinguishing accuracy may not be affected.

Accelerate neural distinguishers by reducing the input length. To prove *Conjecture 2*, we build several NDs against Speck32/64 from scratch. Four optional inputs X^1, X^3, X^5, X^7 in Table 13 are adopted as examples. The deep residual network proposed by Gohr is adopted.

Table 14 summarizes the accuracies of various NDs. All the NDs with new inputs (X^1, X^3, X^5, X^7) can achieve the similar distinguishing accuracy as Gohr's NDs (NN + X).

Table 14. Accuracies of various NDs considering different inputs. NN + X stands for Gohr's NDs. $X = C_{0l}||C_{0r}||C_{1l}||C_{1r}$

	Distinguisher		NN + X	NN + X^1	NN + X^3	NN + X^5	NN + X^7
Speck32/64	**Nr**	5	0.926	0.926	0.926	0.926	0.926
		6	0.783	0.784	0.784	0.784	0.783
		7	0.609	0.61	0.609	0.61	0.61

However, the length of new inputs is three-quarters of that of X, which can reduce the computation of NDs. We have also tested the practical time consumption of various NDs. The time consumption of new NDs is reduced to about three-quarters of that of Gohr's NDs.

Results shown in Table 12 further prove that features corresponding to EDLCT are learned by Gohr's NDs.

Build nearly perfect surrogate models by changing the input. Since most features $\mathcal{X}(\lambda_0, \lambda_1)$ can be derived from optional inputs shown in Table 13, efficient surrogate models will be obtained by changing the input of NDs.

Taking X^1 as the input of the deep residual network proposed by Gohr, we build surrogate models of NDs against Speck32/64. Table 15 summarizes the quantitative studies for the prediction between surrogate models and NDs.

Table 15. A comparison between surrogate models and NDs against Speck32/64.

Nr	Model	Accuracy	Prediction Matching Ratio
5	ND_5	0.926	0.992
	NN + X^1	0.926	
6	ND_6	0.783	0.979
	NN + X^1	0.782	
7	ND_7	0.609	0.964
	NN + X^1	0.607	

As expected, all the surrogate models achieve a very high prediction matching ratio. These surrogate models further strongly prove that features corresponding to the EDLCT are learned by NDs.

8 Conclusion

In this article, we propose the Extended Differential-Linear Connectivity Table (EDLCT). Various machine learning-based distinguishers can be built via a class of features corresponding to the EDLCT. By exploring various NDs, we prove that NDs work by exploiting features corresponding to the EDLCT. All the phenomena related to NDs that are found are explained exactly via EDLCT. We further show how to search differential-linear propagations with a high correlation via machine learning. As a result, we find many better differential-linear propagations for Chaskey and DES. Besides, we propose a method to improve NDs.

Our work not only explains NDs, but also provides a new tool (EDLCT) to describe a cipher. Besides, some cryptanalysis techniques are improved from the perspective of EDLCT. We believe that our work can inspire more interesting research in the future.

References

1. Bar-On, A., Dunkelman, O., Keller, N., Weizman, A.: Dlct: a new tool for differential-linear cryptanalysis. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 313–342. Springer (2019)
2. Barlow, H.B.: Unsupervised learning. Neural computation **1**(3), 295–311 (1989)

3. Batina, L., Bhasin, S., Jap, D., Picek, S.: Poster: Recovering the input of neural networks via single shot side-channel attacks. computer and communications security pp. 2657–2659 (2019)
4. Beaulieu, R., Shors, D., Smith, J., Treatmanclark, S., Weeks, B., Wingers, L.: The simon and speck lightweight block ciphers. design automation conference p. 175 (2015)
5. Beierle, C., Leander, G., Todo, Y.: Improved differential-linear attacks with applications to arx ciphers. In: Annual International Cryptology Conference. pp. 329–358. Springer (2020)
6. Benamira, A., Gerault, D., Peyrin, T., Tan, Q.Q.: A deeper look at machine learning-based cryptanalysis. IACR Cryptol. ePrint Arch **287**, 2021 (2021)
7. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. Journal of CRYPTOLOGY **4**(1), 3–72 (1991)
8. Chen, Y., Yu, H.: Neural aided statistical attack for cryptanalysis. Cryptology ePrint Archive, Report 2020/1620 (2020), https://eprint.iacr.org/2020/1620
9. Chen, Y., Yu, H.: A new neural distinguisher model considering derived features from multiple ciphertext pairs. Cryptology ePrint Archive, Report 2021/310 (2021), https://eprint.iacr.org/2021/310
10. Gohr, A.: Improving attacks on round-reduced speck32/64 using deep learning. international cryptology conference pp. 150–179 (2019)
11. Greydanus, S.: Learning the enigma with recurrent neural networks. arXiv: Neural and Evolutionary Computing (2017)
12. Hecht-Nielsen, R.: Theory of the backpropagation neural network. In: Neural networks for perception, pp. 65–93. Elsevier (1992)
13. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of artificial intelligence research **4**, 237–285 (1996)
14. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: Make some noise: Unleashing the power of convolutional neural networks for profiled side-channel analysis. cryptographic hardware and embedded systems **2019**(3), 148–179 (2019)
15. Kleinbaum, D.G., Dietz, K., Gail, M., Klein, M., Klein, M.: Logistic regression. Springer (2002)
16. Mouha, N., Mennink, B., Van Herrewege, A., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: an efficient mac algorithm for 32-bit microcontrollers. In: International Conference on Selected Areas in Cryptography. pp. 306–323. Springer (2014)
17. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT press (2012)
18. Pavelski, L.M., Delgado, M.R., Almeida, C.P., Gonçalves, R.A., Venske, S.M.: Extreme learning surrogate models in multi-objective optimization based on decomposition. Neurocomputing **180**, 55–67 (2016)
19. Zeiler, M.D., Taylor, G.W., Fergus, R.: Adaptive deconvolutional networks for mid and high level feature learning. In: 2011 International Conference on Computer Vision. pp. 2018–2025. IEEE (2011)
20. Zhu, X., Goldberg, A.B.: Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning **3**(1), 1–130 (2009)