More Insight on Deep Learning-aided Cryptanalysis

Zhenzhen Bao   Jinyu Lu   Yiran Yao   Liu Zhang

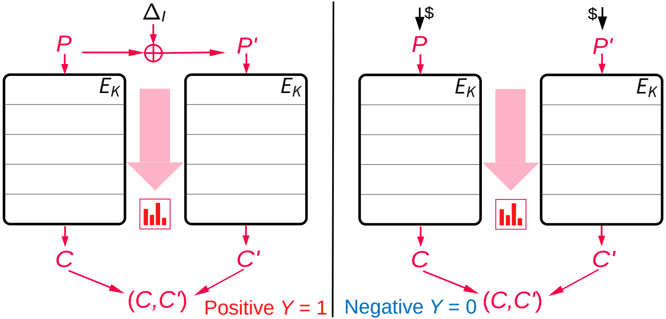
4 – 8 December 2023, Asiacrypt

Differential-based Neural Distinguishers [Goh19]

Task: distinguishing two types of ciphertext pairs

$$\text{Positive } (C, C'),\quad Y = 1,\quad \text{where } (C, C') \xleftarrow{\texttt{Enc}} \big((P, P') \mid P \leftarrow_{\$}, P' = P \oplus \Delta_I\big)$$

$$\text{Negative } (C, C'),\quad Y = 0,\quad \text{where } (C, C') \xleftarrow{\texttt{Enc}} \big((P, P') \mid P \leftarrow_{\$}, P' \leftarrow_{\$}\big)$$

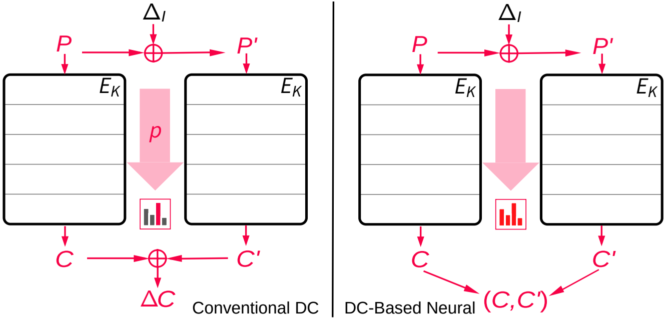


Differential-based Neural Distinguishers [Goh19]

Task: distinguishing two types of ciphertext pairs

$$\text{Positive } (C, C'),\quad Y = 1,\quad \text{where } (C, C') \xleftarrow{\texttt{Enc}} \big((P, P') \mid P \leftarrow_{\$}, P' = P \oplus \Delta_I\big)$$

$$\text{Negative } (C, C'),\quad Y = 0,\quad \text{where } (C, C') \xleftarrow{\texttt{Enc}} \big((P, P') \mid P \leftarrow_{\$}, P' \leftarrow_{\$}\big)$$



No.	w	Train X	Y
0	x	1 0 1 1 0 1 0 0 1 1 1 0 0 1 0 1	1
	y	0 1 0 1 0 0 1 1 1 1 1 0 1 0 1 1	
	x'	0 0 1 1 0 0 1 1 0 0 0 1 1 1 0 1	
	y'	0 1 0 1 1 0 0 0 1 1 0 1 0 1 0 0	
1	x	0 1 0 0 1 1 1 0 0 0 0 0 1 0 0 0	0
	y	0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0	
	x'	1 0 0 1 1 1 1 0 0 1 1 1 0 1 0 0	
	y'	0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0	
2	x	1 0 0 0 1 0 0 0 0 1 1 0 1 0 0 0	1
	y	0 1 1 1 1 1 0 1 1 1 1 0 1 1 1 0	
	x'	0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1	
	y'	0 0 0 1 1 0 1 0 0 0 0 0 1 1 0 0	
3	x	1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 1	0
	y	1 1 0 0 1 0 0 1 1 1 1 0 1 1 1 1	
	x'	1 0 1 0 1 1 1 0 0 1 0 1 0 1 0 0	
	y'	1 1 0 1 1 0 0 1 1 0 0 0 1 0 1 0	
4	x	0 0 1 1 1 0 0 0 1 1 0 0 1 0 1 0	1
	y	0 1 1 0 0 0 0 1 1 0 1 1 0 0 0 1	
	x'	1 1 0 1 1 0 1 0 0 0 1 1 0 1 0 1	
	y'	1 1 0 0 1 0 1 1 1 1 0 1 0 1 1 1	
5	x	0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0	1
	y	0 1 1 0 1 0 0 1 0 1 0 1 1 1 1 0	
	x'	1 0 1 1 1 0 0 1 1 0 0 0 0 0 0 1	
	y'	1 1 0 1 1 0 1 0 0 0 0 1 0 0 1 0	
6	x	1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0	0
	y	1 0 1 1 1 1 0 1 1 0 1 1 1 1 0 1	
	x'	0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 1	
	y'	0 0 0 0 1 1 1 0 1 1 0 1 0 0 0 0	
7	x	1 1 1 1 0 1 1 1 1 1 1 0 0 1 1 0	0
	y	0 1 1 0 1 0 1 0 0 1 1 1 1 0 1 0	
	x'	1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0	
	y'	1 0 0 0 0 1 1 0 0 1 0 1 0 0 0 0	

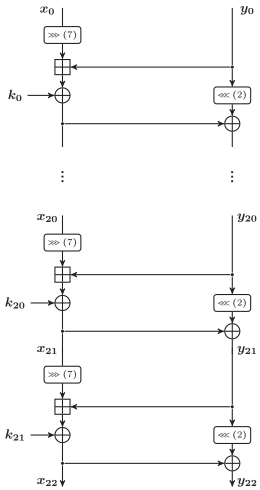
Algorithm 1: Encryption of Speck32/64

Input: $P := (x_0, y_0)$, $\{k_0, \cdots, k_{21}\}$
Output: $C = (x_{22}, y_{22})$
for $r = 0$ *to* 21 **do**

$$x_{r+1} \leftarrow x_r^{\ggg 7} \boxplus y_r \oplus k_r$$
$$y_{r+1} \leftarrow y_r^{\lll 2} \oplus x_{r+1}$$

end

Feistel-like cipher: Speck32/64



Evaluation [Goh19]

No.	w	Verification X																Z	Y	
0	x	0	1	0	0	1	1	1	0	1	1	0	0	0	0	1	1	0.35	0	TN
	y	0	0	1	1	1	1	0	1	1	0	0	1	1	1	1	0			
	x'	1	1	0	0	1	0	0	1	0	0	1	1	1	1	0	0			
	y'	0	0	1	0	1	0	1	0	0	0	0	0	1	1	0	0			
1	x	0	0	0	1	1	1	1	1	1	0	0	1	1	0	1	0	0.67	0	FP
	y	1	1	1	0	0	0	1	1	0	1	1	1	1	1	1	0			
	x'	1	0	1	1	0	0	1	1	1	1	0	1	1	0	1	1			
	y'	0	1	1	1	1	0	0	0	1	1	0	0	0	1	1	1			
2	x	0	1	1	1	0	1	1	1	1	1	1	0	0	0	1	0	0.74	1	TP
	y	1	1	0	0	1	0	0	0	1	1	0	1	1	0	0	1			
	x'	1	0	1	0	0	1	0	0	0	1	1	1	1	1	1	0			
	y'	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1			
3	x	0	0	1	0	0	1	1	0	0	0	1	1	1	1	0	1	0.63	0	FP
	y	1	1	0	0	0	1	0	1	0	1	0	0	0	1	1	1			
	x'	0	1	0	0	1	0	0	0	0	0	1	0	0	1	1	1			
	y'	1	1	0	1	0	0	0	1	0	0	0	0	1	0	1	1			
4	x	0	1	1	1	1	1	0	0	0	0	0	1	0	1	0	0	0.46	1	FN
	y	1	1	1	0	1	0	0	0	1	1	1	0	0	0	1	0			
	x'	0	0	0	0	0	1	1	0	0	0	1	1	0	1	1	1			
	y'	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0			
5	x	0	1	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0.42	0	TN
	y	0	0	1	1	1	1	1	0	1	0	0	0	1	1	1	0			
	x'	1	1	1	0	1	0	1	0	1	0	1	1	1	1	1	1			
	y'	1	0	0	0	0	1	1	0	0	0	1	0	1	0	0	1			
6	x	0	1	0	0	1	1	1	1	1	0	0	0	1	0	0	0	0.66	1	TP
	y	0	0	1	0	1	1	1	0	1	1	0	0	1	1	0	0			
	x'	1	0	0	0	0	1	1	0	0	1	1	1	1	0	1	1			
	y'	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	1			
7	x	1	1	1	0	0	0	1	1	1	1	1	0	0	0	0	1	0.77	1	TP
	y	1	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0			
	x'	0	0	1	1	0	1	0	0	0	0	1	0	0	0	0	0			

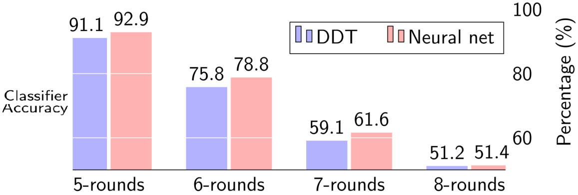
```python
import numpy as np

def evaluate_tiny(net,X,Y):
    Z = net.predict(X,batch_size=10000).flatten();
    Zbin = (Z > 0.5);
    diff = Y - Z;
    mse  = np.mean(diff*diff);
    n    = len(Z);
    n0   = np.sum(Y==0);
    n1   = np.sum(Y==1);
    acc  = np.sum(Zbin == Y) / n;
    tpr  = np.sum(Zbin[Y==1]) / n1;
    tnr  = np.sum(Zbin[Y==0] == 0) / n0;
    return (acc, tpr, tnr, mse)
```

acc: 0.625, tpr: 0.75, tnr: 0.5, mse: 0.20905

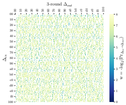
Results [Goh19]

Accuracy of Gohr's neural distinguishers on Speck32/64 [Goh19]

#R	Name	Accuracy	True Positive Rate	True Negative Rate
5	$\mathcal{DD}^{\text{Speck}5R}$	0.911	0.877	0.947
5	$\mathcal{ND}^{\text{Speck}5R}$	0.929	0.904	0.954
6	$\mathcal{DD}^{\text{Speck}6R}$	0.758	0.680	0.837
6	$\mathcal{ND}^{\text{Speck}6R}$	0.788	0.724	0.853
7	$\mathcal{DD}^{\text{Speck}7R}$	0.591	0.543	0.640
7	$\mathcal{ND}^{\text{Speck}7R}$	0.616	0.533	0.699
8	$\mathcal{DD}^{\text{Speck}8R}$	0.512	0.496	0.527
8	$\mathcal{ND}^{\text{Speck}8R}$	0.514	0.519	0.508



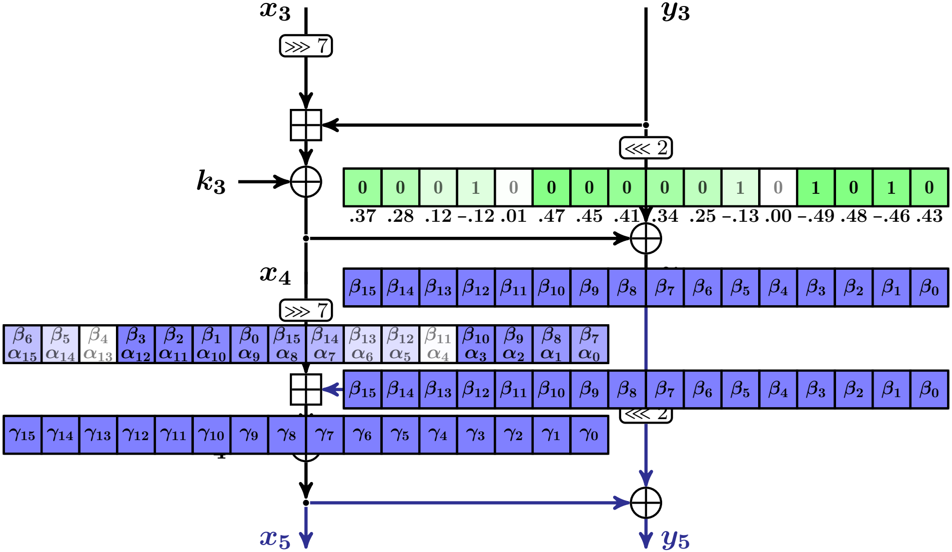
$$Z_{\mathcal{DD}} = \begin{cases} 1 & \text{if } \mathrm{DDT}[\Delta_{in}, \Delta_C] > \frac{1}{2^{32}-1} \\ 0 & \text{otherwise} \end{cases}$$



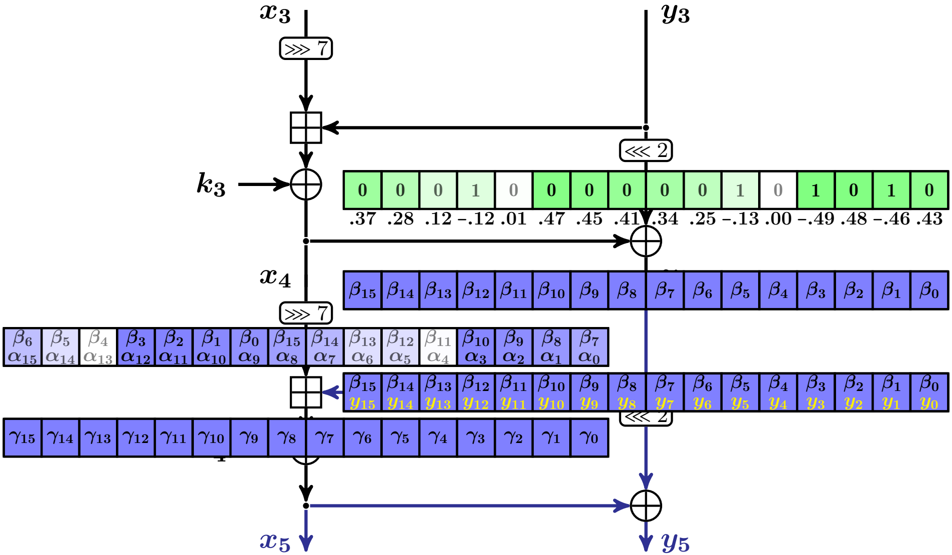
Where the extra advantages of $\mathcal{ND}$ comes from

1. Previous research suggests that these distinguishers rely on differential distributions in the penultimate and antepenultimate rounds [Ben+21].
2. The neural distinguishers can make finer distinctions than mere difference equivalence classes [Goh19].
3. What specific knowledge these neural distinguishers learn beyond DDT?

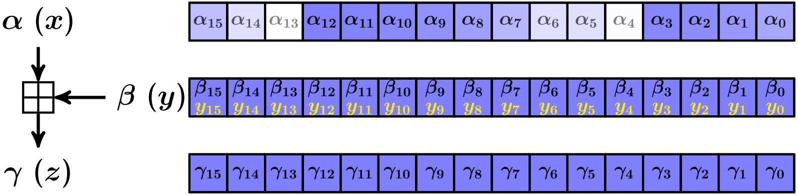
What are the Additional Features that $\mathcal{DD}$ Missed?



What are the Additional Features that $\mathcal{DD}$ Missed?



Linear Constraints for an XOR-differential Through $\boxplus$



Observation ⋆

Let $\delta = (\alpha, \beta \mapsto \gamma)$ be a possible XOR-differential through addition modulo 2^n ($\boxplus$). For (x, y) and $(x \oplus \alpha, y \oplus \beta)$ be a conforming pair of δ, x and y should satisfy the follows. For $0 \leq i < n-1$, if $\mathtt{eq}(\alpha, \beta, \gamma)_i = 0$

$$\left.\begin{array}{lll} x_i \oplus y_i = \mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \alpha_i, & & \text{if } \alpha_i \oplus \beta_i = 0, \\ \left.\begin{array}{ll} x_i \oplus c_i = \mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \alpha_i, & \text{if } \alpha_i \oplus \mathtt{xor}(\alpha,\beta,\gamma)_i = 0, \\ y_i \oplus c_i = \mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i, & \text{if } \alpha_i \oplus \mathtt{xor}(\alpha,\beta,\gamma)_i = 1, \end{array}\right\} & & \text{if } \alpha_i \oplus \beta_i = 1, \end{array}\right\}$$

where c_i is the i-th carry bit, $\mathtt{eq}(a, b, d) = 1$ if and only if $a = b = d$, $\mathtt{xor}(a, b, d) = a \oplus b \oplus d$.

Fixed-y probability of an XOR-differential Through $\boxplus$

Observation ⋆

Let $\delta = (\alpha, \beta \mapsto \gamma)$ be a possible XOR-differential through addition modulo 2^n ($\boxplus$). For (x, y) and $(x \oplus \alpha, y \oplus \beta)$ be a conforming pair of δ, x and y should satisfy the follows. For $0 \leq i < n - 1$, if $\mathtt{eq}(\alpha, \beta, \gamma)_i = 0$

$$\left.\begin{array}{ll} x_i \oplus y_i = \mathtt{xor}(\alpha, \beta, \gamma)_{i+1} \oplus \alpha_i, & \text{if } \alpha_i \oplus \beta_i = 0, \\ \left.\begin{array}{ll} x_i \oplus c_i = \mathtt{xor}(\alpha, \beta, \gamma)_{i+1} \oplus \alpha_i, & \text{if } \alpha_i \oplus \mathtt{xor}(\alpha, \beta, \gamma)_i = 0, \\ y_i \oplus c_i = \mathtt{xor}(\alpha, \beta, \gamma)_{i+1} \oplus \beta_i, & \text{if } \alpha_i \oplus \mathtt{xor}(\alpha, \beta, \gamma)_i = 1, \end{array}\right\} & \text{if } \alpha_i \oplus \beta_i = 1, \end{array}\right\}$$

where c_i is the i-th carry bit, $\mathtt{eq}(a, b, d) = 1$ if and only if $a = b = d$, $\mathtt{xor}(a, b, d) = a \oplus b \oplus d$.

- At bit positions i and $i + 1$, a difference tuple $(\alpha_{i+1,i}, \beta_{i+1,i}, \gamma_{i+1,i})$ that satisfies $\mathtt{eq}(\alpha_i, \beta_i, \gamma_i) = 0$ imposes 1-bit linear constraint on $(x_i,\ y_i)$, $(x_i,\ c_i)$, or $(y_i,\ c_i)$.
- Suppose the probability of $(\alpha, \beta \mapsto \gamma)$ is p, then the fixed-$(x_i,\ y_i,\ c_i)$ probability

$$\tilde{p} = \begin{cases} 2 \cdot p & \text{the constraint is fulfilled,} \\ 0 & \text{the constraint is not fulfilled.} \end{cases}$$

Case No.	Difference	Constraint on values	Known
$\mathrm{Cxy}_{(i+1,i)}$	$\begin{cases} \mathtt{eq}(\alpha,\beta,\gamma)_i = 0, \\ \alpha_i \oplus \beta_i = 0. \end{cases}$	$\mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \alpha_i = x_i \oplus y_i$	None
$\mathrm{Cxc}_{(i+1,i)}$	$\begin{cases} \mathtt{eq}(\alpha,\beta,\gamma)_i = 0, \\ \alpha_i \oplus \beta_i = 1, \\ \alpha_i \oplus \mathtt{xor}(\alpha,\beta,\gamma)_i = 0. \end{cases}$	$\mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \alpha_i = x_i \oplus c_i$	None
$\mathrm{Cyc}_{(i+1,i)}$	$\begin{cases} \mathtt{eq}(\alpha,\beta,\gamma)_i = 0, \\ \alpha_i \oplus \beta_i = 1, \\ \alpha_i \oplus \mathtt{xor}(\alpha,\beta,\gamma)_i = 1. \end{cases}$	$\mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i = y_i \oplus c_i$	$y_i \oplus c_i$

"Known" indicate whether the fulfillment of the condition might be known in SPECK32/64's last $\boxplus$.

- In SPECK32/64, one can only know the value of y among the tuple (x, y, c) for the last $\boxplus$ $\Rightarrow$
- One needs to consider bit positions corresponds to the third case named $\mathrm{Cyc}_{(i+1,i)}$.

Fixed-y probability

- In case $\mathrm{Cyc}_{(i+1,i)}$, the constraint is on $y_i \oplus c_i$. The value of c_i might be unknown, but

$$c_i = x_{i-1}y_{i-1} \oplus (x_{i-1} \oplus y_{i-1})c_{i-1}.$$

- The knowledge on c_i might still be known when the difference at the $(i-1)$-th bit satisfies $\mathtt{eq}(\alpha_{i-1}, \beta_{i-1}, \gamma_{i-1}) = 0$.

Example $\mathrm{Cxy0}_{(i,i-1)}$

- When $\begin{cases}(\alpha_i, \beta_i, \gamma_i) = (0,1,0), \\ (\alpha_{i-1}, \beta_{i-1}, \gamma_{i-1}) = (1,1,0)\end{cases}$, one knows that $\begin{cases}\mathtt{eq}(\alpha,\beta,\gamma)_{i-1} = 0, \\ \alpha_{i-1} \oplus \beta_{i-1} = 0, \\ \mathtt{xor}(\alpha,\beta,\gamma)_i \oplus \alpha_{i-1} = 0.\end{cases}$
- According to the Observation $\star$, it is case $\mathrm{Cxy0}_{(i,i-1)}$, one has that $x_{i-1} \oplus y_{i-1} = 0$.
- Thus, $c_i = x_{i-1}y_{i-1} \oplus (x_{i-1} \oplus y_{i-1})c_{i-1} = y_{i-1}$.
- Therefore, $y_i \oplus c_i = y_i \oplus y_{i-1}$.
- As a consequence, the fulfillment of the constraint in case $\mathrm{Cyc}_{(i+1,i)}$ can be effectively predicted by observing whether $y_i \oplus y_{i-1} = \mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i$.

Cases for deducing the knowledge of the i-th carry bit c_i

Case No.	Difference	Value	Known
$\mathrm{Cy0c0}_{(i,i-1)}$		$y_{i-1} = 0, c_{i-1} = 0$	$c_i = 0$
$\mathrm{Cy1c1}_{(i,i-1)}$		$y_{i-1} = 1, c_{i-1} = 1$	$c_i = 1$
$\mathrm{Cxy0}_{(i,i-1)}$	$\mathrm{Cxy}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \alpha_{i-1} = 0$	$x_{i-1} \oplus y_{i-1} = 0$	$c_i = y_{i-1}$
$\mathrm{Cxy1}_{(i,i-1)}$	$\mathrm{Cxy}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \alpha_{i-1} = 1$	$x_{i-1} \oplus y_{i-1} = 1$	$c_i = c_{i-1}$
$\mathrm{Cxc0}_{(i,i-1)}$	$\mathrm{Cxc}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \alpha_{i-1} = 0$	$x_{i-1} \oplus c_{i-1} = 0$	$c_i = c_{i-1}$
$\mathrm{Cxc1}_{(i,i-1)}$	$\mathrm{Cxc}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \alpha_{i-1} = 1$	$x_{i-1} \oplus c_{i-1} = 1$	$c_i = y_{i-1}$
$\mathrm{Cyc0}_{(i,i-1)}$	$\mathrm{Cyc}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \beta_{i-1} = 0$	$y_{i-1} \oplus c_{i-1} = 0$	$c_i = y_{i-1}$
$\mathrm{Cyc1}_{(i,i-1)}$	$\mathrm{Cyc}_{(i,i-1)}$ and $\mathtt{xor}(\alpha, \beta, \gamma)_i \oplus \beta_{i-1} = 1$	$y_{i-1} \oplus c_{i-1} = 1$	$c_i = x_{i-1}$

- Combining case $\mathrm{Cyc}_{(i+1,i)}$ with cases where c_i can be known, one gets several cases where the knowledge on y can be used to check whether the differential constraints are fulfilled.
- Apart from the general cases (C3 and C4), there are some special cases (C1 and C2) at the two least significant bits since the carry bit c_0 is 0.

Cases where the knowledge on y can be used to check the fulfillment of the differential constraints

Case No.	Difference	Known
C1	$\mathrm{Cyc}_{(0,-1)}$	$\mathtt{xor}(\alpha,\beta,\gamma)_1 \oplus \beta_0 = y_0$
C2	$\mathrm{Cyc}_{(2,1)}$ and $\mathrm{Cy0}_{(1,0)}$	$\mathtt{xor}(\alpha,\beta,\gamma)_2 \oplus \beta_1 = y_1$
C3	$\mathrm{Cyc}_{(i+1,i)}$ and ($\mathrm{Cxy0}_{(i,i-1)}$ or $\mathrm{Cxc1}_{(i,i-1)}$ or $\mathrm{Cyc0}_{(i,i-1)}$)	$\mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i = y_i \oplus y_{i-1}$
C4	$\mathrm{Cyc}_{(i+1,i)}$ and ($\mathrm{Cxy1}_{(i,i-1)}$ or $\mathrm{Cxc0}_{(i,i-1)}$) and ($\mathrm{Cxy0}_{(i-1,i-2)}$ or $\mathrm{Cxc1}_{(i-1,i-2)}$ or $\mathrm{Cyc0}_{(i-1,i-2)}$)	$\mathtt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i = y_i \oplus y_{i-2}$

Case	Observation ⋆					Multi-bit Consts. [Leu13]		Quasi-differential [BR22]		Extended DLCT [CY21]
No.	Differential		Value		Observe	org	new	diff	mask (w)	selected bits
C1	$\alpha_{1,0}$ $\beta_{1,0}$ $\gamma_{1,0}$	*1 *0 *1	$x_{1,0}$ $y_{1,0}$ $z_{1,0}$	** ** **	$y_0 = \alpha_1 \oplus$ $\beta_1 \oplus \gamma_1 \oplus 0$	-x -- -x	-x -0 -x	01 00 01	00 01 $+2^0$ 00	$[x_1, y_1, z_1], [x'_1,$ $y'_1, z'_1, y'_0]$
C2	$\alpha_{2,1,0}$ $\beta_{2,1,0}$ $\gamma_{2,1,0}$	*1* *0* *1*	$x_{2,1,0}$ $y_{2,1,0}$ $z_{2,1,0}$	*** **0 ***	$y_1 = \alpha_2 \oplus$ $\beta_2 \oplus \gamma_2 \oplus 0$	-x? --0 -x?	-x? -00 -x?	010 000 010	000 011 $+2^{-1}$ 000	$[x_2, y_2, z_2, y_0],$ $[x'_2, y'_2, z'_2, y'_1]$
C3	$\alpha_{i+1,i,i-1}$ $\beta_{i+1,i,i-1}$ $\gamma_{i+1,i,i-1}$	*01 *11 *00	$x_{i+1,i,i-1}$ $y_{i+1,i,i-1}$ $z_{i+1,i,i-1}$	*** *** ***	$y_i \oplus y_{i-1} =$ $\alpha_{i+1} \oplus \beta_{i+1} \oplus$ $\gamma_{i+1} \oplus 1$	--x -xx ---	--x ->x ---	001 011 000	000 011 -2^0 000	$[x_{i+1}, y_{i+1}, z_{i+1},$ $y_{i-1}], [x'_{i+1},$ $y'_{i+1}, z'_{i+1}, y'_i]$
C3	$\alpha_{i+1,i,i-1}$ $\beta_{i+1,i,i-1}$ $\gamma_{i+1,i,i-1}$	*11 *00 *11	$x_{i+1,i,i-1}$ $y_{i+1,i,i-1}$ $z_{i+1,i,i-1}$	*** *** ***	$y_i \oplus y_{i-1} =$ $\alpha_{i+1} \oplus \beta_{i+1} \oplus$ $\gamma_{i+1} \oplus 0$	-xx --- -xx	-xx -=- -xx	011 000 011	000 011 $+2^0$ 000	$[x_{i+1}, y_{i+1}, z_{i+1},$ $y_{i-1}], [x'_{i+1},$ $y'_{i+1}, z'_{i+1}, y'_i]$
C4	$\alpha_{i+1,i,i-1,i-2}$ $\beta_{i+1,i,i-1,i-2}$ $\gamma_{i+1,i,i-1,i-2}$	*111 *010 *101	$x_{i+1,i,i-1,i-2}$ $y_{i+1,i,i-1,i-2}$ $z_{i+1,i,i-1,i-2}$	**** **** ****	$y_i \oplus y_{i-2} =$ $\alpha_{i+1} \oplus \beta_{i+1} \oplus$ $\gamma_{i+1} \oplus 0$	-xxx --x- -x-x	-xxx -$\substack{2\\0}$x- -x-x	0111 0010 0101	0000 0101 $+2^0$ 0000	$[x_{i+1}, y_{i+1}, z_{i+1},$ $y_{i-2}], [x'_{i+1},$ $y'_{i+1}, z'_{i+1}, y'_i]$

0: $y_i = y'_i = 0$ 1: $y_i = y'_i = 1$ -: $y_i = y'_i$ x: $y_i \neq y'_i$ $\substack{2\\0}$: 2.5-bit const. "28000014"

=: $y'_i = y_i = y_{i-1}$!: $y'_i = y_i \neq y_{i-1}$ <: $y'_i \neq y_i = y_{i-1}$ >: $y'_i \neq y_i \neq y_{i-1}$

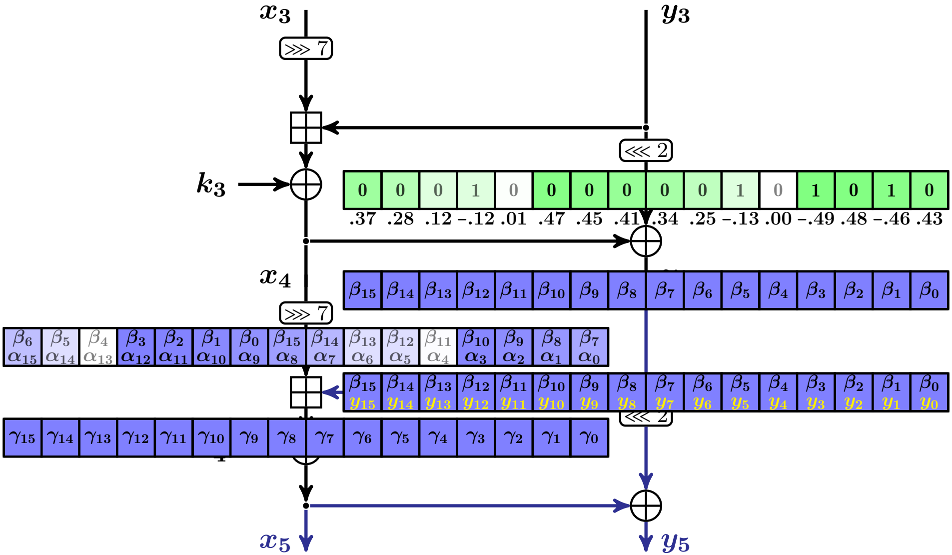
A simple procedure to improve the DDT-based distinguisher: $\mathcal{YD}^{\text{Speck}_{rR}}$

For an r-round Speck32/64, given its DDT$_{\texttt{(0040, 0000)}}$, one does the following to improve a DDT-based distinguisher and gets a distinguisher named $\mathcal{YD}^{\textsc{Speck}_{rR}}$.

1. Compute the bias (towards 0) of each bit of $(\delta_R^{r-2})^{\lll 2}$;
2. Predicts the value of each bit of $(\delta_R^{r-2})^{\lll 2}$ according to the bias (supposes it to be 0 if its bias ≥ 0 and 1 if its bias < 0), and denotes the absolute bias of the i-bit of $((\delta_R^{r-2})^{\lll 2})^{\ggg 7}$ by $\epsilon_\alpha(i)$.
3. For each output pair $((C_L, C_R), (C'_L, C'_R))$ of r-round Speck32/64, one does the follows to predict its classification.

A simple procedure to improve the DDT-based distinguisher: $\mathcal{YD}^{\mathrm{Speck}_{rR}}$

Predicts the value of each bit of $(\delta_R^{r-2})^{\lll 2}$ according to the bias.



A simple procedure to improve the DDT-based distinguisher: $\mathcal{YD}^{\mathrm{Speck}_{rR}}$

1. Get the differential probability p of $(\texttt{0040, 0000}) \mapsto (C_L \oplus C'_L, C_R \oplus C'_R)$ by looking up the table $\mathrm{DDT}_{(\texttt{0040, 0000})}[(C_L \oplus C'_L, C_R \oplus C'_R)]$
2. Compute the following information around the last $\boxplus$ from $((C_L, C_R), (C'_L, C'_R))$:
   1. $\gamma \leftarrow C_L \oplus C'_L$,
   2. $\beta \leftarrow (C_L \oplus C_R \oplus C'_L \oplus C'_R)^{\ggg 2}$,
   3. $\alpha \leftarrow ((\delta_R^{r-2})^{\lll 2} \oplus \beta)^{\ggg 7}$,
   4. $y \leftarrow (C_L \oplus C_R)^{\ggg 2}$.
3. For bit position 0, if $\epsilon_\alpha(1) > \tau$ and $\epsilon_\alpha(0) > \tau$, do:
   1. If $\mathrm{Cyc}_{(0,-1)}$, do: $p \leftarrow (1 + (-1)^{\texttt{xor}(\alpha,\beta,\gamma)_1 \oplus \beta_0 \oplus y_0}) \cdot p$.
4. For bit position 1, if $\epsilon_\alpha(2) > \tau$ and $\epsilon_\alpha(1) > \tau$, do:
   1. If $\mathrm{Cyc}_{(2,1)}$ and $y_0 = 0$, do: $p \leftarrow (1 + (-1)^{\texttt{xor}(\alpha,\beta,\gamma)_2 \oplus \beta_1 \oplus y_1}) \cdot p$.
5. For each bit position i $(1 < i < n-1)$, if $\epsilon_\alpha(i+1) > \tau$ and $\epsilon_\alpha(i) > \tau$ and $\epsilon_\alpha(i-1) > \tau$, do:
   1. If $\mathrm{Cyc}_{(i+1,i)}$ and $(\mathrm{Cxy0}_{(i,i-1)}$ or $\mathrm{Cxc1}_{(i,i-1)}$ or $\mathrm{Cyc0}_{(i,i-1)})$, do:
      $p \leftarrow (1 + (-1)^{\texttt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i \oplus y_i \oplus y_{i-1}}) \cdot p$.
   2. If $\mathrm{Cyc}_{(i+1,i)}$ and $(\mathrm{Cxy1}_{(i,i-1)}$ or $\mathrm{Cxc0}_{(i,i-1)})$ and $(\mathrm{Cxy0}_{(i-1,i-2)}$ or $\mathrm{Cxc1}_{(i-1,i-2)}$ or $\mathrm{Cyc0}_{(i-1,i-2)})$ and $\epsilon_\alpha(i-2) > \tau$, do:
      $p \leftarrow (1 + (-1)^{\texttt{xor}(\alpha,\beta,\gamma)_{i+1} \oplus \beta_i \oplus y_i \oplus y_{i-2}}) \cdot p$.
6. If $p > 2^{-n}$, predict $Z \leftarrow 1$; else predict $Z \leftarrow 0$.

Results of a simple improving of the DDT-based distinguisher

Accuracy of the improved DDT-based distinguishers ($\mathcal{YD}$s) on SPECK32/64 and comparisons with pure DDT-based ($\mathcal{DD}$s) distinguishers

#R	Name	ACC	TPR	TNR	Mem (GBytes)	Time (Secs/2^{20})
4	$\mathcal{DD}^{\textsc{Speck}_{4R}}$	0.9869	0.9869	0.9870	32.5	$2^{-4.98}$
4	$\mathcal{YD}^{\textsc{Speck}_{4R}}$	0.9907	0.9887	0.9928	32.5	$2^{-2.37}$
5	$\mathcal{DD}^{\textsc{Speck}_{5R}}$	0.9107	0.8775	0.9440	32.5	$2^{-4.94}$
5	$\mathcal{YD}^{\textsc{Speck}_{5R}}$	0.9215	0.8947	0.9484	32.5	$2^{-1.87}$
6	$\mathcal{DD}^{\textsc{Speck}_{6R}}$	0.7584	0.6795	0.8371	32.5	$2^{-4.53}$
6	$\mathcal{YD}^{\textsc{Speck}_{6R}}$	0.7663	0.7118	0.8207	32.5	$2^{-2.05}$
7	$\mathcal{DD}^{\textsc{Speck}_{7R}}$	0.5913	0.5430	0.6397	32.5	$2^{-4.49}$
7	$\mathcal{YD}^{\textsc{Speck}_{7R}}$	0.5962	0.5582	0.6343	32.5	$2^{-2.18}$
8	$\mathcal{DD}^{\textsc{Speck}_{8R}}$	0.5116	0.4963	0.5268	32.5	$2^{-4.64}$
8	$\mathcal{YD}^{\textsc{Speck}_{8R}}$	0.5117	0.4967	0.5268	32.5	$2^{-2.99}$

– For $\mathcal{YD}$s, the thresholds τ's for $\sigma_\alpha(i)$'s in building $\mathcal{YD}^{\textsc{Speck}_{4R}}$, $\mathcal{YD}^{\textsc{Speck}_{5R}}$, $\mathcal{YD}^{\textsc{Speck}_{6R}}$, $\mathcal{YD}^{\textsc{Speck}_{7R}}$ are 0.50, 0.30, 0.20, and 0.02, respectively. The number of samples for the accuracy testing is 2^{24}.

Fixed-y Averaging Differential Probability Distinguishers: $\mathcal{AD}_{\mathbf{YD}}^{\mathrm{Speck}_{rR}}$

1. $b \leftarrow 6$ // for practical reason, we consider 6-bit conditional DDT of $\boxplus$, which requires several metabytes.
2. $\mathbf{A}_0, \mathbf{A}_{\mathrm{next}}, \mathbf{A}^c_{\mathrm{next}} \leftarrow \mathrm{GenMultiBitsConditionalDDTs}(b)$
3. $p \leftarrow 0.0,\ q \leftarrow 1.0$
4. Compute the following information around the last $\boxplus$ from $((C_L, C_R), (C'_L, C'_R))$:
   1. $\gamma \leftarrow C_L \oplus C'_L$,
   2. $\beta \leftarrow (C_L \oplus C_R \oplus C'_L \oplus C'_R)^{\ggg 2}$,
   3. $y \leftarrow (C_L \oplus C_R)^{\ggg 2}$.
5. $\alpha \leftarrow \vec{0}, c \leftarrow \vec{0}$
6. $\beta_b \leftarrow$ LSB b bits of β, $\gamma_b \leftarrow$ LSB b bits of γ, $y_b \leftarrow$ LSB b bits of y
7. For $(\alpha_b, pr) \in \mathbf{A}_0[\beta_b, \gamma_b, y_b]$
   1. $\alpha \leftarrow \alpha_b$
   2. For i in $\{0, 1, \ldots, b-2\}$: $\mathrm{ComputeCarryNextBit}(c,\ \alpha,\ \beta,\ \gamma,\ y,\ i)$
   3. $\mathrm{ComputeAlphaPrNextBit}(c,\ \alpha,\ \beta,\ \gamma,\ y,\ b-1,\ q \times pr,\ p)$
8. If $p > 2^{-n}$, predict $Z \leftarrow 1$; else predict $Z \leftarrow 0$.

ComputeAlphaPrNextBit(c, α, β, γ, y, i, q, p) // update c_{i+1}, α_{i+1}, and p in-place

1. If i = WordSize − 1: $p \leftarrow p + q \times \mathcal{DD}^{\text{SPECK}_{r-1R}}(\alpha^{\lll 7} \| \beta)$; return
2. If $\mathtt{eq}(\alpha_i, \beta_i, \gamma_i)$:
    1. $\alpha_{i+1} \leftarrow \beta_{i+1} \oplus \gamma_{i+1} \oplus \beta_i$; ComputeCarryNextBit($c$, α, β, γ, y, i);
    2. ComputeAlphaPrNextBit(c, α, β, γ, y, $i+1$, $q \cdot 1$, p); return
3. Else if $\text{Cyc}_{(i+1,i)}$ and $c_i \neq \perp$:
    1. $\alpha_{i+1} \leftarrow \beta_{i+1} \oplus \gamma_{i+1} \oplus \beta_i \oplus y_i \oplus c_i$; ComputeCarryNextBit($c$, α, β, γ, y, i)
    2. ComputeAlphaPrNextBit(α, β, γ, y, $i+1$, $q \cdot 1$, p); return
4. Else:
    1. $\beta_b \leftarrow \beta_{\{i+1,\ldots,i+2-b\}}$, $\gamma_b \leftarrow \gamma_{\{i+1,\ldots,i+2-b\}}$, $y_b \leftarrow y_{\{i+1,\ldots,i+2-b\}}$, $\alpha_b \leftarrow \alpha_{\{i,\ldots,i+2-b\}}$
    2. If $c_{i+2-b} \neq \perp$: For $(\alpha_{i+1}, pr) \leftarrow \mathbf{A}^{c}_{\text{next}}[\beta_b, \gamma_b, y_b, \alpha_b, c_{i+2-b}]$
        - ComputeCarryNextBit(c, α, β, γ, y, i)
        - ComputeAlphaPrNextBit(α, β, γ, y, $i+1$, $q \cdot pr$, p)
    3. If $c_{i+2-b} = \perp$: For $(\alpha_{i+1}, pr) \leftarrow \mathbf{A}_{\text{next}}[\beta_b, \gamma_b, y_b, \alpha_b]$
        - ComputeCarryNextBit(c, α, β, γ, y, i)
        - ComputeAlphaPrNextBit(α, β, γ, y, $i+1$, $q \cdot pr$, p)

ComputeCarryNextBit(c, α, β, γ, y, i) // update c_{i+1} in-place

1. If $y_i = 0$ and $c_i = 0$: $c_{i+1} \leftarrow 0$
2. Else if $y_i = 1$ and $c_i = 1$: $c_{i+1} \leftarrow 1$
3. Else if $\mathrm{Cxy0}_{(i+1,i)}$ or $\mathrm{Cxc1}_{(i+1,i)}$ or $\mathrm{Cyc0}_{(i+1,i)}$: $c_{i+1} \leftarrow y_i$
4. Else if $\mathrm{Cxy1}_{(i+1,i)}$ or $\mathrm{Cxc0}_{(i+1,i)}$: $c_{i+1} \leftarrow c_i$
5. Else: $c_{i+1} \leftarrow \perp$. // $\perp$ means unknown

GenMultiBitsConditionalDDTs(b)

1. $\mathbf{A}_0 \leftarrow$ Generate b-bit conditional DDT of $\boxplus$, each entry is indexed by (b-bit β, b-bit γ, b-bit y), the values are (b-bit α, non-zero pr). // Table $\mathbf{A}_0$ will be used for the first b bits since one knows that both LSB carry bits are 0.
2. $\mathbf{A}_{\text{next}} \leftarrow$ Generate b-bit conditional DDT of $\boxplus$, each entry is indexed by (b-bit β, b-bit γ, b-bit y, $(b-1)$-bit α), the values are (1-bit α_{next}, non-zero pr). // Table $\mathbf{A}_{\text{next}}$ will be used for the intermediate bits when 1-bit LSB carry c is unknown.
3. $\mathbf{A}^c_{\text{next}} \leftarrow$ Generate b-bit conditional DDT of $\boxplus$, each entry is indexed by (b-bit β, b-bit γ, b-bit y, $(b-1)$-bit α, 1-bit carry c), the values are (1-bit α_{next}, non-zero pr). // Table $\mathbf{A}^c_{\text{next}}$ will be used for the intermediate bits when 1-bit LSB carry c is known.
4. Output $\mathbf{A}_0, \mathbf{A}_{\text{next}}, \mathbf{A}^c_{\text{next}}$

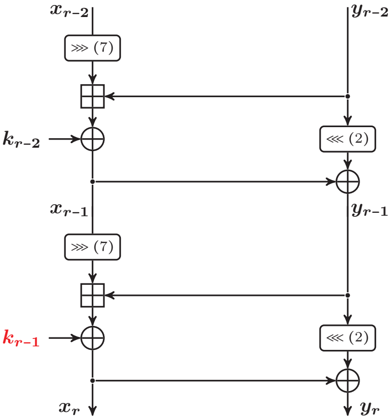
Fixed-y Averaging Differential Probability Distinguishers: $\mathcal{AD}_{\mathbf{YD}}^{\text{Speck}_{rR}}$

#R	Name	ACC	TPR	TNR	Mem (GBytes)	Time (Secs/2^{20})
5	$\mathcal{DD}^{\textsc{Speck}_{5R}}$	0.9107	0.8775	0.9440	32.5	$2^{-4.94}$
5	$\mathcal{ND}^{\textsc{Speck}_{5R}}$	0.9273	0.9011	0.9536	0.0277	$2^{+3.56}$
5	$\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}_{5R}}$	0.9362	0.9173	0.9552	32.5	$2^{+5.46}$
6	$\mathcal{DD}^{\textsc{Speck}_{6R}}$	0.7584	0.6795	0.8371	32.5	$2^{-4.53}$
6	$\mathcal{ND}^{\textsc{Speck}_{6R}}$	0.7876	0.7197	0.8554	0.0277	$2^{+3.54}$
6	$\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}_{6R}}$	0.7949	0.7309	0.8587	32.5	$2^{+5.12}$
7	$\mathcal{DD}^{\textsc{Speck}_{7R}}$	0.5913	0.5430	0.6397	32.5	$2^{-4.49}$
7	$\mathcal{ND}^{\textsc{Speck}_{7R}}$	0.6155	0.5325	0.6985	0.0277	$2^{+3.57}$
7	$\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}_{7R}}$	0.6237	0.5428	0.7048	32.5	$2^{+5.33}$
8	$\mathcal{DD}^{\textsc{Speck}_{8R}}$	0.5116	0.4963	0.5268	32.5	$2^{-4.64}$
8	$\mathcal{ND}^{\textsc{Speck}_{8R}}$	0.5135	0.5184	0.5085	0.0277	$2^{+3.55}$
8	$\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}_{8R}}$	0.5187	0.4914	0.5460	32.5	$2^{+5.51}$

Conclusion

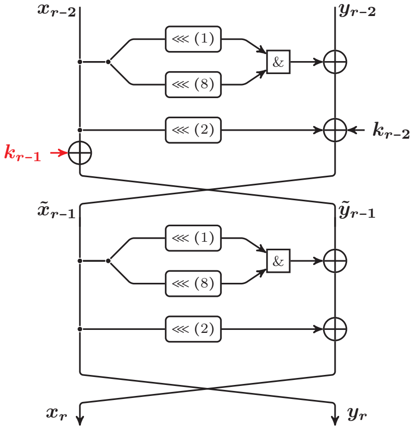
- By utilizing conditional differential distributions when the input and/or output values of the last nonlinear operation are observable, a distinguisher can surpass pure DDT-based counterparts.
- Accordingly, $\mathcal{ND}$'s advantage over pure differential-based distinguishers likely comes from exploiting the conditional differential distribution under the partially known value from ciphertexts input to the last non-linear operation.
- These findings apply not only to the SPECK but also to other block ciphers, such as SIMON and GIFT.

Explainability of Neural Distinguishers on Round-Reduced SIMON32/64



SPECK32/64 last 2-round

r-round $\mathcal{ND}$ can learn additional knowledge beyond r-round full DDT.



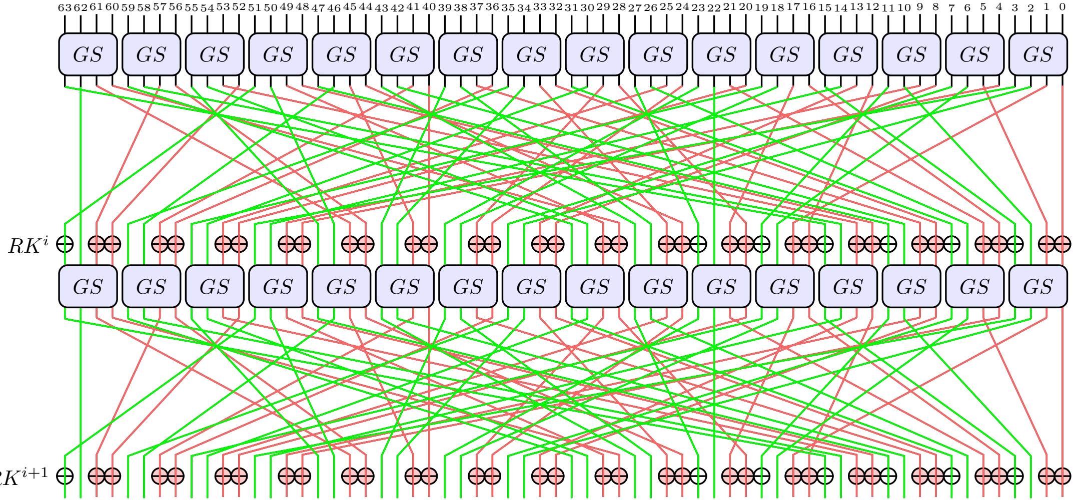
SIMON32/64 last 2-round

r-round $\mathcal{ND}$ can learn $r-1$-round full DDT, but there are no additional knowledge to learn.

Neural Distinguishers on Round-Reduced SIMON32/64

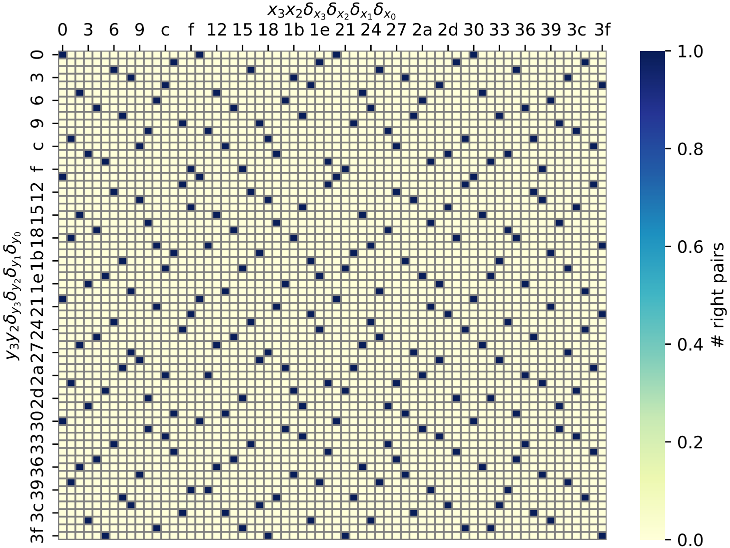
#R	Name	Network	Accuracy	True Positive Rate	True Negative Rate
6	$\mathcal{DD}^{\text{SIMON}6R}_{\mathbf{DD}}$	DDT	0.9918	0.9995	0.9841
7	$\mathcal{ND}^{\text{SIMON}7R}_{\mathbf{VV}}$	ResNet	$0.9823 \pm 1.2 \times 10^{-4}$	$0.9996 \pm 2.7 \times 10^{-5}$	$0.9650 \pm 2.3 \times 10^{-4}$
7	$\mathcal{DD}^{\text{SIMON}7R}_{\mathbf{DD}}$	DDT	0.8465	0.8641	0.8288
8	$\mathcal{ND}^{\text{SIMON}8R}_{\mathbf{VV}}$	SENet	$0.8150 \pm 4.2 \times 10^{-4}$	$0.8418 \pm 5.5 \times 10^{-4}$	$0.7882 \pm 5.1 \times 10^{-4}$
8	$\mathcal{DD}^{\text{SIMON}8R}_{\mathbf{DD}}$	DDT	0.6628	0.5781	0.7476
8	$\mathcal{ND}^{\text{SIMON}8R}_{\mathbf{VD}}$	SENet	$0.6587 \pm 4.8 \times 10^{-4}$	$0.5586 \pm 7.4 \times 10^{-4}$	$0.7588 \pm 5.6 \times 10^{-4}$
9	$\mathcal{ND}^{\text{SIMON}9R}_{\mathbf{VV}}$	SENet	$0.6515 \pm 5.3 \times 10^{-4}$	$0.5334 \pm 7.0 \times 10^{-4}$	$0.7695 \pm 5.7 \times 10^{-4}$
9	$\mathcal{DD}^{\text{SIMON}9R}_{\mathbf{DD}}$	DDT	0.5683	0.4691	0.6674
9	$\mathcal{ND}^{\text{SIMON}9R}_{\mathbf{VD}}$	SENet	$0.5657 \pm 4.9 \times 10^{-4}$	$0.4748 \pm 7.1 \times 10^{-4}$	$0.6565 \pm 6.6 \times 10^{-4}$
10	$\mathcal{ND}^{\text{SIMON}10R\ +}_{\mathbf{VV}}$	SENet	$0.5610 \pm 4.5 \times 10^{-4}$	$0.4761 \pm 6.0 \times 10^{-4}$	$0.6460 \pm 7.2 \times 10^{-4}$
10	$\mathcal{DD}^{\text{SIMON}10R}_{\mathbf{DD}}$	DDT	0.5203	0.5002	0.5404
11	$\mathcal{ND}^{\text{SIMON}11R}_{\mathbf{VV}}$	SENet	$0.5174 \pm 5.3 \times 10^{-4}$	$0.5041 \pm 7.1 \times 10^{-4}$	$0.5307 \pm 7.9 \times 10^{-4}$

Applying to GIFT-64-128



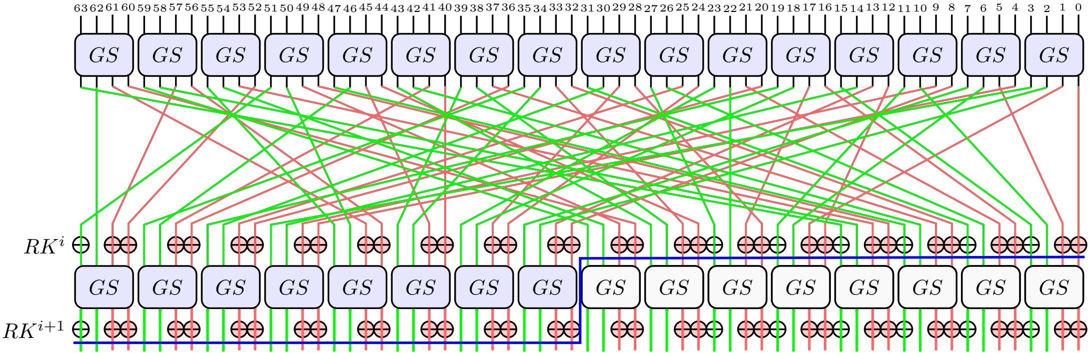
Applying to GIFT-64-128: $\mathcal{ND} + \mathcal{YD}$

The vDDT of GIFT's inverse SBox: $\mathrm{vDDT}_{y_3 y_2 \delta_{y_3} \delta_{y_2} \delta_{y_1} \delta_{y_0} \rightarrow x_3 x_2 \delta_{x_3} \delta_{x_2} \delta_{x_1} \delta_{x_0}}$



Applying to GIFT-64-128

$\mathcal{ND}$s on $(r-1)$.5-round GIFT, $\mathcal{YD}$s basing on $(r-1)$.5-round $\mathcal{ND}$ and 8 vDDTs:



Taking average over $\prod_{i=0}^{7} \#v\delta_i = 4^8 = 2^{16}$ possibilities

Applying to GIFT-64-128

#R	Name	Accuracy	True Positive Rate	True Negative Rate
5	$\mathcal{DD}$	0.8428	0.7693	0.9160
5	$\mathcal{ND}$	0.9001	0.8623	0.9378
5	$\mathcal{ND}_{4.5}$ + 8 vDDTs	0.9009	0.8615	0.9398
6	$\mathcal{DD}$	0.6305	0.4988	0.7623
6	$\mathcal{ND}$	0.6802	0.5571	0.8029
6	$\mathcal{ND}_{5.5}$ + 8 vDDTs	0.6885	0.5692	0.8066
7	$\mathcal{DD}$	0.5019	0.4525	0.5513
7	$\mathcal{ND}$	0.5348	0.5266	0.5431
7	$\mathcal{ND}_{6.5}$ + 8 vDDTs	0.5361	0.5116	0.5633
8	$\mathcal{ND}$	0.5003	1.0	0.0
8	$\mathcal{ND}_{7.5}$ + 8 vDDTs	0.5073	0.3823	0.6282

Explainability of Related-key Neural Distinguishers ($\mathcal{RK}$-$\mathcal{ND}$'s)

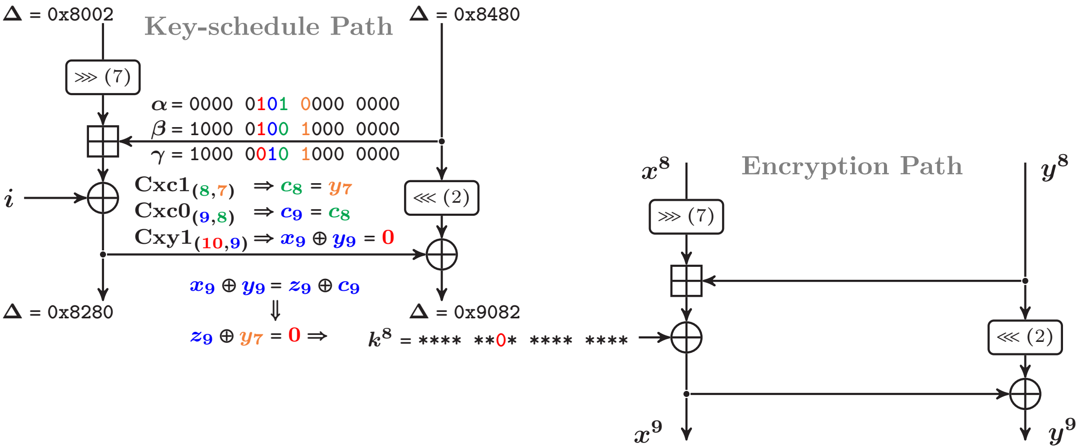
Diff.	#R	Name	Accuracy	True Positive Rate	True Negative Rate
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}_{8R}}$	0.8989	0.8714	0.9264
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}_{8R}}$	0.9259	0.9063	0.9455
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{AD}^{\textsc{Speck}_{8R}}_{\mathbf{YD}}$	0.9315	0.9159	0.9470
ID_2	9	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}_{9R}}$	0.7128	0.6644	0.7612
ID_2	9	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}_{9R}}$	0.7535	0.7035	0.8036
ID_2	9	$\mathcal{RK}\text{-}\mathcal{AD}^{\textsc{Speck}_{9R}}_{\mathbf{YD}}$	0.7574	0.7114	0.8035
ID_3	9	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}_{9R}}$	0.7128	0.6644	0.7612
ID_3	9	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}_{9R}}$	0.7726	0.7247	0.8206
ID_3	9	$\mathcal{RK}\text{-}\mathcal{AD}^{\textsc{Speck}_{9R}}_{\mathbf{YD}}$	0.7574	0.7113	0.8035
ID_3	10	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}_{10R}}$	0.5484	0.5343	0.5624
ID_3	10	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}_{10R}}$	0.5562	0.5361	0.5765
ID_3	10	$\mathcal{RK}\text{-}\mathcal{AD}^{\textsc{Speck}_{10R}}_{\mathbf{YD}}$	0.5713	0.5357	0.6069

Explainability of Related-key Neural Distinguishers ($\mathcal{RK}$-$\mathcal{ND}$'s)

Diff.	#R	Name	Accuracy	True Positive Rate	True Negative Rate
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}8R}$	0.8989	0.8714	0.9264
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}8R}$	0.9259	0.9063	0.9455
ID_2/ID_3	8	$\mathcal{RK}\text{-}\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}8R}$	0.9315	0.9159	0.9470
ID_2	9	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}9R}$	0.7128	0.6644	0.7612
ID_2	9	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}9R}$	0.7535	0.7035	0.8036
ID_2	9	$\mathcal{RK}\text{-}\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}9R}$	0.7574	0.7114	0.8035
ID_3	9	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}9R}$	0.7128	0.6644	0.7612
ID_3	9	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}9R}$	0.7726	0.7247	0.8206
ID_3	9	$\mathcal{RK}\text{-}\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}9R}$	0.7574	0.7113	0.8035
ID_3	10	$\mathcal{RK}\text{-}\mathcal{DD}^{\textsc{Speck}10R}$	0.5484	0.5343	0.5624
ID_3	10	$\mathcal{RK}\text{-}\mathcal{ND}^{\textsc{Speck}10R}$	0.5562	0.5361	0.5765
ID_3	10	$\mathcal{RK}\text{-}\mathcal{AD}_{\mathbf{YD}}^{\textsc{Speck}10R}$	0.5713	0.5357	0.6069

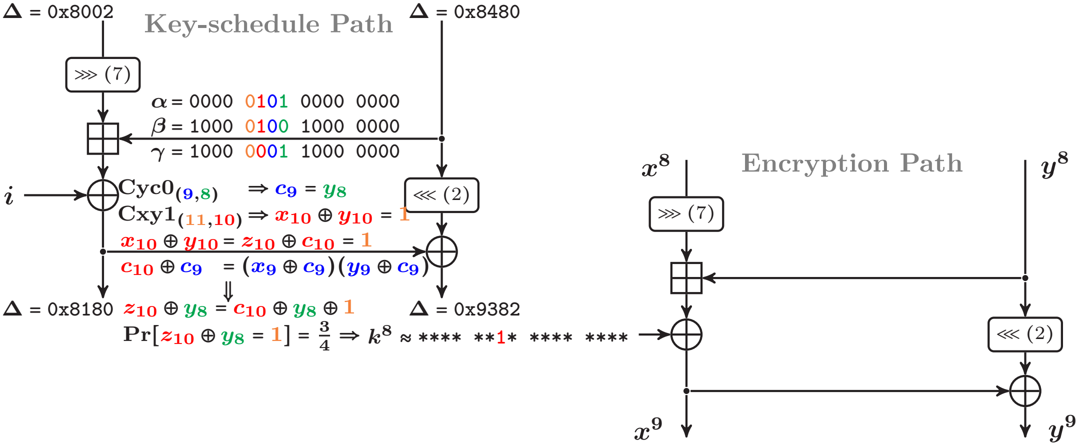
Explainability of Related-key Neural Distinguishers ($\mathcal{RK}$-$\mathcal{ND}$'s)

ID	Set.	Positive Samples	Negative Samples	Acc.
$\mathcal{RK}\text{-}\mathcal{ND}^{\text{Speck}9R}{}_{\text{ID}_{(3,9082)}}$	1-1	$(\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D})$	*Random*	0.7746
	1-2	$(\mathcal{AR}_1,\mathcal{BR}_1,\mathcal{CR}_1,\mathcal{DR}_1)$	*Random*	0.7539



Explainability of Related-key Neural Distinguishers ($\mathcal{RK}$-$\mathcal{ND}$'s)

ID	Set.	Positive Samples	Negative Samples	Acc.
$\mathcal{RK}\text{-}\mathcal{ND}^{\text{Speck}9R}\ \text{ID}_{(2,9382)}$	1-1	$(\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D})$	*Random*	0.7574
	1-2	$(\mathcal{AR}_1,\mathcal{BR}_1,\mathcal{CR}_1,\mathcal{DR}_1)$	*Random*	0.7529



Summary and More results

- Neural network can efficiently exploit complex correlations between ciphertext values, ciphertext differences, and intermediate state differences.
- Those observations on conditional differential probabilities are not intrinsically linked to neural network-based cryptanalysis but are expected to be useful in a wider range of cryptanalysis.
- Addressing the challenge of training high-round, especially 8-round, $\mathcal{ND}$ of SPECK32/64, we introduce the Freezing Layer Method. This method matches Gohr's accuracy but cuts training time and data.
- We introduce related-key ($\mathcal{RK}$) differences to slow down the diffusion of differences, aiding in training $\mathcal{ND}$ for higher rounds. As a result, we achieve a 14-round key recovery attack on SPECK32/64 using related-key neural distinguishers ($\mathcal{RK}$-$\mathcal{ND}$s).

Future Work

- How can we exploit further the Observation $\star$ and the conditional differential probability in traditional cryptanalysis?
- What we did is to explain the ML models, *i.e.*, providing human-understandable descriptions or reasons for the model's performance (Explainable AI); However, how to interpret the internal mechanics of the neural networks (Interpretable AI) to learn how they express the complex relations between input and outputs?
- $\mathcal{ND}$s are not aware of specific details of the ciphers, including their components and structure. Therefore, $\mathcal{ND}$s can be used for ciphers that have unknown components. However, if the machine learning model become knowledgeable about the cipher's specification, could they achieve higher accuracy?
- How to let machine learning model be knowledgeable about the cipher's specification so that it can learn beyond pure data-driven?

Thanks for your attention!

References I

[Goh19] Aron Gohr. "Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning". In: *CRYPTO 2019, Part II*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11693. LNCS. Springer, Heidelberg, Aug. 2019, pp. 150–179. DOI: 10.1007/978-3-030-26951-7_6.

[Ben+21] Adrien Benamira, David Gérault, Thomas Peyrin, and Quan Quan Tan. "A Deeper Look at Machine Learning-Based Cryptanalysis". In: *EUROCRYPT 2021, Part I*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12696. LNCS. Springer, Heidelberg, Oct. 2021, pp. 805–835. DOI: 10.1007/978-3-030-77870-5_28.

[Bel+22] Emanuele Bellini, David Gérault, Anna Hambitzer, and Matteo Rossi. "A Cipher-Agnostic Neural Training Pipeline with Automated Finding of Good Input Differences". In: *IACR Cryptol. ePrint Arch.* (2022), p. 1467. URL: https://eprint.iacr.org/2022/1467.

[GLN22] Aron Gohr, Gregor Leander, and Patrick Neumann. *An Assessment of Differential-Neural Distinguishers*. Cryptology ePrint Archive, Report 2022/1521. https://eprint.iacr.org/2022/1521. 2022.

[LM02] Helger Lipmaa and Shiho Moriai. "Efficient Algorithms for Computing Differential Properties of Addition". In: *FSE 2001*. Ed. by Mitsuru Matsui. Vol. 2355. LNCS. Springer, Heidelberg, Apr. 2002, pp. 336–350. DOI: 10.1007/3-540-45473-X_28.

[Leu12] Gaëtan Leurent. "Analysis of Differential Attacks in ARX Constructions". In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Springer, Heidelberg, Dec. 2012, pp. 226–243. DOI: 10.1007/978-3-642-34961-4_15.

[Leu13] Gaëtan Leurent. "Construction of Differential Characteristics in ARX Designs Application to Skein". In: *CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. LNCS. Springer, Heidelberg, Aug. 2013, pp. 241–258. DOI: 10.1007/978-3-642-40041-4_14.

[BR22] Tim Beyne and Vincent Rijmen. "Differential Cryptanalysis in the Fixed-Key Model". In: *CRYPTO 2022, Part III*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13509. LNCS. Springer, Heidelberg, Aug. 2022, pp. 687–716. DOI: 10.1007/978-3-031-15982-4_23.

[CY21] Yi Chen and Hongbo Yu. *Bridging Machine Learning and Cryptanalysis via EDLCT*. Cryptology ePrint Archive, Report 2021/705. https://eprint.iacr.org/2021/705. 2021.