Rich Input Representations in Neural Differential Cryptanalysis: A Taxonomy and Survey

Alireza Gholizadeh Shahrbejari*
Department of Computer Engineering, University of Guilan, Rasht, Iran

Reza Ebrahimi Atani†
Department of Computer Engineering, University of Guilan, Rasht, Iran

Abstract

Neural differential distinguishers have become an active research direction in symmetric-key cryptanalysis since the introduction of deep-learning-based attacks on round-reduced SPECK. Early neural distinguishers typically used a single ciphertext pair or ciphertext difference as input. Recent studies, however, show that richer input representations can substantially affect the information available to the classifier, the data cost of each labeled sample, and the relevance of the distinguisher to practical attacks. Examples include multi-pair, multi-difference, matrix-style, multi-round, structured-encoding, and score-aggregation based inputs.

This paper provides a taxonomy and survey of rich input representations in neural differential cryptanalysis. We introduce a representation-centric framework that describes an input representation by its difference set, number of observations per sample, sharing structure, encoding function, and ciphertext cost. Using this framework, we organize existing works into representation families and compare their motivations, benefits, and limitations. We also argue that representation-rich distinguishers require cost-aware evaluation: fixed-sample comparisons and fixed-ciphertext comparisons answer different questions and may lead to different conclusions. Finally, we identify open problems related to automated representation search, theoretical explanation of representation gain, cipher-family transferability, interpretability, reproducibility, and key-recovery integration. The survey highlights that rich input representations should be treated as first-class cryptanalytic design choices rather than secondary implementation details.

Keywords: neural cryptanalysis, neural differential distinguisher, input representation, multi-pair representation, multi-difference representation, lightweight block ciphers, differential cryptanalysis

1 Introduction

Differential cryptanalysis is one of the most influential techniques for evaluating the security of symmetric-key primitives. Since the seminal work of Biham and Shamir [5], many attacks on block ciphers have relied on studying how carefully chosen input differences propagate to output differences through reduced-round versions of a cipher. Traditionally, such distinguishers are designed using mathematical analysis, differential trails, experimentally estimated distributions, or search procedures over differential characteristics.

*Email: gholizadeh.a2000@gmail.com
†Email: rebrahimi@guilan.ac.ir

A major shift occurred when Gohr introduced a neural differential distinguisher for round-reduced SPECK32/64 and demonstrated that deep learning could improve practical attacks in a differential cryptanalysis setting [8]. In this paradigm, a neural network is trained to distinguish ciphertext-derived samples generated from structured plaintext pairs from samples generated according to a random-reference distribution. This result motivated a growing body of work on neural-aided cryptanalysis, including studies of what neural distinguishers learn, how their predictions relate to classical differential features, and how they can be used in key-recovery attacks [4]. Subsequent works have improved neural differential cryptanalysis along several directions. Some studies analyzed the behavior and limitations of neural distinguishers and compared them with classical differential evidence [4, 9, 20]. Others improved training procedures, neural architectures, or attack pipelines [2, 11, 26]. A complementary line of work studied automated input-difference search and dataset construction, showing that the quality of the generated training data can strongly affect the resulting distinguisher [3, 17]. These developments suggest that neural differential cryptanalysis is shaped not only by the classifier, but also by the way cryptanalytic evidence is generated and encoded.

Early neural differential distinguishers typically used a single ciphertext pair, or a single ciphertext difference, as the input sample. This setting is natural because it closely mirrors classical differential cryptanalysis: one fixes an input difference Δ, generates plaintext pairs of the form $(P, P \oplus \Delta)$, encrypts them through a reduced-round cipher, and gives a ciphertext-derived representation to a classifier. However, this single-pair viewpoint exposes only one local differential observation per sample. If the useful signal is weak, distributed across several related observations, or visible only after aggregation, then a single ciphertext pair may not provide enough information for the model.

More recent work has continued this trend through improved dataset construction, high-accuracy distinguishers for GIFT-128 and ASCON, polytopic differential-neural distinguishers for ARX families, related-key differential neural distinguishers for SPN ciphers, and IoT-friendly neural distinguisher frameworks for lightweight SPN ciphers [25, 16, 7, 13].

These developments suggest that input representation is not merely an implementation detail. It is a central design choice that determines what evidence is shown to the neural model, how much data is consumed per labeled sample, and how easily the resulting distinguisher can be compared with prior work or integrated into an attack. A richer representation may improve classification accuracy by aggregating weak statistical cues, but it may also increase the ciphertext cost per sample, depend strongly on the selected input differences, or make fair comparison with single-pair baselines more difficult.

This perspective is also relevant to practical security evaluation. Reduced-round cryptanalysis is commonly used as evidence for assessing security margins, especially for lightweight primitives intended for constrained environments. Therefore, representation-rich neural distinguishers should not be judged only by classification accuracy, but also by how their data cost, computational cost, and attack relevance affect their value as evaluation tools.

This survey develops a representation-centric view of neural differential cryptanalysis. Rather than surveying all uses of machine learning in cryptanalysis, we focus on how neural differential distinguishers construct and organize their input samples. In particular, the survey emphasizes representations that go beyond the single-pair baseline, including multi-pair, multi-difference, matrix-style, structured-encoding, and score-aggregation based inputs. We use the term input representation broadly: it includes the choice of input differences, the number of ciphertext-derived observations placed inside one sample, the sharing structure among those observations, the encoding function applied to ciphertexts or differences, and the ciphertext cost required to generate each labeled example.

The main thesis of this paper is that input representation should be treated as a first-class cryptanalytic design parameter. In particular, representation-rich distinguishers should not be evaluated only by accuracy or number of distinguished rounds. They should also be

compared according to the data cost of their samples, the structure of their input differences, their dependence on cipher families, and their relevance to key-recovery attacks.

The main contributions of this paper are as follows:

- We introduce a representation-centric framework for neural differential distinguishers. The framework describes an input representation by its difference set $\mathcal{D}$, number of observations per sample m, sharing structure s, encoding function ϕ, and ciphertext cost c.
- We provide a taxonomy of input representation families, including single-pair, multi-pair, multi-difference, matrix-style, multi-round, structured-encoding, and score-aggregation based representations.
- We organize representative works into a comparative literature map that emphasizes representation structure, target cipher family, design objective, and attack relevance.
- We discuss evaluation fairness for representation-rich neural distinguishers and distinguish between fixed-sample and fixed-ciphertext comparisons. We argue that both viewpoints are necessary for understanding whether richer representations are genuinely efficient.
- We identify open problems and future directions, including automated representation search, theory of representation gain, cipher-family transferability, interpretability of learned evidence, standard benchmarks, and integration into key-recovery attacks.

The rest of the paper is organized as follows. Section 2 introduces the neural differential distinguishing setting and the single-pair baseline. Section 3 presents the representation-centric framework. Section 4 develops the taxonomy of input representation families. Section 5 maps representative works according to this taxonomy. Section 6 discusses cost-aware evaluation and fair comparison protocols. Section 7 presents open problems and future directions. Section 8 concludes the paper.

2 Background

2.1 Differential-Neural Distinguishers

Let $E_K : \{0,1\}^n \to \{0,1\}^n$ denote an n-bit block cipher under a secret key K, and let E_K^r denote its r-round reduced version. In classical differential cryptanalysis, the analyst studies how an input difference Δ propagates through the cipher and induces non-random behavior in the corresponding output differences. For a plaintext pair (P, P') satisfying

$$P' = P \oplus \Delta,$$

the corresponding ciphertexts are

$$C = E_K^r(P), \qquad C' = E_K^r(P').$$

If the reduced-round cipher preserves some detectable statistical structure, then the ciphertext pair (C, C'), or its difference $C \oplus C'$, may be distinguishable from a random reference distribution.

Neural differential distinguishers follow the same general idea, but replace an explicitly designed statistical test with a trained classifier. A dataset is generated from two classes. In the real class, plaintext pairs follow a prescribed input difference and are encrypted through the reduced-round cipher. In the random class, the relation between the two ciphertexts is removed or replaced by a random reference. A neural network is then trained to predict whether a given ciphertext-derived sample belongs to the real or random class.

The Gohr-style paradigm also motivated more generic machine-learning-assisted differential distinguishers. Baksi et al. studied machine-learning-assisted differential distinguishers for lightweight ciphers [1], while Yadav and Kumar proposed a differential-ML distinguisher as a generic extension of classical differential cryptanalysis [24]. These works are not always representation-rich in the same sense as multi-pair or multi-difference neural distinguishers, but they are important for the broader transition from handcrafted statistical tests to learned differential evidence.

From the perspective of this survey, the important point is that a neural differential distinguisher is not determined only by the neural architecture. It is also determined by the way the cryptanalytic evidence is encoded before being given to the model. The input sample may contain one ciphertext pair, several ciphertext pairs, multiple ciphertext differences, intermediate-round information, or other derived features. Therefore, input representation design is a central part of neural differential cryptanalysis.

2.2 Single-Pair Input Representation

The simplest and most common representation is the single-pair input. In this setting, a sample is generated from one plaintext pair $(P, P \oplus \Delta)$ and the corresponding ciphertext pair (C, C'). The neural model may receive the two ciphertexts directly, their XOR difference, or a binary encoding derived from them. A typical input can be written as

$$X = C \oplus C',$$

or, in a richer but still single-pair form, as

$$X = (C, C', C \oplus C').$$

The associated label y indicates whether the sample comes from the real differential distribution or from the random reference distribution.

This representation has several advantages. It is simple, close to the classical differential setting, and inexpensive in terms of data generation: each labeled sample requires only one plaintext pair and its corresponding ciphertext pair. It also provides a clean baseline for comparing different neural architectures and training strategies. For this reason, single-pair representations appear in many neural distinguisher studies, either as the main input format or as a baseline against richer constructions.

However, the single-pair representation also has an important limitation. Each sample exposes only one local view of the cipher's differential behavior. If the useful signal is weak, distributed across several related pairs, or visible only after aggregating multiple ciphertext-derived features, then a single pair may not provide enough information for the classifier. This limitation has motivated a line of work on richer input representations, including multiple ciphertext-pair inputs [6, 10], multi-round and multi-splicing constructions [14], and approaches based on multiple ciphertext differences or score distributions [18]. Before reviewing these representation families, we first introduce a representation-centric framework that provides common terminology for comparing them.

3 A Representation-Centric Framework

A central difficulty in comparing neural differential distinguishers is that different works often change several design choices at the same time. A proposed distinguisher may use more ciphertext pairs, a different input difference, a larger neural architecture, a new encoding of the ciphertexts, or a different random-reference construction. When these choices are not separated, it becomes difficult to determine whether an observed improvement comes from the neural model, the data-generation procedure, or the input representation itself.

To address this issue, this section introduces a representation-centric framework for describing neural differential distinguisher inputs. The goal is not to impose a single formal model on all existing works, but to provide common terminology for comparing single-pair, multi-pair, multi-difference, multi-round, and score-aggregation based representations. In this view, an input representation is treated as a cryptanalytic design choice that determines what evidence is shown to the classifier and at what data cost.

3.1 Representation Components

We describe an input representation by the tuple

$$\mathcal{R} = (\mathcal{D}, m, s, \phi, c),$$

where each component captures one aspect of the sample-construction process.

The set $\mathcal{D}$ denotes the input difference or the collection of input differences used to generate structured plaintexts. In the simplest case, $\mathcal{D} = \{\Delta\}$ contains a single difference. In richer settings, $\mathcal{D}$ may contain several differences

$$\mathcal{D} = \{\Delta_1, \Delta_2, \ldots, \Delta_m\},$$

which may be selected manually, obtained from classical differential analysis, or found by an automated search procedure.

The parameter m denotes the number of differential observations included in one labeled sample. In a single-pair representation, $m = 1$. In a multi-pair or multi-difference representation, $m > 1$, and each sample contains several ciphertext pairs, ciphertext differences, or derived features. We refer to m as the differential richness of the representation.

The component s denotes the sharing structure among the observations inside a sample. For example, the observations may be generated from independent plaintext bases, or they may share a common plaintext base. This distinction is important because two representations with the same number of rows may induce different statistical dependencies and require different numbers of encryption queries.

The function ϕ denotes the encoding function that maps the generated ciphertexts into the actual input given to the neural model. Depending on the work, ϕ may output a raw ciphertext pair, an XOR difference, a concatenation of several ciphertexts, a binary matrix, a vector of handcrafted features, or a distribution of neural scores.

Finally, c denotes the ciphertext cost of one labeled sample. This quantity measures how many ciphertext blocks, or equivalently how many reduced-round encryption queries, are required to construct one sample under the representation $\mathcal{R}$. The cost component is essential for comparing representations fairly, because richer inputs may consume more cryptanalytic data per sample.

3.2 Observation Count and Differential Richness

The most direct way to enrich a neural distinguisher input is to increase the number of observations per sample. In a single-pair representation, the model receives only one local view of the cipher's differential behavior. In contrast, a multi-pair representation may use

$$X = \phi\big((C_1, C_1'), (C_2, C_2'), \ldots, (C_m, C_m')\big),$$

where each pair is generated under the same or related differential condition. A multi-difference representation may instead use several differences around one or more plaintext bases, producing an input of the form

$$X = \phi\big(C_1 \oplus C_1',\, C_2 \oplus C_2',\, \ldots,\, C_m \oplus C_m'\big).$$

Increasing m can make the input more informative because the classifier observes several weak statistical cues at once. This can be useful when no individual ciphertext pair contains a strong enough signal, but several related observations jointly deviate from the random-reference distribution. However, increasing m may also reduce the number of independent training samples available under a fixed data budget. Thus, differential richness creates a trade-off between information density per sample and the number of independent examples available for learning.

3.3 Sharing Structure

The sharing structure s specifies whether the observations inside one sample are generated independently or share some common element. In an independent multi-pair representation, each row may come from a separate plaintext pair:

$$(P_i, P_i \oplus \Delta), \qquad i = 1, \ldots, m.$$

This construction aggregates several independent differential observations into one input sample. If each row requires two encryptions, the ciphertext cost is typically

$$c = 2m.$$

In a shared-base construction, several differences may be applied around the same plaintext base P:

$$P, \quad P \oplus \Delta_1, \quad P \oplus \Delta_2, \quad \ldots, \quad P \oplus \Delta_m.$$

The corresponding ciphertexts can be used to form several difference rows with respect to the same base ciphertext. In this case, the ciphertext cost can be as low as

$$c = m + 1,$$

because one base ciphertext is shared across several rows.

This distinction matters for two reasons. First, independent and shared-base representations may expose different statistical information to the classifier. Second, they have different cost profiles. A comparison that treats both constructions as simply "m-row inputs" may miss an important cryptanalytic difference: the number of ciphertexts required to build each sample is not the same.

3.4 Encoding Functions

The encoding function ϕ determines how ciphertext-derived information is presented to the learning algorithm. Even when the same plaintext differences and ciphertexts are used, different encodings may lead to different classifiers. Common choices include

$$\phi(C, C') = C \oplus C',$$

which gives only the ciphertext difference, and

$$\phi(C, C') = (C, C', C \oplus C'),$$

which exposes both ciphertexts and their XOR relation.

For richer representations, ϕ may arrange multiple observations as a binary matrix:

$$X = \begin{bmatrix} \mathrm{bits}(C_1 \oplus C_1') \\ \mathrm{bits}(C_2 \oplus C_2') \\ \vdots \\ \mathrm{bits}(C_m \oplus C_m') \end{bmatrix}.$$

This matrix-style encoding is natural for neural models because it preserves the row structure of the observations while allowing the classifier to learn patterns across both bit positions and difference rows.

Other works may use more processed encodings. For example, a first-level classifier may assign scores to several ciphertext-derived candidates, and the distribution of these scores may then be used as a higher-level feature. Such score-aggregation approaches replace raw ciphertext information with a derived statistical summary. They may improve performance when weak evidence is spread across several differences, but they also introduce additional design choices that affect interpretability and reproducibility.

3.5 Cost Models

The cost component c is necessary because neural differential distinguishers are often compared at a fixed number of labeled samples. This convention is convenient for machine learning experiments, but it can be misleading from a cryptanalytic perspective. If two representations have different ciphertext costs per sample, then training both on N samples does not imply that they use the same amount of data.

Let Q denote a fixed ciphertext budget and let $c(\mathcal{R})$ denote the number of ciphertexts required to construct one sample under representation $\mathcal{R}$. Then the number of available labeled samples is approximately

$$N_{\mathcal{R}} = \left\lfloor \frac{Q}{c(\mathcal{R})} \right\rfloor .$$

A richer representation may increase the information content of each sample, but it may also reduce $N_{\mathcal{R}}$. Therefore, fair evaluation should distinguish between at least two comparison modes.

In a fixed-sample comparison, two representations are trained and evaluated using the same number of labeled samples. This measures the effect of changing the input format, but it does not control the number of ciphertexts consumed. In a fixed-ciphertext comparison, the total number of ciphertexts is held constant, so a representation with higher per-sample cost must use fewer samples. This comparison is closer to the data-complexity viewpoint of cryptanalysis.

Both comparison modes can be useful, but they answer different questions. Fixed-sample evaluation asks whether a richer representation is more informative per labeled example. Fixed-ciphertext evaluation asks whether the richer representation is a better use of a limited cryptanalytic data budget. A complete evaluation of representation-rich neural distinguishers should ideally report both.

The framework introduced in this section will be used in the following sections to organize existing works according to their input differences, differential richness, sharing structure, encoding function, and data cost. This representation-centric view makes it possible to compare approaches that otherwise appear quite different, and it highlights why input design should be considered a first-class component of neural differential cryptanalysis.

Figure 1 summarizes the proposed framework and shows how its components correspond to the main families of rich input representations.

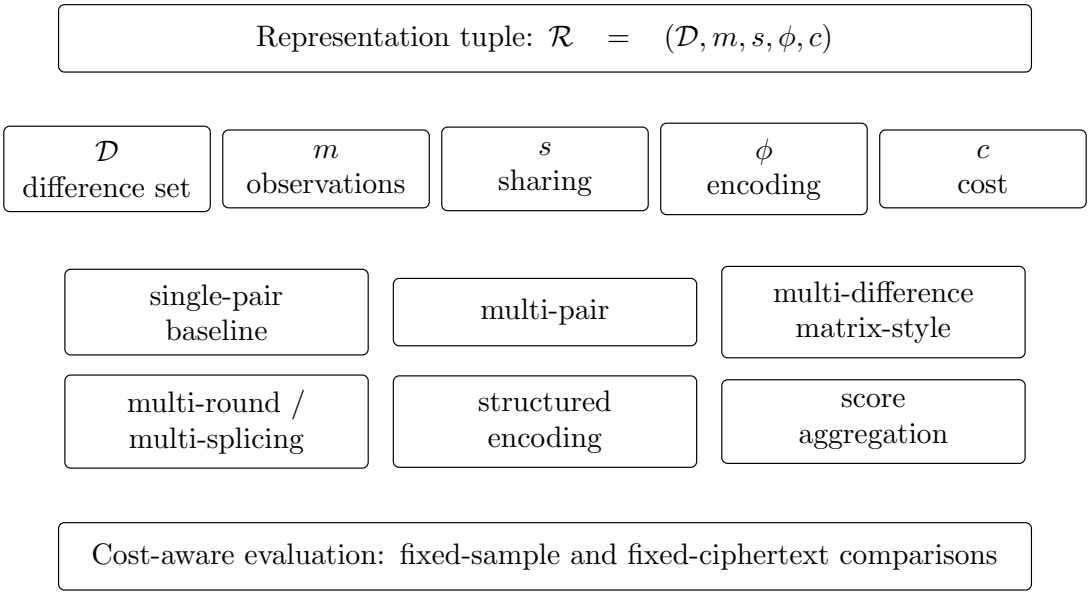

Figure 1: A representation-centric map of rich input representations in neural differential cryptanalysis. The tuple $\mathcal{R} = (\mathcal{D}, m, s, \phi, c)$ separates the main design components of a neural distinguisher input: the difference set, observation count, sharing structure, encoding function, and ciphertext cost. Rich representations go beyond the single-pair baseline by modifying one or more of these components, and should be evaluated under explicit cost models.

4 Taxonomy of Input Representations

The framework of the previous section allows us to describe existing neural differential distinguisher inputs in terms of five components: the difference set $\mathcal{D}$, the number of observations per sample m, the sharing structure s, the encoding function ϕ, and the ciphertext cost c. In this section, we use this viewpoint to organize the main representation families that appear in neural differential cryptanalysis.

The families below should not be understood as mutually exclusive categories. Several works combine more than one idea, for example by using multiple ciphertext pairs together with a specialized neural architecture, or by aggregating score distributions over multiple ciphertext differences. Nevertheless, the taxonomy is useful because it separates the main ways in which a neural distinguisher input can be enriched beyond the single-pair baseline.

4.1 Single-Pair Representations

The single-pair representation is the baseline from which most richer constructions can be understood. In the notation of the framework, it corresponds to a singleton difference set $\mathcal{D} = \{\Delta\}$, one observation per sample $m = 1$, and an encoding function ϕ that maps one ciphertext pair to a binary input. The most common choices are

$$\phi(C, C') = C \oplus C'$$

or

$$\phi(C, C') = (C, C', C \oplus C').$$

The ciphertext cost is typically $c = 2$, since one related plaintext pair gives two ciphertexts.

This representation is attractive because it is simple, inexpensive, and close to the classical differential setting. Gohr's neural distinguisher for SPECK32/64 is the standard example of this paradigm [8]. A single input difference is fixed, many plaintext pairs are generated under

this difference, and the neural model learns to separate cipher-derived samples from a random-reference class.

From the representation-centric viewpoint, the main limitation is that $m = 1$. Each sample exposes only one local differential observation. If the useful statistical signal is weak or distributed across several related observations, the classifier may not receive enough information from a single pair. This limitation motivates the richer representation families discussed below.

4.2 Multiple Ciphertext-Pair Representations

A direct way to enrich the input is to increase the number of ciphertext pairs per sample. In this family, the representation has $m > 1$, while the difference set may still be a singleton $\mathcal{D} = \{\Delta\}$. A typical input has the form

$$X = \phi\big((C_1, C_1'), (C_2, C_2'), \ldots, (C_m, C_m')\big),$$

where the pairs are generated under the same or related differential condition. The sharing structure is often independent-base: each row comes from a separate plaintext pair. In that case, the ciphertext cost is usually $c = 2m$.

Chen et al. proposed a neural distinguisher that considers features derived from multiple ciphertext pairs simultaneously [6]. The motivation is that several ciphertext pairs may expose statistical information that is not visible from one pair alone. Hou et al. further explored multiple ciphertext-pair inputs for SPECK and SIMON and combined richer dataset construction with neural design choices [10]. These works suggest that multi-pair inputs can strengthen neural distinguishers, especially when individual pairs carry only weak evidence.

Earlier work also explored improvements to neural distinguishers for SIMON and SPECK, including constructions that make richer use of ciphertext-derived information [11]. Related studies on SIMECK considered multiple differential distinguishers based on deep learning, further illustrating that aggregating several ciphertext-derived observations can be useful when a single differential view is weak [23].

The main trade-off is cost. A multi-pair representation may improve accuracy at a fixed number of labeled samples, but each sample consumes more ciphertexts. Therefore, if two representations are compared at the same sample count, the multi-pair method may receive a larger encryption-query budget. In the notation of the framework, the performance gain should be interpreted together with the increase from $c = 2$ to $c = 2m$.

4.3 Multiple-Difference and Matrix-Style Representations

A second enrichment strategy is to expose several input differences inside one sample. Here, the difference set is no longer a singleton:

$$\mathcal{D} = \{\Delta_1, \Delta_2, \ldots, \Delta_m\}.$$

The corresponding ciphertext-derived observations can be arranged as a vector or matrix:

$$X = \begin{bmatrix} \mathrm{bits}(C_1 \oplus C_1') \\ \mathrm{bits}(C_2 \oplus C_2') \\ \vdots \\ \mathrm{bits}(C_m \oplus C_m') \end{bmatrix}.$$

In this family, the role of the representation is not only to increase the number of observations, but also to provide complementary differential views of the cipher.

This family is particularly important for representation-rich neural distinguishers because it changes both the number and the diversity of differential observations shown to the model. Related multiple-differential viewpoints have been studied for lightweight ciphers such as SIMECK

[23], while score- and difference-aggregation based approaches have been explored for GIFT-128 and ASCON [18]. These works suggest that the benefit of a richer input may come not only from seeing more ciphertext-derived rows, but also from exposing complementary differential behavior.

Multiple-difference representations can be instantiated with different sharing structures. In an independent construction, each row is generated from a separate plaintext pair, and the cost is close to $c = 2m$. In a shared-base construction, the rows may be generated around a common plaintext base P, for example from

$$P, \quad P \oplus \Delta_1, \quad P \oplus \Delta_2, \quad \ldots, \quad P \oplus \Delta_m.$$

In this case, one base ciphertext can be shared across several rows, and the cost can be closer to $c = m + 1$. Thus, two matrix-style representations with the same number of rows may have different data costs and different statistical dependencies.

Matrix-style encodings are useful because they preserve the row structure of the input. A neural model can learn patterns across bit positions as well as across difference rows. This is particularly relevant when several individually weak differences become informative when viewed together. However, the effectiveness of such representations depends strongly on the choice of $\mathcal{D}$. Poorly selected differences may add cost without adding useful signal, while complementary differences may provide a much richer view of the cipher's reduced-round behavior.

4.4 Multi-Round and Multi-Splicing Representations

A third family enriches the input by constructing features from multiple rounds or by splicing together ciphertext-derived components. In the framework notation, these methods usually modify both the encoding function ϕ and the observation structure. The input may still contain multiple pairs or differences, but the key design choice is that the raw ciphertext information is transformed into a more structured representation before being given to the neural model.

Liu et al. proposed improved neural distinguishers based on multi-round and multi-splicing constructions [14]. Such representations are designed to expose information that may not be conveniently available in the final ciphertext difference alone. By combining round-dependent or spliced components, the input may align better with the statistical structure that the neural network can learn.

Other improvement-oriented works also modified the data format or training pipeline of neural distinguishers for lightweight ciphers. For example, Lyu et al. studied improved deep-learning-based differential distinguishers with applications to SIMECK [15], while Yue and Wu proposed an improved neural differential distinguisher model for SPECK [26]. These works reinforce the observation that gains in neural distinguishers often arise from a combination of input format, training procedure, and architecture.

From the taxonomy viewpoint, these methods are more complex than simple multi-pair or multi-difference inputs because several design choices are changed simultaneously. The observed gain may come from a larger observation count m, a more informative encoding ϕ, a different use of round-dependent information, or the neural architecture used to process the representation. This makes attribution difficult. For this reason, multi-round and multi-splicing distinguishers are important examples of representation-rich design, but they also highlight the need to separate representation effects from architecture and training effects.

4.5 Structured Encoding and Feature-Construction Approaches

Some works focus less on increasing the number of ciphertext pairs and more on how the available ciphertext information is encoded. In the framework, these methods primarily modify ϕ. For example, the same underlying ciphertext pair or set of pairs may be encoded as raw ciphertexts, XOR differences, concatenated bit strings, bit matrices, or other structured features.

This family is important because it shows that representation design is not only about using more data. Even when the ciphertext cost c and the difference set $\mathcal{D}$ are fixed, changing the encoding function ϕ can affect what the model can learn. A poor encoding may hide useful statistical relations, while a structured encoding may make them easier to exploit.

Wang et al. investigated enhanced input formats for neural differential distinguishers and studied how different input organizations affect performance on SPECK and SIMON [22]. Related improvement-oriented studies also suggest that modifying the input format can change the effectiveness of the resulting distinguisher even when the target cipher family is unchanged [21, 26]. A recent SPN-oriented study also emphasizes the role of training data format in neural distinguishers for IoT-friendly lightweight block ciphers such as SKINNY and MIDORI [13].

At the same time, structured encodings can be cipher-dependent. An encoding that matches the word-oriented structure of an ARX cipher may not transfer directly to an SPN cipher with bit permutations and S-box layers. Therefore, feature-construction approaches should be evaluated not only by their accuracy on one primitive, but also by their transferability across cipher families.

4.6 Score-Distribution and Feature-Aggregation Approaches

A final family replaces raw ciphertext-derived inputs with aggregated evidence. In this setting, one may first evaluate several ciphertext-difference candidates and then summarize the resulting evidence through a score vector or score distribution. Shen et al. used this idea in neural differential distinguishers for GIFT-128 and ASCON, where multiple ciphertext differences are summarized through score distributions [18].

In terms of the framework, score-distribution methods usually have a nontrivial difference set $\mathcal{D}$, a potentially large observation count m, and an encoding function ϕ that maps raw observations to derived statistical features. This separates the representation into two levels. The first level extracts evidence from individual ciphertext-derived observations, while the second level aggregates the resulting scores into a higher-level input.

The advantage is that weak signals may become visible only after aggregation. This can be useful when no single difference gives a strong distinguishing advantage. The limitation is that the representation becomes harder to interpret and reproduce. Its performance depends on the chosen differences, the scoring procedure, and the aggregation rule. Moreover, if the score-generation process is tuned to a particular cipher or round setting, the resulting representation may not transfer easily to other primitives.

4.7 Summary

The representation families above enrich the neural distinguisher input in different ways. Multi-pair representations mainly increase the number of repeated observations. Multi-difference and matrix-style representations expose complementary differential views. Multi-round and multi-splicing constructions modify the structure of the evidence shown to the model. Structured encodings change how ciphertext information is organized, while score-distribution methods aggregate lower-level evidence into higher-level features.

The common theme is that input representation determines both the information available to the classifier and the cost of obtaining that information. Therefore, the central question is not only whether a richer representation improves distinguishing accuracy, but also whether the improvement remains meaningful under a fair cost model and whether the representation can be integrated into a realistic cryptanalytic workflow. The next section compares representative works according to these criteria.

5 Comparative Literature Map

The taxonomy in Section 4 describes input representations according to their structure. We now use this taxonomy to map representative works in neural differential cryptanalysis. The purpose of this section is not to rank distinguishers by accuracy, because reported results are obtained under different ciphers, rounds, datasets, input differences, and model architectures. Instead, our goal is to clarify how the literature has evolved from single-pair distinguishers toward richer representation designs.

5.1 Comparison Criteria

We compare representative works using five criteria derived from the representation-centric framework. First, we record the target cipher or cipher family, since the usefulness of a representation may depend on whether the primitive is ARX-based, SPN-based, or permutation-based. Second, we identify the main representation type, such as single-pair, multi-pair, multi-difference, multi-round, or score-aggregation based input. Third, we summarize the main design objective of the work, for example improving distinguishing accuracy, supporting key recovery, automating difference selection, or improving dataset quality. Fourth, we note whether the work is mainly representation-oriented or whether representation changes are combined with architecture, training, or attack-level changes. Finally, we record the main limitation from the viewpoint of representation comparison.

This comparison is intentionally representation-centered. A work may be important even if it does not introduce a new representation family, because it may clarify how neural distinguishers behave, how input differences should be selected, or how distinguishers can be used in attacks. Conversely, a work may propose a richer input format but leave open questions about cost, transferability, or the source of the observed gain.

5.2 Evolution of the Literature

The development of neural differential cryptanalysis can be viewed along two related directions. The first direction established the neural differential paradigm, analyzed what neural distinguishers learn, and studied their relation to classical differential evidence. The second direction enriched the input representation itself, for example by using multiple ciphertext pairs, multiple differences, score distributions, or more structured encodings. We separate these two directions in Tables 1 and 2 to make the literature map easier to read.

Table 1: Foundational, analytical, and methodology-oriented works in neural differential cryptanalysis.

Work	Main focus	Relevance to representation design	Main lesson
Gohr, 2019 [8]	Neural differential distinguisher and key recovery for SPECK32/64	Establishes the single-pair neural differential baseline.	Neural distinguishers can be integrated into practical differential attacks.
Baksi et al., 2020 [1]	ML-assisted differential distinguishers for lightweight ciphers	Connects learned classifiers with differential cryptanalysis beyond a single target.	ML can support differential distinguishing beyond handcrafted tests.
Benamira et al., 2021 [4]	Analysis of machine-learning-based cryptanalysis	Helps interpret what neural distinguishers may learn from ciphertext-derived inputs.	Neural predictions should be related back to classical cryptanalytic features.
Yadav and Kumar, 2021 [24]	Generic differential-ML distinguisher	Frames learned distinguishers as an extension of classical differential cryptanalysis.	Learned distinguishers can be studied as generic differential tools.
Bao et al., 2022 [2]	Enhancing differential-neural cryptanalysis	Improves the complete neural cryptanalysis pipeline rather than only the input format.	Representation effects may interact with training and attack-level refinements.
Gohr et al., 2022 [9]	Assessment of differential-neural distinguishers	Provides analytical perspective on the behavior and limitations of learned distinguishers.	Neural improvements require careful interpretation and validation.
Bellini et al., 2023 [3]	Automated search for useful input differences	Shows that the choice of input difference is a key part of sample construction.	Representation quality depends strongly on difference selection.
Wang and Wang, 2024 [20]	Balance between classical and neural distinguishing	Motivates fair comparison between learned evidence and classical differential evidence.	Neural distinguishers should be evaluated in relation to classical baselines and costs.
Seok and Lee, 2025 [17]	Dataset construction for differential-neural cryptanalysis	Emphasizes that generated data quality affects the resulting neural distinguisher.	Input representation and dataset construction should be reported explicitly.

Table 1 shows that representation design is connected to several broader methodological issues. Neural distinguishers are not only neural networks trained on ciphertexts; they depend on input-difference selection, dataset construction, random-reference generation, and the way learned evidence is interpreted. These works provide the context in which richer input representations should be evaluated.

Table 2: Representation-rich neural differential distinguishers and related input-design works.

Work	Target cipher(s)	Representation contribution	Main limitation or open issue
Hou et al., 2021 [11]	SIMON and SPECK	Improves neural distinguishers through data and model design choices.	Representation, training, and architecture effects are not fully separated.
Sun et al., 2022 [19]	TinyJAMBU-128 and GIFT-64	Extends neural distinguishing experiments to lightweight primitives beyond SPECK-like ARX ciphers.	Mainly distinguishing-oriented; representation comparison is limited.
Wang et al., 2022 [23]	SIMECK32/64	Studies multiple differential evidence using deep learning.	The cost and independence of multiple observations require careful interpretation.
Chen et al., 2023 [6]	SPECK, PRESENT, DES, Chaskey, SHA3-256	Introduces neural inputs based on features derived from multiple ciphertext pairs.	Multi-pair gains should be interpreted together with their higher data cost.
Liu et al., 2023 [14]	SPECK and SIMON	Uses multi-round and multi-splicing constructions to enrich the neural input.	Several design factors change simultaneously, making attribution difficult.
Yue and Wu, 2023 [26]	SPECK	Improves neural differential distinguishing through model and data-format refinements.	Mainly cipher-specific; transferability remains unclear.
Zhang et al., 2023 [27]	SIMECK32/64	Studies differential-neural cryptanalysis and key recovery with richer distinguishing settings.	Attack relevance is considered, but representation cost remains an important factor.
Shen et al., 2024 [18]	GIFT-128 and ASCON	Uses multiple ciphertext differences and score-distribution based evidence.	Aggregated scores improve evidence but can reduce interpretability and reproducibility.
Wang et al., 2024 [22]	SPECK and SIMON	Investigates enhanced input formats and structured encodings.	Encoding choices may be cipher-family dependent.
Yadav and Kumar, 2025 [25]	GIFT-128 and ASCON	Extends high-accuracy ML-based distinguishing to additional lightweight primitives.	Further cost-aware and cross-family comparisons are needed.
Hou et al., 2025 [10]	SPECK and SIMON	Combines multiple ciphertext-pair inputs with improved dataset and neural design.	Representation gains and architecture gains are intertwined.
Liu et al., 2025 [13]	SKINNY and MIDORI	Studies an IoT-friendly neural distinguisher framework for lightweight SPN ciphers, emphasizing training data format and two-dimensional input construction.	Representation and architecture effects are studied together, so their individual contributions require careful separation.
Mirzaali et al., 2026 [16]	SIMON, SIMECK, and SPECK	Studies polytopic differential-neural distinguishers for ARX-oriented block ciphers.	Richer differential structures require systematic cost and transferability analysis.
Ge and Wang, 2026 [7]	SKINNY-64/64 and PRESENT-64/80	Develops a systematic related-key differential neural distinguisher framework with feature-enhancement and sample-enhancement mechanisms.	Focused on the related-key setting; representation gains are coupled with related-key data generation.

Table 2 highlights the trend that motivates this survey. Recent works increasingly move beyond the single-pair setting by enriching the input sample through multiple pairs, multiple differences, structured encodings, or aggregated score features. These representations can expose more cryptanalytic evidence to the classifier, but they also raise new questions about data cost, independence of observations, cipher-family dependence, and attack integration.

Several trends are visible from the two-part literature map. First, the field has moved from single-pair neural distinguishers toward richer sample formats. Second, representation design is often studied together with other improvements, such as architecture changes, training refinements, dataset construction, or attack integration. This makes it difficult to isolate the effect of the representation alone. Third, the target cipher family matters: many representation-rich approaches have been studied on ARX ciphers such as SPECK, SIMON, and SIMECK, while fewer systematic studies focus on SPN or permutation-based primitives. Finally, recent work suggests that the next stage of the field should not only propose stronger neural inputs, but also evaluate them under explicit cost models and reproducible comparison protocols.

5.3 Representation Families Across Works

Table 3 summarizes the main representation families using the notation introduced in Section 4. The table abstracts away from individual neural architectures and focuses on the sample-construction mechanism.

Table 3: Representation families and their main trade-offs.

Family	Framework profile	Representative works	Main benefit	Main risk
Single-pair	$\mathcal{D} = \{\Delta\}$, $m = 1$, usually $c = 2$	Gohr [8]	Simple, inexpensive, and close to classical differential cryptanalysis.	Gives the model only one local differential observation per sample.
Multiple ciphertext-pair	Usually $\mathcal{D} = \{\Delta\}$, $m > 1$, often independent-base, $c = 2m$	Chen et al. [6], Hou et al. [10]	Aggregates several observations and can amplify weak statistical evidence.	Higher per-sample data cost; fixed-sample comparisons may be misleading.
Multiple-difference matrix	$\lvert\mathcal{D}\rvert > 1$, $m > 1$, independent or shared-base, $c \approx 2m$ or $m+1$	Shen et al. [18]	Exposes complementary differential views in a structured input.	Depends strongly on difference selection and row construction.
Multi-round and multi-splicing	Modifies m, s, and ϕ through round-dependent or spliced features	Liu et al. [14]	May align the input more closely with learnable cipher structure.	Gains are difficult to attribute to one design factor.
Structured encoding	Keeps data source similar but changes ϕ	Wang et al. [22]	Shows that input organization can matter even without simply adding more data.	Encoding choices may be cipher-family dependent.
Score distribution and aggregation	Uses $\lvert\mathcal{D}\rvert > 1$, $m > 1$, and a derived aggregate ϕ	Shen et al. [18]	Converts several weak signals into higher-level statistical evidence.	Harder to interpret, reproduce, and transfer across ciphers.
Dataset and difference-selection methods	Mainly affects $\mathcal{D}$ and the generated sample distribution	Bellini et al. [3], Seok and Lee [17]	Improves the quality of the evidence shown to the model.	May overfit to specific ciphers, rounds, or selection protocols.

The comparison shows that richer representations can be obtained in several distinct ways. Some methods increase the number of observations m, some enlarge or optimize the difference set $\mathcal{D}$, and others change the encoding function ϕ. These mechanisms should not be conflated. For example, a multi-pair distinguisher and a structured-encoding distinguisher may both improve accuracy, but the former mainly changes the amount of evidence per sample while the latter changes how similar evidence is presented to the model.

Another important distinction is whether a representation changes the data cost c. Structured encodings may keep c fixed while changing ϕ. In contrast, multi-pair and multi-difference inputs usually increase c, unless sharing is used to reuse a common plaintext or ciphertext base. This distinction is central for fair evaluation, because a representation that improves accuracy at a fixed number of labeled samples may not be better under a fixed ciphertext budget.

5.4 Attack Relevance

A representation-rich neural distinguisher is most useful when it can be connected to an attack workflow. Some works explicitly consider this connection, for example by using the neural distinguisher in key-recovery procedures or by comparing it with classical differential tools [8, 6, 27]. Other works mainly focus on distinguishing accuracy or representation quality.

This distinction matters because not every representation that improves classification accuracy is equally convenient for key recovery. A representation may require many related ciphertext pairs, several input differences, score aggregation over many candidates, or a large number of neural evaluations per key guess. Such requirements can affect time, data, and memory complexity. Therefore, attack relevance should be treated as a separate criterion from test accuracy.

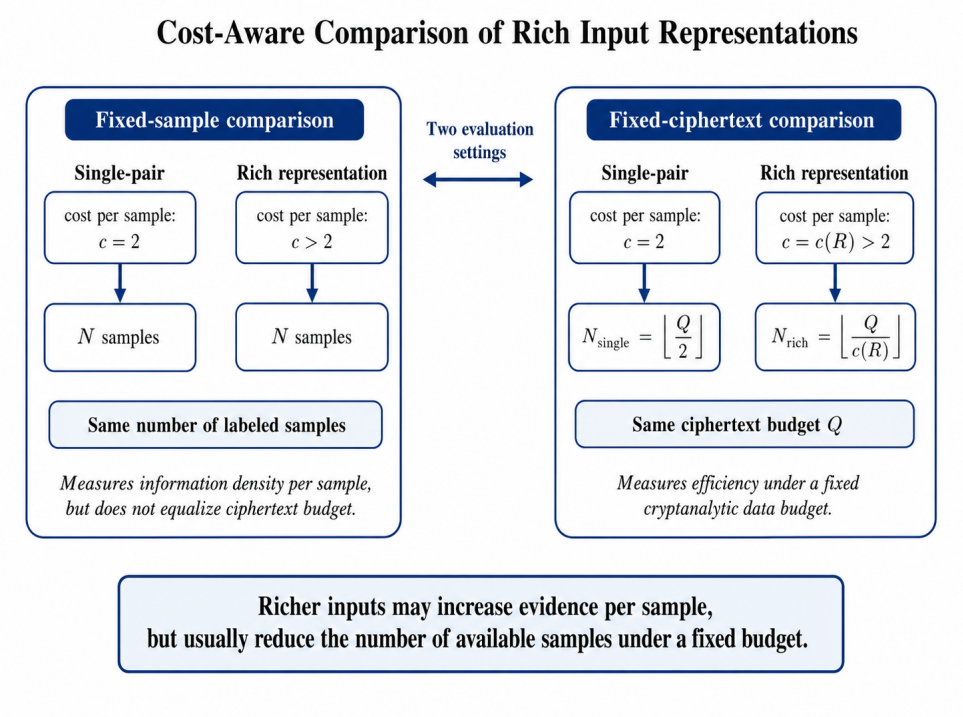

Figure 2: Cost-aware comparison of rich input representations. In a fixed-sample comparison, single-pair and richer representations are evaluated using the same number of labeled samples, but they may consume different ciphertext budgets. In a fixed-ciphertext comparison, the total ciphertext budget is fixed, so richer representations with higher per-sample cost generally yield fewer labeled samples.

From a survey perspective, the current literature suggests that representation-rich neural distinguishers are promising but not yet standardized. The field lacks a common reporting format for input structure, ciphertext cost, difference selection, and attack integration. The next section focuses on this issue and discusses why evaluation fairness is essential when comparing richer input representations.

6 Evaluation Fairness and Cost Models

A central challenge in evaluating representation-rich neural distinguishers is that input representations may differ not only in their information content, but also in their data cost. In machine learning experiments, models are often compared at a fixed number of labeled samples. This convention is convenient and easy to reproduce, but it can be misleading in a cryptanalytic setting. A labeled sample is not a unit-cost object: depending on the representation, it may require one ciphertext pair, several ciphertext pairs, several differences around a shared base, or an additional score-aggregation procedure.

This section discusses why cost-aware evaluation is necessary for comparing neural differential distinguishers. We distinguish between fixed-sample and fixed-ciphertext comparisons, explain how representation gains can be confounded with data-budget gains, and propose reporting guidelines for future work.

Figure 2 illustrates the distinction between fixed-sample and fixed-ciphertext comparisons for representation-rich neural distinguishers.

6.1 Fixed-Sample Comparisons

In a fixed-sample comparison, two distinguishers are trained and tested using the same number of labeled examples. If representation $\mathcal{R}_1$ and representation $\mathcal{R}_2$ are both trained on N samples, then the comparison controls the number of machine-learning examples:

$$N_{\mathcal{R}_1} = N_{\mathcal{R}_2}.$$

This is useful for studying whether one input format is more informative per labeled example. For example, if a multi-pair representation outperforms a single-pair representation at the same sample count, this suggests that each multi-pair example contains more learnable evidence.

However, fixed-sample evaluation does not control the cryptanalytic data cost. Suppose that a single-pair representation has cost $c = 2$, while a multi-pair representation with m independent pairs has cost $c = 2m$. Training both on N samples means that the multi-pair representation consumes m times more ciphertexts:

$$Q_{\text{single}} = 2N, \qquad Q_{\text{multi}} = 2mN.$$

Thus, an improvement observed at fixed sample count may reflect both a better representation and a larger encryption-query budget.

This does not make fixed-sample comparisons invalid. They answer a legitimate question: which representation gives the model more information per labeled sample? The problem is that this is not the same as asking which representation is more efficient under a fixed cryptanalytic data budget. Therefore, fixed-sample results should be interpreted as representation-density measurements rather than complete cryptanalytic comparisons.

This concern is related to broader attempts to compare neural distinguishers with classical distinguishing evidence. Wang and Wang emphasize that classical and neural distinguishers should be kept in balance when interpreting improvements in differential-neural cryptanalysis [20]. From the representation-centric viewpoint, this balance also requires reporting the cost of constructing each neural input sample.

6.2 Fixed-Ciphertext Comparisons

In a fixed-ciphertext comparison, the total number of ciphertexts or encryption queries is held constant. Let Q be the available ciphertext budget and let $c(\mathcal{R})$ be the cost of one labeled sample under representation $\mathcal{R}$. Then the number of training samples available to that representation is

$$N_{\mathcal{R}} = \left\lfloor \frac{Q}{c(\mathcal{R})} \right\rfloor.$$

Under this comparison mode, a richer representation receives fewer labeled samples if it has a higher per-sample cost. The evaluation therefore captures a different trade-off: whether the additional information inside each sample compensates for the loss of independent examples.

This distinction is especially important for multi-pair and multi-difference inputs. A multi-pair representation may give the classifier a stronger view of the differential distribution, but it may also reduce the number of independent plaintext bases available for training. Conversely, a single-pair representation may provide weaker individual samples but allow many more independent observations. A fair comparison should therefore ask whether the gain in differential richness outweighs the reduction in sample count.

Fixed-ciphertext evaluation is closer to the data-complexity viewpoint of classical cryptanalysis. In a real attack scenario, the number of chosen plaintexts or ciphertexts is a limited resource. A distinguisher that performs well only by consuming many more ciphertexts per sample may be less attractive than its fixed-sample accuracy suggests.

6.3 Independent Samples versus Information-Dense Samples

Representation-rich inputs create a tension between two sources of learning signal: independence and information density. Independent samples help generalization because they provide many separate draws from the underlying distribution. Information-dense samples, on the other hand, may expose several related statistical cues inside one example.

This trade-off is visible in the representation framework. Increasing m increases the number of observations per sample, but it may also increase $c(\mathcal{R})$. Under a fixed budget Q, this reduces $N_{\mathcal{R}}$. Therefore, a richer representation is beneficial only if the extra information per sample compensates for the smaller number of samples:

$$\text{higher information per sample} \quad \text{versus} \quad \text{fewer independent samples.}$$

The optimal balance is unlikely to be universal. It may depend on the cipher, the number of rounds, the selected differences, the model capacity, and the training regime.

This observation also explains why comparing only the maximum number of distinguished rounds can be incomplete. A representation may extend the number of rounds at the cost of substantially higher data usage. Conversely, a cheaper representation may be preferable if it achieves slightly lower accuracy but requires fewer ciphertexts and simpler attack integration.

6.4 Separating Representation Gains from Architecture Gains

Another fairness issue arises when representation changes are introduced together with architecture changes. For example, a work may propose a richer input format and also use a larger neural network, a new training schedule, or additional preprocessing. If the new method outperforms a baseline, it may be unclear which component is responsible for the gain.

A representation-centered evaluation should therefore include controlled comparisons whenever possible. Ideally, the same model family and training protocol should be tested with different representations, and the same representation should be tested with different model families. This separates two questions:

$$\text{What does the representation contribute?}$$

and

$$\text{What does the architecture contribute?}$$

Both questions are important, but they should not be conflated.

This issue is especially relevant for multi-round, multi-splicing, and score-aggregation approaches, where the representation may be tightly coupled with a particular neural architecture or decision rule. In such cases, the method may be effective as a complete pipeline, but its representation-level contribution is harder to isolate.

6.5 Reporting Guidelines

To make future comparisons more transparent, representation-rich neural distinguisher studies should report the input construction in a cost-aware way. At minimum, the following information should be specified:

- the input difference or set of differences $\mathcal{D}$;
- the number of observations or rows per sample m;
- the sharing structure s, such as independent-base or shared-base construction;
- the encoding function ϕ, including whether the model receives raw ciphertexts, XOR differences, matrices, handcrafted features, or score distributions;

- the ciphertext or encryption-query cost $c(\mathcal{R})$ per labeled sample;
- the number of training, validation, and test samples;
- whether results are reported under a fixed-sample, fixed-ciphertext, or other comparison model;
- whether the representation is evaluated independently from architecture and training changes;
- whether the distinguisher is used only for classification or integrated into a key-recovery attack.

These reporting guidelines are not meant to impose a single benchmark on all works. Different papers may have different goals: some may focus on stronger distinguishers, some on attack integration, and others on understanding neural behavior. Nevertheless, without explicit reporting of representation structure and data cost, it is difficult to compare results across papers or to determine whether a richer representation is genuinely more efficient.

6.6 Summary

Fair evaluation of neural differential distinguishers requires more than reporting accuracy or the number of rounds distinguished. When input representations differ in ciphertext cost, fixed-sample comparisons and fixed-ciphertext comparisons answer different questions. The former measures information density per labeled example, while the latter measures the use of a limited cryptanalytic data budget. A mature evaluation methodology should report both whenever possible.

The broader lesson is that representation design, data generation, model architecture, and attack integration should be treated as separate but interacting components. Making these components explicit is necessary for understanding which neural distinguisher improvements are due to richer cryptanalytic evidence, which are due to larger models, and which remain useful in realistic attack settings.

7 Open Problems and Future Directions

The previous sections showed that rich input representations have become an important component of neural differential cryptanalysis. However, the field still lacks a unified methodology for designing, comparing, and interpreting representation-rich neural distinguishers. This section summarizes several open problems that arise from the representation-centric view developed in this survey.

7.1 Automated Representation Search

Most neural differential distinguishers rely on manually selected input differences, manually designed input formats, or cipher-specific construction heuristics, although recent work has made progress on automated difference search and dataset construction [3, 17]. A natural extension is to search not only for a good input difference, but for a complete representation tuple

$$\mathcal{R} = (\mathcal{D}, m, s, \phi, c).$$

Such a search space includes the difference set $\mathcal{D}$, the number of observations per sample m, the sharing structure s, the encoding function ϕ, and the cost model c. Exploring this space manually is difficult, especially when the best representation may depend on the cipher, number of rounds, selected model family, and available data budget.

A representation-search procedure would need to balance several objectives at once. It should improve distinguishing performance, but also control ciphertext cost, avoid excessive model complexity, and remain robust across keys, datasets, and related cipher instances. This makes the problem different from ordinary hyperparameter search: the search space is cryptanalytic as well as statistical.

A major challenge is avoiding overfitting to a specific cipher and round number. A representation found by automated search may perform well in one experimental setting but fail to transfer. Therefore, automated representation search should be evaluated not only by peak accuracy, but also by robustness across keys, differences, datasets, and related cipher instances.

7.2 Theory of Representation Gain

A recurring empirical observation is that richer representations can improve neural distinguishers. However, the theoretical reason for this improvement is often unclear. A multi-pair or multi-difference input may help because it aggregates weak statistical signals, because it exposes dependencies that are invisible in a single pair, or because it makes the learning problem easier for the neural architecture.

A useful theoretical direction is to characterize representation gain in terms of statistical distance between the real and random-reference distributions. In a single-pair setting, the classifier observes one sample from a ciphertext-derived distribution. In a richer representation, the classifier observes a structured object built from several related ciphertext-derived observations. The key question is whether this transformation increases the distinguishable statistical signal per unit of cryptanalytic cost.

This question is closely related to the distinction between information density and sample independence. Increasing m may provide more evidence inside each sample, but it may also reduce the number of independent samples under a fixed budget. A theoretical account of representation gain should explain when aggregation helps, when it causes overfitting, and when the loss of independent examples outweighs the benefit of richer inputs.

7.3 Cipher-Family Transferability

Many representation-rich neural distinguishers have been studied on ARX ciphers such as SPECK, SIMON, and SIMECK [8, 14, 27, 10]. Other works consider lightweight SPN or permutation-based primitives such as PRESENT, GIFT, ASCON, DES, and SHA3-related settings [6, 18, 19]. The diversity of target primitives is valuable, but it also raises a transferability question: does a representation that works well for one cipher family remain effective for another?

This question is important because different cipher structures may expose different types of learnable information. ARX ciphers contain modular addition, rotation, and XOR operations, while SPN ciphers combine S-box layers, bit permutations, and linear diffusion. A word-oriented structured encoding may align well with an ARX primitive, whereas a matrix of bit-level differences may be more natural for an SPN design. Score-distribution methods may behave differently again, especially for permutation-based primitives.

Future work should therefore evaluate representation families across several cipher families under comparable protocols. Without such cross-family evaluation, it is difficult to determine whether a representation is generally useful or mainly tuned to one design style.

7.4 Interpretability of Learned Evidence

Another open problem is interpretability. Neural distinguishers can achieve high accuracy, but it is often unclear what cryptanalytic evidence they use. This issue becomes more difficult for richer representations. In a single-pair setting, one may try to relate the classifier's behavior to

output differences, bit biases, or classical differential features. In a multi-pair, multi-difference, or score-aggregation setting, the learned signal may involve interactions across rows, differences, or derived scores.

Interpretability is not only a machine-learning concern. It affects cryptanalytic trust. If a representation improves accuracy but the source of the improvement is unclear, then it is harder to judge whether the result reflects meaningful cipher structure, dataset artifacts, or accidental properties of the experimental setup. This is especially important when the representation is later used inside a key-recovery procedure.

A useful direction is to develop analysis tools that connect learned features back to classical differential quantities. Examples include studying which input differences contribute most to the prediction, which bit positions dominate the classifier's decision, or whether the neural score correlates with known differential probabilities. Recent interpretability-oriented work has also connected Fourier features learned by neural distinguishers to local constraints in ARX operations, illustrating how learned signals may be related back to concrete cipher structure [12]. Such analyses would make representation-rich distinguishers more transparent and easier to compare with classical cryptanalytic tools.

7.5 Standard Benchmarks and Reproducibility

The current literature uses different ciphers, round numbers, input differences, random reference constructions, dataset sizes, neural architectures, and evaluation metrics. This makes it difficult to compare representation families directly. A result may appear stronger because of a better representation, but also because of a larger model, a better difference, more training data, or a more favorable random-reference distribution.

A standard benchmark for representation-rich neural distinguishers should specify at least the following elements: target cipher and round number, key-generation procedure, difference set $\mathcal{D}$, representation structure, ciphertext cost per sample, model family, training budget, test protocol, and evaluation metric. Such a benchmark should also include both fixed-sample and fixed-ciphertext comparisons when the representations have different costs.

Reproducibility is particularly important for neural cryptanalysis because results may be sensitive to random seeds, training hyperparameters, and dataset generation. Future work should therefore report not only final accuracy values, but also variation across runs, confidence intervals when appropriate, and implementation details sufficient to reproduce the data-generation pipeline.

7.6 Integration into Key-Recovery Attacks

A neural distinguisher is most cryptanalytically meaningful when it can be used inside a key-recovery attack. Gohr's work demonstrated this connection for SPECK32/64 [8], and later works have also considered attack integration in different settings [6, 27]. However, many representation-focused studies mainly report distinguishing accuracy.

The gap between classification accuracy and attack relevance remains a major open problem. A richer representation may improve the distinguisher but make the attack harder to implement. For example, it may require several related plaintext structures, many ciphertext rows per key guess, or repeated score aggregation. These requirements may increase time, data, or memory complexity.

Future work should therefore evaluate representation-rich distinguishers not only as standalone classifiers, but also as components in attack pipelines. Important questions include whether the representation reduces the number of required plaintexts, whether it improves key-ranking performance, whether it remains effective under wrong-key decryption, and whether its computational cost is acceptable in a practical key-recovery setting.

7.7 Summary of Open Problems

Table 4 summarizes the main open problems discussed in this section.

Table 4: Open problems in representation-rich neural differential cryptanalysis.

Open problem	Main question	Why it matters
Automated representation search	How can one search over $\mathcal{D}$, m, s, and ϕ jointly?	Manual representation design may miss stronger or more efficient constructions.
Theory of representation gain	When does a richer input increase useful statistical evidence per unit cost?	Empirical gains need a clearer cryptanalytic explanation.
Cipher-family transferability	Which representation families transfer across ARX, SPN, and permutation-based designs?	Cipher-specific results may not generalize.
Interpretability	What cryptanalytic evidence is learned from multi-row or aggregated inputs?	Understanding learned signals improves trust and attack design.
Standard benchmarks	How should representation-rich distinguishers be compared reproducibly?	Current comparisons often mix representation, data, and architecture effects.
Key-recovery integration	Do richer representations improve full attacks, not just classifier accuracy?	Distinguishing performance alone is not sufficient for cryptanalytic relevance.

These open problems show that representation design is still an emerging part of neural differential cryptanalysis. Richer inputs can improve distinguishers, but the field needs better methods for designing them, explaining their gains, comparing their costs, and integrating them into complete attacks.

8 Conclusion

This paper surveyed rich input representations in neural differential cryptanalysis from a representation-centric viewpoint. Instead of treating the neural architecture as the only major design factor, we focused on how ciphertext-derived evidence is constructed, organized, and presented to the classifier. We argued that rich input representations are first-class cryptanalytic choices because they affect classification accuracy, data cost, difference selection, transferability, and attack integration.

To support this view, we introduced a framework that describes a representation by its difference set $\mathcal{D}$, number of observations per sample m, sharing structure s, encoding function ϕ, and ciphertext cost c. This framework provides common terminology for comparing representations beyond the single-pair baseline, including multi-pair, multi-difference, matrix-style, multi-round, structured-encoding, and score-aggregation based inputs. It also makes explicit that richer representations may increase the information content of each sample while simultaneously increasing the ciphertext cost required to construct that sample.

Our comparative literature map showed that the field has gradually moved from single-pair neural distinguishers toward richer and more structured input formats. These developments have improved neural distinguishers in several settings, but they also make evaluation more difficult. In many works, representation changes are combined with architecture changes, training refinements, dataset-construction choices, or attack-level modifications. As a result, it is often difficult to determine whether an observed improvement comes from the representation itself or from other components of the pipeline.

We therefore emphasized the need for cost-aware evaluation. Fixed-sample comparisons and fixed-ciphertext comparisons answer different questions: the former measures the information density of a labeled example, while the latter measures the efficiency of a representation under a limited cryptanalytic data budget. For representation-rich neural distinguishers, both viewpoints are important. Future work should report the input structure, the ciphertext cost per sample, the selected differences, the comparison model, and whether the distinguisher is evaluated as a standalone classifier or as part of a key-recovery attack.

Several open problems remain. The field needs automated methods for searching over representation components, theoretical explanations of representation gain, better understanding of transferability across cipher families, interpretability tools for learned multi-row evidence, standardized benchmarks, and more systematic integration of richer representations into full attacks. Addressing these issues would help move neural differential cryptanalysis from isolated representation improvements toward a more principled methodology for designing and evaluating rich neural distinguisher inputs.

References

[1] Anubhab Baksi, Jakub Breier, Xiaoyang Dong, and Chen Yi. Machine learning assisted differential distinguishers for lightweight ciphers. Cryptology ePrint Archive, Paper 2020/571, 2020.

[2] Zhenzhen Bao, Jian Guo, Meicheng Liu, Li Ma, and Yi Tu. Enhancing differential-neural cryptanalysis. In *Advances in Cryptology – ASIACRYPT 2022*, Lecture Notes in Computer Science, pages 318–347. Springer, 2022.

[3] Emanuele Bellini, David Gerault, Anna Hambitzer, and Matteo Rossi. A cipher-agnostic neural training pipeline with automated finding of good input differences. *IACR Transactions on Symmetric Cryptology*, 2023(3):184–212, 2023.

[4] Adrien Benamira, David Gerault, Thomas Peyrin, and Quan Quan Tan. A deeper look at machine learning-based cryptanalysis. In *Advances in Cryptology – EUROCRYPT 2021*, Lecture Notes in Computer Science, pages 805–835. Springer, 2021.

[5] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.

[6] Yi Chen, Yantian Shen, Hongbo Yu, and Sitong Yuan. A new neural distinguisher considering features derived from multiple ciphertext pairs. *The Computer Journal*, 66(6):1419–1433, 2023.

[7] Chuchu Ge and Qichun Wang. Improved related-key differential neural distinguishers for spn block ciphers. *Cryptology ePrint Archive*, 2026.

[8] Aron Gohr. Improving attacks on round-reduced SPECK32/64 using deep learning. In *Advances in Cryptology – CRYPTO 2019*, volume 11693 of *Lecture Notes in Computer Science*, pages 150–179. Springer, 2019.

[9] Aron Gohr, Gregor Leander, and Patrick Neumann. An assessment of differential-neural distinguishers. Cryptology ePrint Archive, Paper 2022/1521, 2022.

[10] Yufei Hou, Jie Liu, Shouxu Han, Zhongjun Ma, Xi Ye, and Xuan Nie. Improving deep learning-based neural distinguisher with multiple ciphertext pairs for SPECK and SIMON. *Scientific Reports*, 15:13696, 2025.

[11] ZeZhou Hou, JiongJiong Ren, and ShaoZhen Chen. Improve neural distinguishers of SIMON and SPECK. *Security and Communication Networks*, 2021:9288229, 2021.

[12] Yunjae Hwang, Sunyeop Kim, Hanbeom Shin, Deukjo Hong, Seokhie Hong, Dongjae Lee, Jaechul Sung, and Byoungjin Seok. Local constraints behind fourier analysis of neural distinguishers for speck32/64. *Cryptology ePrint Archive*, 2026.

[13] Jiashuo LIU, Manman LI, Jiongjiong REN, and Shaozhen CHEN. A highly efficient neural distinguisher framework for iot-friendly lightweight spn block ciphers. *IEICE Transactions on Information and Systems*, 2025.

[14] JiaShuo Liu, JiongJiong Ren, ShaoZhen Chen, and ManMan Li. Improved neural distinguishers with multi-round and multi-splicing construction. *Journal of Information Security and Applications*, 74:103461, 2023.

[15] Lijun Lyu, Yi Tu, and Yingjie Zhang. Improving the deep-learning-based differential distinguisher and applications to SIMECK. In *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 465–470. IEEE, 2022.

[16] Iman Mirzaali, Sattar Sadeghi, and Nasour Bagheri. Improved polytopic differential neural distinguishers for SIMON, SIMECK, and SPECK block ciphers. *Cybersecurity*, 9:70, 2026.

[17] Byoungjin Seok and Changhoon Lee. A novel approach to construct a good dataset for differential-neural cryptanalysis. *IEEE Transactions on Dependable and Secure Computing*, 22(1):246–262, 2025.

[18] Dongsu Shen, Yijian Song, Yuan Lu, Saiqin Long, and Shujuan Tian. Neural differential distinguishers for GIFT-128 and ASCON. *Journal of Information Security and Applications*, 82:103758, 2024.

[19] Tao Sun, Dongsu Shen, Saiqin Long, Qingyong Deng, and Shiguo Wang. Neural distinguishers on TinyJAMBU-128 and GIFT-64. In *Neural Information Processing*, volume 1792 of *Communications in Computer and Information Science*, pages 419–431. Springer, 2022.

[20] Gao Wang and Gaoli Wang. Keeping classical distinguisher and neural distinguisher in balance. *Journal of Information Security and Applications*, 84:103816, 2024.

[21] Gao Wang, Gaoli Wang, and Yu He. Improved machine learning assisted (related-key) differential distinguishers for lightweight ciphers. In *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 164–171. IEEE, 2021.

[22] Gao Wang, Gaoli Wang, and Siwei Sun. Investigating and enhancing the neural distinguisher for differential cryptanalysis. *IEICE Transactions on Information and Systems*, E107.D(8):1016–1028, 2024.

[23] Huijiao Wang, Jiapeng Tian, Xin Zhang, Yongzhuang Wei, and Hua Jiang. Multiple differential distinguisher of SIMECK32/64 based on deep learning. *Security and Communication Networks*, 2022:7564678, 2022.

[24] Tarun Yadav and Manoj Kumar. Differential-ML distinguisher: Machine learning based generic extension for differential cryptanalysis. In *Progress in Cryptology – LATINCRYPT 2021*, volume 12912 of *Lecture Notes in Computer Science*, pages 191–212. Springer, 2021.

[25] Tarun Yadav and Manoj Kumar. ML based improved differential distinguisher with high accuracy: Application to GIFT-128 and ASCON. In *Security, Privacy, and Applied Cryptography Engineering – SPACE 2024*, volume 15351 of *Lecture Notes in Computer Science*, pages 287–316. Springer, 2025.

[26] Xiaoteng Yue and Wanqing Wu. Improved neural differential distinguisher model for lightweight cipher SPECK. *Applied Sciences*, 13(12):6994, 2023.

[27] Liu Zhang, Jinyu Lu, Zilong Wang, and Chao Li. Improved differential-neural cryptanalysis for round-reduced SIMECK32/64. *Frontiers of Computer Science*, 17(6):176817, 2023.