# Breaking Indistinguishability with Transfer Learning: A First Look at SPECK32/64 Lightweight Block Ciphers

Jimmy Dani, Kalyan Nakka, Nitesh Saxena
Texas A&M University
College Station, TX
{danijy,kalyan,nsaxena}@tamu.edu

## ABSTRACT

In this research, we introduce MIND-Crypt[1], a novel attack framework that uses deep learning (DL) and transfer learning (TL) to challenge the indistinguishability of block ciphers, specifically SPECK32/64 encryption algorithm in CBC mode (Cipher Block Chaining) against Known Plaintext Attacks (KPA). Our methodology includes training a DL model with ciphertexts of two messages encrypted using the same key. The selected messages have the same byte-length and differ by only one bit at the binary level. This DL model employs a residual network architecture. For the TL, we use the trained DL model as a feature extractor, and these features are then used to train a shallow machine learning, such as XGBoost. This dual strategy aims to distinguish ciphertexts of two encrypted messages, addressing traditional cryptanalysis challenges.

Our findings demonstrate that the deep learning model achieves an accuracy of approximately 99% under consistent cryptographic conditions (Same Key or Rounds) with the SPECK32/64 cipher. However, performance degrades to random guessing levels (50%) when tested with ciphertext generated from different keys or different encryption rounds of SPECK32/64. To enhance the results, the DL model requires retraining with different keys or encryption rounds using larger datasets ($10^7$ samples). To overcome this limitation, we implement TL, achieving an accuracy of about 53% with just 10,000 samples, which is better than random guessing. Further training with 580,000 samples increases accuracy to nearly 99%, showing a substantial reduction in data requirements by over 94%. This shows that an attacker can utilize machine learning models to break indistinguishability by accessing pairs of plaintexts and their corresponding ciphertexts encrypted with the same key, without directly interacting with the communicating parties.

## 1 INTRODUCTION

Indistinguishability is the basis for building secure encryption systems. Concretely, indistinguishability means that the adversary can not tell the difference between the ciphertexts corresponding to two plaintexts with a probability significantly better than 0.50. It is an important notion underlying encryption security since it implies that the adversaries are unable to decipher any useful information about the plaintext given the ciphertext. Moreover, a broken indistinguishability property exposes deterministic or predictable patterns in the encryption process, making the system susceptible to more effective attacks, such as ciphertext-only attacks where the plaintext is deciphered without the key. This not only undermines the trust and reliability of the cryptographic system but also paves the way for practical decryption techniques that could exploit this predictability. Therefore, preserving indistinguishability is essential to maintain the overall integrity and security of encryption schemes.

**Lightweight Block Ciphers.** The Internet of Things (IoT) exemplifies a domain where cryptography's vital role is particularly pronounced, due to its explosive growth and the evolving capabilities of connected devices. With projections estimating over 75 billion devices connected by 2025 [18], the diversity of applications—from smart home devices enhancing residential convenience and security, to advanced systems in healthcare monitoring and industrial IoT (IIoT)—is transforming traditional industries. However, many IoT devices operate under constraints of processing power and memory, necessitating cryptographic solutions that optimize security without imposing significant computational burdens. Among lightweight block ciphers, the SPECK32/64 cipher, designed by the National Security Agency (NSA), stands out for its operational efficiency and simplicity, tailored specifically to meet the needs of these resource-constrained environments [1, 9].

**Cryptanalysis and Machine Learning.** As cryptographic systems evolve in complexity and sophistication, so too does cryptanalysis – the study and practice of deciphering codes, ciphers, and encrypted messages without the use of actual key. This discipline has seen significant advancements through a variety of techniques, reflecting the ongoing arms race between cryptography and cryptanalysis. Traditional methods such as side-channel attacks [27, 33, 34, 46], fault injection attacks [11, 19, 21, 36], mathematical analysis [6, 29, 43], and brute-force attacks [23, 31, 39, 42] have continually been refined in tandem with advancements in cryptographic techniques. However, as cryptographic algorithms become more complex, the effectiveness of these traditional approaches is increasingly challenged, necessitating newer methodologies. This evolving landscape has sparked considerable interest in integrating machine learning with cryptanalysis, offering novel approaches to breaking cryptographic systems and presenting new challenges to their robustness.

In 2019, Gohr [22] proposed a differential attack on round-reduced SPECK32/64, focusing on the development of neural distinguishers that could effectively distinguish ciphertexts differing by a specific difference delta from random text. This approach leveraged DL, specifically deep residual neural networks, which demonstrated superior performance compared to traditional cryptographic distinguishers. Further enhancing the practicality of his method, Gohr integrated a novel key search policy based on Bayesian optimization, significantly improving the efficiency of key recovery processes. Following Gohr's work, Benamira et al. [14] conducted detailed analysis and showed neural distinguisher developed by Gohr generally relies on the differential distribution on the ciphertext pairs, but

[1]We refer to our attack framework as MIND-Crypt, which stands for "***M***achine learning based attack framework against ***IND***istinguishability of ***Crypt***ographic Algorithms."

also on the differential distribution in penultimate and antepenultimate rounds. This approach not only showcased DL's potential in enhancing traditional cryptanalysis but also emphasizes the need to probe deeper into the cipher's behavior by exploring the notion of indistinguishability.

**Focus of Our Research.** In contrast, our research shifts the focus from differential attack strategies to breaking the broader concept of indistinguishability within SPECK32/64. Unlike Gohr's approach, which targets specific, known differential paths for key recovery, our study employs DL and TL to assess whether a model can distinguish between ciphertexts encrypted under different configurations of keys and rounds. Crucially, this approach highlights the attacker's significant capability to train robust models using self-generated datasets, thereby reducing reliance on extensive victim data. By needing only a small amount of actual ciphertext from the victim, our method showcases a powerful, scalable attack model that efficiently breaches cryptographic security. This analysis is pivotal in evaluating the cipher's robustness against a more comprehensive range of adaptive threats, thereby addressing a fundamentally different and more crucial aspect of cryptographic security.

Formally, in our study, we address the following research question: *Can machine learning techniques be employed to break the indistinguishability of lightweight block ciphers (e.g., SPECK32/64), challenging not only their implementation but also the very foundations of the algorithms themselves?* We answer this question affirmatively by introducing MIND-Crypt, an attack based on deep learning (DL) and transfer learning (TL) that efficiently distinguishes ciphertexts of two messages encrypted with the same key.

When designing MIND-Crypt, we considered assumptions typical of Known Plaintext Attack (KPA) scenario, where the attacker has access to both plaintexts and their corresponding ciphertexts encrypted under the same key. Our primary focus is on the ability of the attacker to differentiate ciphertexts that correspond to different plaintext messages, assessing the indistinguishability of the SPECK32/64 cipher in a controlled environment that simulates potential real-world attack scenarios on IoT devices. In our study, we focus on both its standard 22-round configuration and round-reduced versions to understand how these variations affect its resistance to cryptographic attacks.

**Our Methodology & Experiments.** We approach this challenge by framing the task as a binary classification problem, where the machine learning (ML) classifier is trained on previously-known ciphertexts $\mathcal{C}_1$ and $\mathcal{C}_2$ corresponding to two fixed plaintexts $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, and using the trained model to predict whether any new challenge ciphertexts correspond to $\mathcal{P}_1$ or $\mathcal{P}_2$. To train the model, the attacker generates ciphertexts of these messages by encrypting them using SPECK32/64 under the same key. This scenario is realistic in many practical applications and provides the necessary data (ciphertexts) for training the ML model. We hypothesize that ML algorithms can detect patterns and dependencies in ciphertexts, achieving an accuracy significantly better than random guessing.

In order to enhance the effectiveness of our attack, we leverage transfer learning, a technique that significantly amplifies the capabilities of attackers, particularly in symmetric key contexts. Traditionally, attackers are constrained by the limited availability of ciphertexts, often requiring unauthorized access to data from potential victims. Transfer learning modifies this constraint by allowing attackers to initially train models on extensive datasets generated autonomously. These models are subsequently fine-tuned using smaller, specific datasets that mirror the operational conditions of the victim's environment. This methodology not only facilitates sophisticated cryptanalytic efforts but also establishes TL as a formidable tool in the cryptographic landscape, thereby increasing the viability of deploying machine learning techniques in cryptanalysis.

To empirically validate our approach, we conducted four experiments under different cryptographic conditions: 1) Same Rounds, Same key, where the model showed high accuracy, indicating a vulnerability in configurations where both rounds and key are consistent; 2) Same Rounds, Different Key, which tested the model's robustness against key variability with a notable drop in performance; 3) Different Rounds, Same Key, revealing the cipher's effectiveness in mitigating risks through varying rounds; and 4) Different Rounds, Different Key, the most challenging scenario, resulting in the lowest accuracy, underscoring the cipher's strong defense against attacks with no consistent cryptographic settings. These results highlight the nuanced vulnerabilities of ciphers and underscore the potential of deploying advanced machine learning techniques in cryptanalysis under diverse operational conditions.

Our results show that the DL model achieved 99% accuracy when both training and evaluation used ciphertexts encrypted under the same key or number of rounds, with a training set of $10^7$ samples. However, when the model was evaluated using a challenge ciphertexts with a different key or number of rounds than those used in training, its accuracy plummeted to 50%, equivalent to random guessing. This drop highlights the model's sensitivity to changes in cryptographic configurations and the substantial data requirements needed to retrain the model for each new configuration.

To address these challenges, we implemented a TL approach that significantly reduced the dependency on large datasets. By reusing a pre-trained deep learning model as a feature extractor and training a lightweight machine learning classifier like XGBoost on just 10,000 samples, we enhance the model's accuracy to approximately 53% under new cryptographic conditions. This represents a significant improvement over random guessing. Further, when the sample size was increased to 580,000, the accuracy of the transfer learning approach reached 99%, matching the performance of the original DL model but with a substantial reduction in the number of required samples. Specifically, this approach represents a data reduction of over 94%, demonstrating the efficacy of transfer learning in making cryptanalysis techniques more data-efficient and adaptable. This unsettling discovery underscores the critical significance (or the lack of) indistinguishability and highlights the potential vulnerabilities in a lightweight block ciphers deployed in resource constrained environments. The work can be similarly extended to analyze the indistinguishability of other block ciphers, public key encryption schemes and even post-quantum crypto algorithms.

**Our Contributions and Summary of Results:** The main contributions and findings are summarized as follows:

(1) ***A Novel Attack Framework for Breaking Indistinguishability:*** We introduce MIND-Crypt, a novel attack framework that utilized various machine learning techniques to investigate the

indistinguishability of lightweight block ciphers, SPECK32/64. More specifically, we leverage deep learning and transfer learning techniques to implement MIND-Crypt.

(2) ***A Novel Threat Model:*** As a part of MIND-Crypt design, we introduce a novel threat model to assess the indistinguishability of the SPECK32/64 lightweight block cipher in CBC mode under passive Known Plaintext Attacks (KPA), focusing on IoT devices. In our threat model, the attacker collects large number of ciphertexts of messages $\mathcal{P}_1$ and $\mathcal{P}_2$ to train the DL model. Later, attacker uses this trained DL model to determine if the challenge ciphertext obtained from victim belongs to message $\mathcal{P}_1$ or $\mathcal{P}_2$.

(3) ***Enhanced Cryptographic Analysis through Transfer Learning:*** Our methodology demonstrates how transfer learning can effectively reduce the reliance on large training datasets typically required for cryptanalysis under varied encryption settings. By employing a pretrained deep learning model as a feature extractor, followed by training a shallow machine learning model like XGBoost, we achieve substantial improvements in model adaptability and performance. We documented a reduction in data needs by over 94%, illustrating how our approach can achieve high accuracy (matching that of extensively trained deep learning models) with significantly fewer data samples. This contribution is particularly valuable where data availability and computational resources are often limited.

(4) ***Comprehensive Evaluation Across Cryptographic Settings:*** We thoroughly evalute the proposed attack in various cryptographic configurations, demonstrating robustness of the proposed framework. We trained and evaluated our models across four distinct settings—same rounds with the same key, same rounds with a different key, different rounds with the same key, and different rounds with a different key—providing a comprehensive assessment of model performance under changing conditions. This extensive evaluation demonstrates the models' adaptability and effectiveness in distinguishing ciphertexts under varied encryption scenarios.

**Reproducibility.** To reproduce our findings, we will make code and datasets publicly available upon the publication of this research.

## 2 BACKGROUND & PRELIMINARIES

**SPECK32/64 Block Cipher.** SPECK is a family of lightweight block ciphers, denoted as SPECK$M/N$ where $M, N$ are block size and key size respectively in bits, developed by Beaulieu, Treatman-Clark, Shors, Weeks, Smith and Wingers [13] for NSA. It is an add-rotate-xor (ARX) cipher with operations like modular addition (mod $2^k$) $\boxplus$, bitwise addition $\oplus$, and bitwise rotation (left $\ll$ and right $\gg$) applied on $k$-bit words, aimed to build efficient cipher for software implementations in IoT devices [12]. The round function of SPECK $F : \mathbb{F}_2^{2k} \times \mathbb{F}_2^{2k} \rightarrow \mathbb{F}_2^{2k}$, computes the next round state $(L_{i+1}, R_{i+1})$ using a $k$-bit subkey $K$ and current round state $(L_i, R_i)$ as, $L_{i+1} = ((L_i \gg \alpha) \boxplus R_i) \oplus K$, and $R_{i+1} = (R_i \ll \beta) \oplus L_{i+1}$. Here, $\alpha, \beta$ are rotation constants ($\alpha = 7, \beta = 2$ for SPECK32/64 and $\alpha = 8, \beta = 3$ for remaining). The ciphertext is produced from the input plaintext by employing this round function for a fixed number of times (22 rounds for SPECK32/64).

**Residual Neural Networks (ResNets).** He et al. [25] introduced ResNets to address the vanishing gradients problem in deep neural network (DNN) training by utilizing residual blocks. These blocks, featuring stacked convolutional layers with skip connections, allow the network to learn residual functions, focusing on differences rather than complete transformations. ResNets have been successfully applied in various security applications [3, 35, 38, 40].

In cryptanalysis, ResNets model complex patterns effectively, aiding in tasks like automated cipher breaking and differential cryptanalysis. Their architecture allows for more accurate and efficient prediction of differential characteristics, enhancing encryption analysis and vulnerability insights. Adrien et al. [14] discuss how machine learning, including ResNets, advances cryptanalytic and cyber defense techniques.

**Transfer learning (TL).** ML tasks often face challenges with limited datasets for specific problems. TL addresses these by leveraging knowledge from related tasks. This approach uses a pre-trained model, initially trained on a vast dataset for a different task, as a feature extractor. This model captures essential representations, which are then utilized to train a new, shallow model tailored for the target task.

In our research, we leverage transfer learning with feature extraction to enhance the performance of our binary classification task. We employ a pre-trained deep residual network (ResNet) as a feature extractor. In TL, the DL models are originally trained on a large dataset for a different task, such as image classification. By focusing on the early layers of the ResNet, known to capture more generic features, we extract informative representations from the data [44]. These features are then fed into a separate, shallow model specifically designed for our binary classification problem. This two-stage approach allows the XGBoost model to concentrate on learning the relationships between the extracted features and the target classes, ultimately leading to potentially improved performance and faster training times [15, 32, 41, 47].

## 3 THREAT MODEL & ASSUMPTIONS

Our study investigates the security of the SPECK32/64 lightweight block cipher in CBC mode (Cipher Block Chaining) against Known Plaintext Attacks (KPA). We primarily focus on an attacker's ability to distinguish between the ciphertexts of two different messages encrypted using SPECK32/64 under the same key for IoT devices that operate under significant energy constraints and require efficient cryptographic solutions, such as SPECK 32/64 cipher.

In our attack model, we consider passive attack scenario where the attacker gains excess ciphertexts, all encrypted with the same key, without performing active attacks such as Chosen-Ciphertext Attacks (CCA). For instance, the attacker might observe ciphertexts corresponding to specific actions, like "buying" or "selling" stocks, and confirm these actions through public outcomes, collecting pairs of plaintexts and ciphertexts. Our model extends these concepts by allowing the attacker to simulate data generation without direct access, avoiding the active manipulation typical of CCA. The attacker aims to identify patterns, anomalies, or relationships in the ciphertexts that differentiate those corresponding to two distinct, same-byte-length plaintexts.

Mathematically, we denote the plaintext by $\mathcal{P}$, the ciphertext by $\mathcal{C}$, and the secret key by $\mathcal{K}$. The encryption function $\mathcal{E}_K$ uses the key $\mathcal{K}$ to transform plaintext into ciphertext. A cipher maintains indistinguishability if no polynomial-time adversary can distinguish between the ciphertexts of two different plaintexts encrypted with the same key with a probability significantly better than 0.5.

The attacker selects two different fixed plaintexts, $\mathcal{P}_1$ and $\mathcal{P}_2$ (e.g., "buying" or "selling"), which are encrypted using the same secret key $\mathcal{K}$, resulting in ciphertexts $\mathcal{C}_1$ and $\mathcal{C}_2$. Subsequently, the attacker employs a deep learning model, trained with multiple instances of ciphertexts $\mathcal{C}_1$ and $\mathcal{C}_2$. This model is then utilized to classify new challenge ciphertexts, determining whether they correspond to $\mathcal{P}_1$ or $\mathcal{P}_2$, potentially breaching the indistinguishability property of the encryption scheme. Differentiating ciphertexts beyond chance agreement signifies vulnerabilities in the block cipher, whereas failure to do so would validate the cipher's robustness under passive attack setting.

The application of ML models to this problem introduces a novel threat models. By generating independent datasets, attackers can train robust models to enhance their capability of distinguishing ciphertexts beyond random chance. This scenario contrasts with traditional models where attackers rely directly on victim data. Our results demonstrate that ML can significantly impact the security of lightweight ciphers and reveal potential vulnerabilities in SPECK32/64 cipher under KPA, highlighting the need for robust cryptographic defenses against such passive attacks. These threat models are described as follows:

(1) ***Threat model 1 (DeepAnalyst):*** In this model, the attacker generates a large dataset of ciphertexts of $\mathcal{P}1$ and $\mathcal{P}2$ and trains a DL model. The attacker then collects ciphertexts from the victim and uses the trained DL model to identify if the ciphertext belongs to $\mathcal{P}1$ or $\mathcal{P}2$. If the encryption key or the number of encryption rounds changes, the DL model must be retrained with a large amount of data to accurately distinguish the ciphertexts. To overcome this data requirement, we propose threat model 2 (TransferAnalyst).
(2) ***Threat model 2 (TransferAnalyst):*** In this model, the attacker generates a large dataset of ciphertexts of $\mathcal{P}1$ and $\mathcal{P}2$ and trains a DL model. When the encryption key or number of encryption rounds changes, the attacker uses the trained DL model as a feature extractor for new ciphertext pairs of messages $\mathcal{P}1$ and $\mathcal{P}2$ generated under a different key or round. The attacker then trains a shallow ML model (such as XGBoost), which requires significantly less data. Later, the attacker uses the trained shallow ML model to distinguish the ciphertexts of messages $\mathcal{P}1$ and $\mathcal{P}2$.

## 4 MIND-CRYPT DESIGN & METHODOLOGY

In this section, we provide detailed introduction to the ciphertext indistinguishability attack MIND-Crypt. We first describe the procedure for message selection, and encrypting a message for generating ciphertext. Following this,we describe the training process of a deep learning model specifically designed to differentiate between these ciphertexts in threat model 1 (DeepAnalyst). Finally, we explain how this trained model serves as a feature extractor for applying transfer learning in threat model 2 (TransferAnalyst).

In this study, we introduce a DL based attack designed to distinguish between the ciphertexts originating from two distinct, same-byte length messages $\mathcal{P}_1$ and $\mathcal{P}_2$. These messages undergo encryption using SPECK32/64 block cipher with CBC mode of operation. Our DL model is trained for binary classification task of separating ciphertexts into two classes: $\xi_1$ and $\xi_2$. To explain, $\xi_1$ includes the ciphertexts of $\mathcal{P}_1$, labeled as $\mathcal{C}1_i$ ($\mathcal{C}1_i = Enc_k(\mathcal{P}_1)$), collected across a range of indices $i \in \{1, 2, \ldots, n\}$. Similarly, $\xi_2$ includes the ciphertexts of $\mathcal{P}_2$, labeled as $\mathcal{C}2_i$ ($\mathcal{C}2_i = Enc_k(\mathcal{P}_2)$) for $i \in \{1, 2, \ldots, n\}$.

Figure 1 illustrates our attack framework, outlining a detailed process that begins with the collection and preparation of data. We represent this data as binary vectors, in line with recommendations by Gohr et al. [22]. Initially, we select two messages of identical byte length but differ at a single binary bit. These messages undergo multiple encryption cycles using the same key, utilizing the SPECK32/64 block cipher algorithm in CBC mode.

After encryption, we convert the ciphertexts into binary format, as proposed by Gohr et al. [22] for examining differential attacks on SPECK32/64. Utilizing this methodology, we feed these binary ciphertexts into a DL model, originally designed by Gohr [22] for differential attack analysis. Unlike Gohr's application, we adapt this model to study the indistinguishability for SPECK32/64. The DL model comprises an initial 1D convolution layer with batch normalization and ReLU activation in Block 1, followed by Block 2 with two convolution layers, each complemented by batch normalization and ReLU activation. The final segment (Block 3) responsible for classification includes three fully-connected (FC) layers, interspersed with batch normalization and ReLU activations, concluding with a sigmoid function. The architecture of the DL model, alongside the feature extractor with the shallow ML model for the TL approach, is depicted in Figure 3 of Appendix C. The input layer of the model receives vectors representing ciphertexts of $\mathcal{P}_1$ and $\mathcal{P}_2$. The ciphertexts pass through a series of convolution operations in Block 1. Block 2 includes residual connections, which facilitate the training of deeper models by preserving the gradients [25]. These connections enable the network to focus on learning residual mappings rather than direct mappings. For DeepAnalyst threat model, this DL model is used to distinguish the challenge ciphertexts acquired from the victim.

For TransferAnalyst threat model, which uses transfer learning for indistinguishability, we deploy the DL model solely as a feature extractor. During this process, the weights of the layers involved in extracting features are frozen. This setup is depicted in Figure 1. These extracted features were then used to train an XGBoost classifier, chosen for its robustness and efficiency in handling structured data. XGBoost is a gradient boosting framework that builds an ensemble of weak learners, typically decision trees, to create a strong classification model, categorizing the ciphertexts into classes $\xi_1$ and $\xi_2$. Notably, this transfer learning approach allows us to enhance the accuracy of the classification with fewer data points compared to using the feature extractor of the DL model. This combination of DL-based feature extraction and XGBoost classification aimed to enhance the model's ability to distinguish between ciphertexts with fewer data points compared to the standalone DL model. This

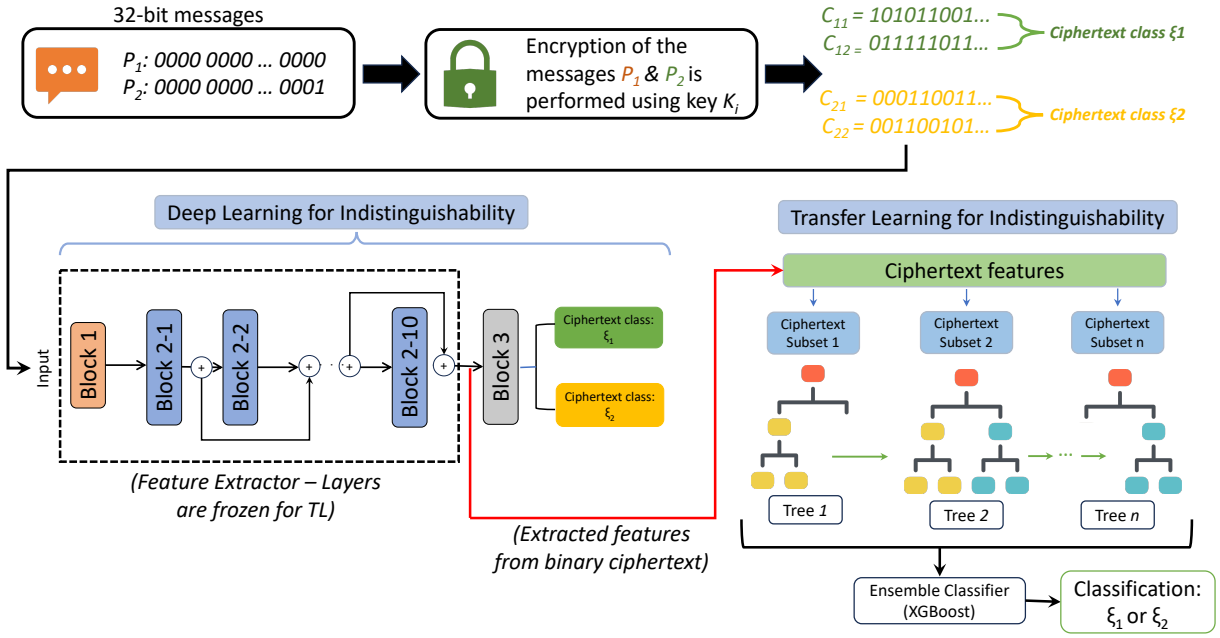


Figure 1: The MIND-Crypt attack framework - Investigating the indistinguishability of SPECK32/64 block ciphers. Two plaintext messages encrypted with the same key, represented in binary format, form the basis for training and evaluating a DL model. To overcome limitations observed in DL, we apply transfer learning, utilizing the DL as a feature extractor whose output features are then classified using an XGBoost model. This hybrid approach leverages the strengths of both deep learning and machine learning to enhance classification accuracy between ciphertext classes $\xi_1$ and $\xi_2$ with fewer data samples required for training.

hybrid approach capitalized on the strengths of both deep learning and gradient boosting techniques, providing a comprehensive analysis of the cipher's security under the KPA attack model.

## 5 EVALUATION FRAMEWORK

In this section, we define the framework to evaluate the efficacy of MIND-Crypt in distinguishing ciphertexts of the two messages. We describe the datasets, experiment settings, and evaluation metric considered for evaluating our attack.

### 5.1 Description of the Dataset

In our study, we evaluated the effectiveness of the MIND-Crypty utilizing a publicly available implementation of SPECK32/64 provided by Gohr et al. [22]. Our objective was to investigate the principle of indistinguishability, which required control over the inputs provided for encryption. To this end, we made several modifications to the original SPECK32/64 implementation provided in [22]:

(1) *Message Selection:* We chose two specific messages of identical 32-bit length, differing by only a single bit at the binary level, labeled '0' and '1'. This allowed us to directly assess the effect of minimal input variation on the encryption output.
(2) *Encryption Mode:* We shifted from the ECB mode used in Gohr's original code to CBC mode. This change involved encrypting the messages using CBC with randomly generated initialization vectors (IVs) and applying an XOR operation to the messages before encryption.
(3) *Key Usage:* Unlike the original implementation that used varying keys, we utilized a single, fixed key that was securely generated using Gohr's methodology. This consistency was vital for comparing the indistinguishability of outputs. The specific keys used for our study are provided in Appendix A (Table 7).
(4) *Correctness:* To ensure the correctness of our modifications, we decrypted the ciphertexts to verify that they reverted accurately to the original plaintexts, '0' and '1'.

Additionally, for exploring indistinguishability using DL, we collected $10^7$ training samples, and $10^6$ testing samples across '$\mathcal{R}$' rounds of SPECK32/64 encryption scheme. The training data was used to train a DL model, while the testing data was utilized to evaluate the performance of the trained model on an unseen dataset. This allowed the DL model to detect subtle differences in ciphertexts of the selected messages. To facilitate the learning process, the ciphertexts were represented as 32-bit binary vectors, providing a consistent input format for the DL.

### 5.2 Experiment Settings

In this study, the implementation of the MIND-Crypt was done using the Python programming language along with three open-source libraries: `TensorFlow` [2], eXtreme Gradient Boosting (`XGBoost`), and `Optuna` [5]. `TensorFlow` was used for training and evaluating the DL model, `XGBoost` for training the shallow ML algorithm under the Gradient Boosting framework, and `Optuna` for automating hyperparameter optimization for the ML models.

**DL Model & Feature Extractor training.** We utilized DL model developed by Gohr et al. [22] to classify ciphertexts of two messages

$\mathcal{P}_1$ and $\mathcal{P}_2$ generated by encrypting them using same key. We used a supervised learning approach, where the DL model learns to distinguish between ciphertexts. The architecture of the DL model, as detailed in [22], consists of three types of blocks, each containing specific layers and operations. These blocks are designed to process input data through a series of convolution, normalization, and activation layers. The input layer of the model is provided with vectors representing ciphertexts of $\mathcal{P}_1$ and $\mathcal{P}_2$. For more details on model training and hyperparameter selection, refer to [22].

**XGBoost training.** To leverage the learned features from the DL model for further analysis, we employed transfer learning. The penultimate layer, specifically the `Flatten()` layer, of the trained DL model was used as a feature extractor. This layer converts the multidimensional output from the convolutional layers into a one-dimensional feature vector, capturing essential patterns and relationships identified during training. For training the XGBoost classifier, we used the extracted features from the DL model as input. The classifier was trained on the same dataset of ciphertexts, with the labels indicating whether the ciphertext corresponded to $\mathcal{P}_1$ or $\mathcal{P}_2$. We performed hyperparameter tuning for the `XGBoost` classifier using Optuna, an automatic hyperparameter optimization framework, to identify the optimal parameters. The search space for the hyperparameters is detailed in Table 8 in Appendix D.

## 5.3 Evaluation Scenarios

Figure 2 provides a comprehensive matrix view of the four distinct evaluation settings: "Same Key, Same Round," "Different Round, Same Key," "Same Round, Different Key," and "Different Round, Different Key." Each cell in the matrix in Figure 2 corresponds to a specific scenario, illustrating the configuration used for testing the model's performance and generalizability across different encryption keys and rounds.

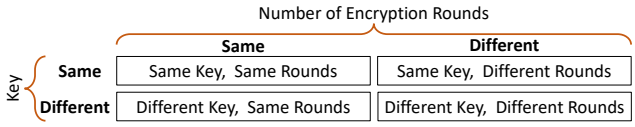


Figure 2: Four testing configurations used to assess the performance of MIND-Crypt using deep learning and transfer learning: 1) Same Key, Same Round; 2) Different Round, Same Key; 3) Same Round, Different Key; and 4) Different Round, Different Key. Each scenario tests the model's ability to generalize across changes in encryption keys and rounds.

**Same Rounds, Same Key.** In this setting, we trained the DL model on the ciphertexts of the two messages encrypted under key $\mathcal{K}_1$ for $\mathcal{R}$ rounds of SPECK32/64. This trained model is used for distinguishing the unseen ciphertexts generated under identical configurations - the same key and the same number of rounds - ensuring a fair assessment of the attack. Meaning that the ciphertexts used to train the classifier were encrypted using key $\mathcal{K}_1$ for $\mathcal{R}$ number of rounds, and the challenge ciphertexts used for attack evaluation were also encrypted using the same key $\mathcal{K}_1$ for $\mathcal{R}$ number of rounds.

**Same Round, Different Key.** In this setting, we evaluate the performance of MIND-Crypt using a DL model trained on ciphertexts from two messages encrypted under key $\mathcal{K}_1$. The objective of this sub-experiment is to assess the model's ability to generalize across different encryption keys while maintaining the same encryption rounds. We then tested this model with challenge ciphertexts from the same two messages but encrypted under a different key, $\mathcal{K}_2$. It should be noted that both training and evaluation phases used the same number of encryption rounds, $\mathcal{R}$, ensuring consistency across the experiment.

**Different Round, Same Key.** In this sub-experiment, we evaluate the performance of MIND-Cryptusing a DL model that was initially trained on ciphertexts from two messages encrypted with key $\mathcal{K}_1$ for $\mathcal{R}$ rounds of SPECK32/64. The main objective is to determine the model's ability to adapt and maintain accuracy when the complexity of the encryption increases, albeit under the same encryption key. For this purpose, we test the model using challenge ciphertexts that are also encrypted under the same key, $\mathcal{K}_1$, but with an increased or decreased number of encryption rounds, specifically $\mathcal{R} \pm 1$. This approach allows us to isolate the impact of increased or decreased encryption rounds on model performance without the additional variable of a different encryption key. It provides insights into how well the model can generalize its learned patterns to more complex encryption scenarios while retaining the same underlying key.

**Different Round, Different Key.** In this sub-experiment, we assessed the performance of MIND-Crypt using a DL model trained on ciphertexts encrypted with $\mathcal{K}_1$ for $\mathcal{R}$ rounds of SPECK32/64. The objective was to evaluate the adaptability and robustness of the model when the key changes and encryption complexity are increased (number of rounds). We tested this model using challenge ciphertexts encrypted under a different key, $\mathcal{K}_2$, for $\mathcal{R} \pm 1$ rounds. This setup aimed to challenge the model with new encryption keys and an additional round of encryption to understand its limitations and capabilities in varied settings.

## 5.4 Evaluation Metrics

To evaluate the efficacy of the MIND-Crypt across different settings, we performed a comprehensive assessment using a DL model to classify ciphertexts into two distinct classes, $\xi_1$ and $\xi_2$. This evaluation employs three key metrics: Accuracy, True Positive Rate (TPR), and True Negative Rate (TNR), similar to the metrics considered in studies by [14, 22] that explore differential attacks in the SPECK32/64 encryption scheme.

Accuracy gauges the model's overall effectiveness at correctly classifying ciphertexts belonging to class $\xi_1$ or $\xi_2$. We calculate accuracy as the proportion of correct classification—both true positives and true negatives—out of the total ciphertexts examined. High accuracy indicates the model's competence in consistently distinguishing between ciphertexts derived from the SPECK32/64 encryption algorithm, under variations in keys or encryption rounds.

True Positive Rate (TPR), or sensitivity, specifically measures the model's precision in identifying ciphertexts that genuinely belong to class $\xi_1$. This metric is crucial for cryptographic applications as it reflects the model's ability to capture the unique characteristics expected from $\xi_1$ under particular encryption conditions. High TPR is vital, especially in situations where failing to correctly identify a ciphertext from $\xi_1$ could pose significant security threats.

True Negative Rate (TNR), or specificity, evaluates the model's accuracy in classifying ciphertexts into class $\xi_2$ when they do not

belong to class $\xi_1$. This measure is essential for ensuring the model effectively identifies ciphertexts that do not adhere to the characteristics of class $\xi_1$, thus preventing false positives. A high TNR underscores the model's reliability in excluding non-conforming encryption outputs, pivotal for upholding robust cryptographic defenses.

By considering these metrics evaluation, we establish a balanced framework to rigorously assess the model's capability to differentiate between the classes $\xi_1$ and $\xi_2$. This comprehensive approach not only measures the model's accuracy but also its sensitivity and specificity, providing a holistic view of its performance in the context of cryptographic indistinguishability.

## 6 EVALUATION & RESULTS

**Experiment A - DL for attacking Indistinguishability.** In this experiment, we evaluate the efficacy of MIND-Crypt using the DL model developed by Gohr et al. [22] to determine its ability to distinguish ciphertexts generated from two messages encrypted under the same key, $\mathcal{K}_1$ in DeepAnalyst threat model. The experiment is conducted under four distinct settings as discussed in Section 5.3. The objective is to determine if the model could identify distinguishing features inherent to the ciphertexts that depend on the underlying encrypted messages.

**Same Round, Same Key.** Table 1 shows the performance of our attack in these settings. It was observed that when the DL model is evaluated with a challenge ciphertexts generated using $\mathcal{K}_1$ (the same key as the model was trained on), the model can distinguish them with significantly higher across four different rounds. We considered two rounds, 5 and 6, from the initial rounds of SPECK32/64, and we considered the last two rounds, 21 and 22, for this experiment. Additionally, TPR and TNR across different rounds are also significantly higher. These results suggest potential vulnerabilities in the indistinguishability of SPECK32/64 under "Same Round, Same Key" settings. These findings indicate that cryptographic configurations of SPECK32/64 may not sufficiently obscure differences between ciphertexts under MIND-Crypt.

**Same Round, Different Key.** Table 1 details the performance of our attack in the "Same Rounds, Different Key" setting. When the DL model is evaluated with ciphertexts generated under a different key from that used for training, its performance dramatically declines to random guessing. This significant drop in accuracy underscores the challenges in achieving model generalizability across different encryption keys.

Table 1: Performance of MIND-Crypt using DL in distinguishing ciphertexts in **'Same Round, Same Key'** and **'Same Round, Different Key'** settings for different rounds of SPECK32/64.

| | Same Key | | | Different Key | | |
|---|---|---|---|---|---|---|
| **No. of Rounds** | **Accuracy** | **TPR** | **TNR** | **Accuracy** | **TPR** | **TNR** |
| **5** | 0.9944 | 0.9933 | 0.9954 | 0.5001 | 0.5158 | 0.4845 |
| **6** | 0.9907 | 0.9885 | 0.9928 | 0.5038 | 0.4903 | 0.5172 |
| **21** | 0.9923 | 0.9929 | 0.9918 | 0.4994 | 0.5089 | 0.4899 |
| **22** | 0.9915 | 0.9906 | 0.9924 | 0.4999 | 0.5037 | 0.4960 |

**Different Round, Same Key.** Table 2 shows the performance of our attack in the "Different Round, Same Key" settings, both for increased or decreased rounds of encryption. The results show that when the DL model, initially trained on ciphertexts encrypted with $\mathcal{K}_1$ for $\mathcal{R}$ rounds of SPECK32/64, is subsequently evaluated on ciphertexts generated with the same key $\mathcal{K}_1$ but for an increased number of rounds, $\mathcal{R}+1$, the performance of the model decreases significantly to random guessing.

Conversely, the model shows consistent challenges when evaluated on fewer rounds than it was trained on. For instance, when trained on 6 rounds and tested on 5, the accuracy remains near the baseline chance at 0.5016, with TPR and TNR closely mirroring each other at 0.5046 and 0.4985. A similar pattern is observed when the model is trained on 22 rounds and evaluated on 21 rounds, achieving an accuracy of 0.5015 with TPR and TNR at 0.5060 and 0.4970, respectively.

These observations underscore a significant adaptability issue within the model: its performance approaches to random guessing in both scenarios of increased and decreased encryption rounds. This suggests that the model is highly sensitive to conditions under which it was trained and struggles with any deviation from this training scenario.

Table 2: Performance of MIND-Crypt using DL in distinguishing ciphertexts of two messages in **'Different Round, Same Key'** settings for different rounds of SPECK32/64.

| **No. of Rounds - $\mathcal{R}$ (Training)** | **Accuracy (Round $\mathcal{R}$)** | **No. of Rounds - $\mathcal{R}+1$ (Evaluation)** | **Accuracy (Round $\mathcal{R}+1$)** | **TPR** | **TNR** |
|---|---|---|---|---|---|
| 5 | 0.9944 | 6 | 0.4990 | 0.4927 | 0.5052 |
| 21 | 0.9907 | 22 | 0.4999 | 0.5107 | 0.4891 |
| 6 | 0.9923 | 5 | 0.5016 | 0.5046 | 0.4985 |
| 22 | 0.9915 | 21 | 0.5015 | 0.5060 | 0.4970 |

**Different Round, Different Key.** Table 3 show the performance of our attack in the "Different Round, Different Key" settings. Specifically, we first trained the model on ciphertexts generated using $\mathcal{K}_1$ for 5 rounds and then evaluated on ciphertexts generated under $\mathcal{K}_2$ for 6 rounds. Similarly, another evaluation was conducted when the model trained on ciphertexts for 21 rounds was tested on ciphertexts for 22 rounds. In each case, when the DL model faced both a new key and an additional round of encryption, its performance significantly declined to random guessing.

This substantial decrease in model performance highlights the challenges in maintaining efficacy when both encryption rounds and keys vary, indicating challenges in the generalizability of the DL model under varied cryptographic conditions.

Table 3: Performance of MIND-Crypt using DL in distinguishing ciphertexts of two messages in **'Different Round, Different Key'** settings for different rounds of SPECK32/64.

| **No. of Rounds - $\mathcal{R}$ - Key (Training)** | **Accuracy (Round $\mathcal{R}$ - $\mathcal{K}_i$)** | **No. of Rounds - $\mathcal{R}+1$ - $\mathcal{K}_j$ (Evaluation)** | **Accuracy ($\mathcal{R}+1$, $\mathcal{K}_j$)** | **TPR** | **TNR** |
|---|---|---|---|---|---|
| 5 - $\mathcal{K}_1$ | 0.9944 | 6 - $\mathcal{K}_2$ | **0.4986** | **0.4950** | **0.5022** |
| 6 - $\mathcal{K}_1$ | 0.9907 | 5 - $\mathcal{K}_2$ | **0.5004** | **0.5017** | **0.4991** |

**Summary of Experiment A.** In sum, the series of sub-experiments highlights key limitation of the DL model used in MIND-Crypt its performance declines significantly when tested with different keys or additional encryption rounds. To achieve robustness under varied cryptographic conditions, the model requires retraining with a substantial amount of data, posing practical challenges.

To address this limitation, we conducted Experiment B, which employs TL techniques. This approach leverages base knowledge from the initial training and requires significantly fewer data points

to adapt the model to new keys or rounds. Experiment B aims to enhance model adaptability with reduced data dependency, potentially overcoming the highlighted challenges of Experiment A.

**Experiment B - TL for attacking Indistinguishability.** In this experiment, we addressed the significant challenges identified in Experiment A related to the DL model's performance deterioration when classifying ciphertexts under conditions that vary either in encryption keys or in rounds. Notably, the high data demands for retraining the DL model emerged as a practical barrier in scenarios where data is scarce.

To address these issues, Experiment B introduces a TL approach. We utilized the DL model, initially trained on ciphertexts encrypted with key $\mathcal{K}_1$, as a feature extractor for ciphertexts encrypted with a different key, $\mathcal{K}_2$. This method leverages the DL capabilities of the pre-trained DL model to capture essential features from the new set of ciphertexts, despite the change in keys.

By extracting features from ciphertexts generated under $\mathcal{K}_2$ using the DL model trained on $\mathcal{K}_1$, and subsequently training an XGBoost classifier with these features, we were able to achieve performance comparable to the original DL setup with significantly fewer data points. This strategy not only enhances the adaptability of the system but also substantially reduces the data requirements. Our results demonstrate that TL addresses the main limitations observed in Experiment A, showcasing its potential to maintain indistinguishability in cryptographic systems even when encryption parameters vary. The approach confirms the utility of combining DL for robust feature extraction with the efficiency of a shallow ML model like XGBoost, providing a compelling solution to the challenges of key variability.

For TL objectives, we have chosen to focus on the "Different Key" and "Different Round" scenarios for this experiment. The "Same Round, Same Key" setting, while useful for baseline comparisons, does not challenge the model in ways that align with our goals of assessing adaptability and reducing data requirements. By concentrating on scenarios that involve key and round variations, we aim to test the effectiveness of employing TL to adapt to changes that more accurately reflect practical cryptographic applications where conditions may not remain constant.

**Same Rounds, Different Key.** In this sub-experiment, we assess the efficacy of a TL approach where a DL model, initially trained on ciphertexts encrypted with $\mathcal{K}_1$ for $\mathcal{R}$ rounds, is utilized as a feature extractor. The objective was to analyze how well the model can generalize features extracted from a new set of ciphertexts generated with a different key, $\mathcal{K}_2$, while keeping the number of encryption rounds constant. The results show that the performance of the model improves significantly with increasing number of samples for training.

Table 4 shows performance of TL in "Same Rounds, Different Key" settings. For smaller sample sizes, such as 5,000 samples per class, the model's performance substantially exceeds random guessing, achieving approximately 53% accuracy. This performance is particularly notable given the challenge of distinguishing between ciphertexts encrypted under different keys but with the same number of rounds. As the sample size increases, all performance metrics notably improve; at 290,000 samples per class, the model achieves an accuracy close to 99.4%. It can also be observed that number of rounds, ranging from 5 to 22, does not significantly affect performance, indicating that the model's feature extraction capabilities are robust across different encryption complexities when trained with adequate data. This robustness is essential for practical applications where encryption settings might vary but the key differences remain the central variable.

Table 4: Performance of TL in distinguishing ciphertexts of two messages across different sample size for **'Same Round, Different Key'** sub-experiment.

| No. of Samples per class (in 1000s) | No. of Rounds | Evaluation with TL on $\mathcal{K}_2$ | | |
|---|---|---|---|---|
| | | Accuracy | TPR | TNR |
| 5 | 5 | 0.5372 | 0.5379 | 0.5366 |
| | 6 | 0.5387 | 0.5489 | 0.5286 |
| | 21 | 0.5361 | 0.5289 | 0.5432 |
| | 22 | 0.5357 | 0.5273 | 0.5441 |
| 10 | 5 | 0.5693 | 0.5625 | 0.5760 |
| | 6 | 0.5721 | 0.5745 | 0.5698 |
| | 21 | 0.5717 | 0.5711 | 0.5723 |
| | 22 | 0.5721 | 0.5729 | 0.5712 |
| 100 | 5 | 0.8914 | 0.8896 | 0.8933 |
| | 6 | 0.8895 | 0.8909 | 0.8882 |
| | 21 | 0.8907 | 0.8898 | 0.8915 |
| | 22 | 0.8913 | 0.8900 | 0.8926 |
| 200 | 5 | 0.9759 | 0.9760 | 0.9758 |
| | 6 | 0.9770 | 0.9771 | 0.9769 |
| | 21 | 0.9744 | 0.9750 | 0.9739 |
| | 22 | 0.9761 | 0.9771 | 0.9751 |
| 290 | 5 | 0.9941 | 0.9942 | 0.9940 |
| | 6 | 0.9930 | 0.9930 | 0.9930 |
| | 21 | 0.9933 | 0.9937 | 0.9929 |
| | 22 | 0.9941 | 0.9945 | 0.9937 |

**Different Rounds, Same Key.** In this setting, we evaluate the efficacy of TL approach utilizing a DL model pre-trained on ciphertexts generated with $\mathcal{K}_1$ as a feature extractor for ciphertexts encrypted under the same key but with varying rounds. The objective of this sub-experiment is to demonstrate that with TL, significantly fewer samples are needed to achieve higher accuracy compared to traditional DL approach.

Our results shows that the TL approach not only achieves an accuracy significantly above random guessing with as few as 5,000 samples per class but also reaches an accuracy comparable to the DL approach with 290,000 samples per class. Specifically, the accuracy at the lowest sample size is already substantial at ≈54% and increases to ≈99% with 290,000 samples, matching the performance achieved by the DL approach for disntinguishing ciphertexts with 5 million samples per class. This represents reduction of about 94.2% in the number of samples required per class. Table 5 shows the accruacy, TPR, and TNR with increasing samples size for different rounds, same key sub-experiment.

The experiment shows that the model adapts well to both increase and decrease in the number of encryption rounds, with performance metrics improving consistently as the number of samples increases. This robust adaptability, demonstrated across transitions such as from 5 to 6 rounds and 6 to 5 rounds, underscores that TL's ability to generalize effectively across varying encryption complexities. Importantly, the performance does not merly increases with increase with more data; it rapidly approaches optimal levels previously attainable with significantly larger datasets using DL.

**Different Rounds, Different Key.** In this setting, we evaluate the efficacy of TL approach utilizing a DL model pre-trained on ciphertexts generated with $\mathcal{K}_1$ for $\mathcal{R}$ rounds of SPECK32/64, repurposed

Table 5: Performance of TL in distinguishing ciphertexts of two messages across different sample size for **'Different Rounds, Same Key'** sub-experiment.

| No. of Samples per class (in 1000s) | Feature Extractor Rounds | Evaluation | | | |
|---|---|---|---|---|---|
| | | Round | Accuracy | TPR | TNR |
| 5 | 5 | 6 | 0.5408 | 0.5396 | 0.5355 |
| | 6 | 5 | 0.5315 | 0.5234 | 0.5396 |
| | 21 | 22 | 0.5381 | 0.5420 | 0.5342 |
| | 22 | 21 | 0.5384 | 0.5427 | 0.5341 |
| 10 | 5 | 6 | 0.5692 | 0.5770 | 0.5615 |
| | 6 | 5 | 0.5695 | 0.5668 | 0.5721 |
| | 21 | 22 | 0.5695 | 0.5685 | 0.5705 |
| | 22 | 21 | 0.5705 | 0.5633 | 0.5777 |
| 100 | 5 | 6 | 0.8901 | 0.8918 | 0.8884 |
| | 6 | 5 | 0.8901 | 0.8898 | 0.8905 |
| | 21 | 22 | 0.8910 | 0.8931 | 0.8889 |
| | 22 | 21 | 0.8919 | 0.8924 | 0.8914 |
| 200 | 5 | 6 | 0.9764 | 0.9759 | 0.9769 |
| | 6 | 5 | 0.9765 | 0.9769 | 0.9761 |
| | 21 | 22 | 0.9770 | 0.9760 | 0.9781 |
| | 22 | 21 | 0.9750 | 0.9737 | 0.9763 |
| 290 | 5 | 6 | 0.9942 | 0.9936 | 0.9948 |
| | 6 | 5 | 0.9940 | 0.9945 | 0.9936 |
| | 21 | 22 | 0.9935 | 0.9932 | 0.9937 |
| | 22 | 21 | 0.9936 | 0.9930 | 0.9942 |

as a feature extractor for the ciphertexts under a different key $\mathcal{K}_2$, with both increased and decreased rounds.

Our results show that even with as few as 5,000 samples per class, the TL approach achieved accuracy significantly above random guessing. Accuracy for transitions from 5 to 6 rounds and from 21 to 22 rounds started at approximately 53.43%. Interestingly, in reverse round scenarios like 6 to 5 and 22 to 21, accuracies improved slightly, reaching 53.92% with the same minimal sample size.

When sample sizes were increased to 10,000, accuracy across forward and reverse transitions improved noticeably, averaging around 57.43%. With 100,000 samples, accuracy for all transitions exceeded 88.59%. This upward trend continued with 200,000 samples achieving near 97.75% accuracy, and 290,000 samples reaching peak accuracy close to 99.44%. These results matched the performance levels achieved by the traditional DL approach, which required up to 5 million samples per class, demonstrating a reduction of about 94% in sample size requirements. Table 6 in shows the accruacy, TPR, and TNR with increasing samples size for this sub-experiment.

The adaptability of the TL across both forward and reverse round changes has significant implications for cryptographic security. This flexibility implies that an attacker does not need to know the exact round configurations used by a victim, as the model can generalize across different rounds with high accuracy. For instance, if an attacker trains a model on 22 rounds and the victim uses only 21 rounds, or vice versa, the robustness of the TL approach allows the attacker's model to adapt and maintain efficacy.

**Summary of Experiment B.** To summarize, the sub-experiments addresses the limitations identified in Experiment A by implementing TL. This approach efficiently utilized a smaller dataset to adapt the model to new cryptographic scenarios, such as different keys and different encryption rounds. The results demonstrate significant improvement in the model's adaptability and robustness with reduced data dependency, offering a practical solution to the challenges previously observed.

Table 6: Performance of TL in distinguishing ciphertexts of two messages across different sample size for **'Different Rounds, Different Key'** sub-experiment.

| No. of Samples per class (in 1000s) | Feature Extractor Trained on Round $\mathcal{R}$ - $\mathcal{K}_1$ | Evaluation | | | |
|---|---|---|---|---|---|
| | | No. of Rounds ($\mathcal{R}$ + 1) - Key ($\mathcal{K}_2$) | Accuracy | TPR | TNR |
| 5 | 5 - $\mathcal{K}_1$ | 6 - $\mathcal{K}_2$ | 0.5343 | 0.5366 | 0.5319 |
| | 21 - $\mathcal{K}_1$ | 22 - $\mathcal{K}_2$ | 0.5343 | 0.5366 | 0.5319 |
| | 6 - $\mathcal{K}_1$ | 5 - $\mathcal{K}_2$ | 0.5376 | 0.5376 | 0.5375 |
| | 22 - $\mathcal{K}_1$ | 21 - $\mathcal{K}_1$ | 0.5392 | 0.5415 | 0.5368 |
| 10 | 5 - $\mathcal{K}_1$ | 6 - $\mathcal{K}_2$ | 0.5743 | 0.5832 | 0.5655 |
| | 21 - $\mathcal{K}_1$ | 22 - $\mathcal{K}_2$ | 0.5744 | 0.5737 | 0.5751 |
| | 6 - $\mathcal{K}_1$ | 5 - $\mathcal{K}_2$ | 0.5689 | 0.5706 | 0.5671 |
| | 22 - $\mathcal{K}_1$ | 21 - $\mathcal{K}_2$ | 0.5717 | 0.5773 | 0.5661 |
| 100 | 5 - $\mathcal{K}_1$ | 6 - $\mathcal{K}_2$ | 0.8859 | 0.8867 | 0.8852 |
| | 21 - $\mathcal{K}_1$ | 22 - $\mathcal{K}_2$ | 0.8878 | 0.8878 | 0.8878 |
| | 6 - $\mathcal{K}_1$ | 5 - $\mathcal{K}_2$ | 0.8911 | 0.8903 | 0.8919 |
| | 22 - $\mathcal{K}_1$ | 21 - $\mathcal{K}_2$ | 0.8917 | 0.8933 | 0.8901 |
| 200 | 5 - $\mathcal{K}_1$ | 6 - $\mathcal{K}_2$ | 0.9765 | 0.9773 | 0.9756 |
| | 21 - $\mathcal{K}_1$ | 22 - $\mathcal{K}_2$ | 0.9757 | 0.9756 | 0.9758 |
| | 6 - $\mathcal{K}_1$ | 5 - $\mathcal{K}_2$ | 0.9760 | 0.9763 | 0.9757 |
| | 22 - $\mathcal{K}_1$ | 21 - $\mathcal{K}_2$ | 0.9763 | 0.9754 | 0.9771 |
| 290 | 5 - $\mathcal{K}_1$ | 6 - $\mathcal{K}_2$ | 0.9937 | 0.9934 | 0.9940 |
| | 21 - $\mathcal{K}_1$ | 22 - $\mathcal{K}_2$ | 0.9944 | 0.9945 | 0.9942 |
| | 6 - $\mathcal{K}_1$ | 5 - $\mathcal{K}_2$ | 0.9935 | 0.9936 | 0.9934 |
| | 22 - $\mathcal{K}_1$ | 21 - $\mathcal{K}_2$ | 0.9940 | 0.9941 | 0.9939 |

## 7 DISCUSSIONS

**Summary and Further Insights.** In this study, we investigate the application of ML techniques, specifically DL and TL, in cryptanalysis using MIND-Crypt. Our findings indicate that the proposed attack can distinguish between ciphertexts of two plaintext messages generated using the same key with SPECK32/64, highlighting vulnerabilities in this lightweight block cipher. The attacks shows that DL and TL models can distinguish ciphertexts with significantly higher accuracy when tested under conditions identical to their training. However, their performance deteriorate when the key or number of rounds was altered. This decline in performance is attributed to the change in data distribution under different cryptographic settings, a challenge in machine learning known as out-of-distribution generalization, which remains an open problem across various domains [37] including cryptanalysis. This indicates that while attacks under static conditions are successful, altering cryptographic conditions such as the encryption key or rounds can effectively thwart such attacks. This underscores the importance of dynamic security protocols, suggesting that regular updates to cryptographic configurations can significantly enhance security against ML-based attacks.

However, our findings also reveal a critical vulnerability: attackers can quickly retrain these models to adapt to new cryptographic conditions, effectively distinguishing ciphertexts even under varied encryption parameters. This rapid adaptability of DL and TL models demonstrates a significant breach in the security of the SPECK32/64 encryption scheme, indicating that it may be fundamentally compromised. The ability of attackers to swiftly adjust and launch successful attacks under different conditions underscores the urgent need for reevaluating and enhancing the resilience of cryptographic systems against such evolving threats. We tested – Same Rounds, Same Key; Same Rounds, Different Key; Different Rounds, Same Key; and Different Rounds, Different Key – experimental scenarios each providing unique challenges and vulnerabilities that could have profound implications in real-world cryptographic applications. Under the condition of using the same rounds and the same key, our models demonstrated optimal performance, suggesting a

potential vulnerability in static operational environments where cryptographic settings remain unchanged, increasing susceptibility to pattern recognition attacks. Conversely, using the same rounds but different keys led to a significant drop in performance, affirming SPECK32/64's robustness against simple key substitution, though it may still be prone to more sophisticated adaptive attacks. Changing to different rounds while keeping the same key showed a decline in model accuracy, highlighting the importance of the number of encryption rounds in enhancing security. Systems using a static number of rounds could be vulnerable if attackers discern the round configuration. The most secure setting tested involved different rounds and different keys, which severely challenged our models, underscoring the cipher's strength against advanced ML attacks.

**Limitations.** Our study specifically uses the SPECK32/64 cipher, and the DL model published by Gohr, which may limit the generalizability of our attack. Exploring other types of DNNs, such as ResNet-50 [25], Wide Residual Networks [45], or PyramidNet [24], could potentially make the attack more powerful, by and providing a better feature set, addressing some of these limitations. Additionally, while our research incorporates TL through XGBoost, considering other shallow ML classifiers such as Support Vector Machines, Logistic Regression, or Decision Trees could offer new insights and improvements in the performance of our approach. These alternatives could provide different strengths in handling the dataset and feature sets used in cryptographic analysis, potentially enhancing the robustness and adaptability of our models against various cryptographic configurations.

**Future Work.** Future studies should include various families of block ciphers with different key sizes, public-key cryptography, and assess the adaptability and effectiveness of ML-based cryptanalysis across different cryptographic algorithms. They should develop techniques to reduce dependency on extensive datasets, perhaps by exploring unsupervised or semi-supervised learning methods, which could address the limitations posed by data availability. Additionally, to enhance the model's robustness to withstand changes in cryptographic configurations without extensive retraining, potentially through approaches like meta-learning or few-shot learning, making ML-based cryptanalysis tools more flexible and widely applicable. Integrating these ML and TL approaches with traditional cryptanalysis techniques can create a more robust hybrid method, and extending the study to post-quantum cryptography algorithms would provide insights into their resistance to emerging cryptanalytic challenges.

## 8 RELATED WORK

**Linear & Differential Cryptanalysis.** Albrecht et al. [7] introduced a unified framework that synergistically incorporates various differential cryptanalysis techniques, including standard, truncated, and impossible differentials. These methods are particularly effective in extending the capabilities of known attacks against lightweight block ciphers such as KATAN-32. Following a similar thematic exploration, Dinur et al. [20] and Blondeau et al. [17] refined differential cryptanalysis techniques specifically for a round-reduced version of SPECK, highlighting potential weaknesses of these ciphers under constrained operational conditions. In parallel, Ashur et al. [10] examined the SPECK cipher using linear cryptanalysis, revealing vulnerabilities across various block sizes and demonstrating that linear approximations could be exploited to undermine the cipher's integrity. Complementing these analyses, Biryukov et al. [16] developed a branch-and-bound method that identifies linear and differential trails in ARX-based ciphers. They specifically applied this approach to enhance cryptanalytic attacks against SPECK. Further studies on the operational constraints of these ciphers also support these findings [4, 8].

**ML for Cryptanalysis.** ML has emerged as a powerful tool in cryptanalysis, Mehmood et al. [30] undertook a comprehensive evaluation of distinguishability on the ciphertexts of AES-128 cipher in CBC and ECB modes. Their methodology involved employing Support Vector Machine, k-Nearest Neighbours, and Random Forest Classifiers trained on the frequency distribution of characters in the ciphertexts. The results underscored the susceptibility of the ECB mode, thereby emphasizing the need for robust encryption techniques. Building upon this foundation, Hu et al. [26] explored by applying Random Forest classifiers to diverse block ciphers, reinforcing the vulnerability of the ECB mode. These collective endeavors not only showcase the evolving landscape of machine learning-based cryptanalysis but also highlight its pivotal role in ensuring the resilience of cryptographic algorithms.

Xiao et al. [43] significantly contributed to the field of neural network (NN) based cryptanalysis by introducing a novel approach that not only focuses on the development of neural distinguishers but also emphasizes metrics for efficacy assessment. Their framework, applied to Cyber-Physical Systems (CPS) ciphers, adds depth to the understanding of NN-based cryptanalysis. In parallel, Gohr et al. [23] provided a detailed analysis of a neural distinguisher surpassing state-of-the-art cryptanalysis on SPECKM/N. The paper explores the distinguisher's reliance on differential distribution, likening it to constructing an approximation of the Differential Distribution. Further research regarding the application of ML in cryptanalysis are addressed in Appendix B, providing additional insights into the topic.

## 9 CONCLUSION

In conclusion, our research examined indistinguishability within the SPECK32/64 cipher (CBC mode) using deep learning (DL) and transfer learning (TL) techniques. This cipher was specifically chosen due to its prevalence in resource constrained applications (e.g., IoT devices). We introduced an attack, referred to as MIND-Crypt, that effectively distinguishes between the ciphertexts of two messages with high accuracy under consistent cryptographic conditions. However, its efficacy drops to random chance when cryptographic settings vary, highlighting the model's sensitivity to changes in key and rounds. By employing TL, we managed to significantly reduce data requirements by 94% while maintaining high accuracy. This indicates that the encryption scheme can be compromised using fewer data points, highlighting a potential vulnerability in its resilience against sophisticated attacks. By identifying vulnerabilities, our work underscores the importance of rigorous security assessments to ensure that cryptographic schemes can withstand ML-based cyber attacks.

## REFERENCES

[1] 2022. Internet of Things | TechCrunch — techcrunch.com. https://techcrunch.com/tag/internet-of-things/.

[2] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. USENIX Association, Savannah, GA, 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

[3] Adeeb Abbas, Vasil Pano, Geoffrey Mainland, and Kapil R. Dandekar. 2022. Radio Modulation Classification Using Deep Residual Neural Networks. *MILCOM 2022 - 2022 IEEE Military Communications Conference (MILCOM)* (2022), 311–317. https://api.semanticscholar.org/CorpusID:256244037

[4] Farzaneh Abed, Eik List, Stefan Lucks, and Jakob Wenzel. 2014. Differential Cryptanalysis of Round-Reduced Simon and Speck. In *Fast Software Encryption Workshop*. https://api.semanticscholar.org/CorpusID:43915551

[5] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[6] Aiman Al-Sabaawi. 2021. Cryptanalysis of Classic Ciphers: Methods Implementation Survey. In *2021 International Conference on Intelligent Technologies (CONIT)*. 1–6. https://doi.org/10.1109/CONIT51480.2021.9498530

[7] Martin R. Albrecht and Gregor Leander. 2012. An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers. *IACR Cryptol. ePrint Arch.* 2012 (2012), 401. https://api.semanticscholar.org/CorpusID:8073420

[8] Martin R. Albrecht and Gregor Leander. 2012. An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers. *IACR Cryptol. ePrint Arch.* 2012 (2012), 401. https://api.semanticscholar.org/CorpusID:8073420

[9] Michael Appel, Andreas Bossert, Steven Cooper, Tobias Kußmaul, Johannes Löffler, Christof Pauer, and Alexander Wiesmaier. 2016. Block ciphers for the iot–simon, speck, katan, led, tea, present, and sea compared. *Proc. Appel Block CF* (2016), 1–37.

[10] Tomer Ashur and Daniël Bodden. 2016. Linear cryptanalysis of reduced-round speck. https://api.semanticscholar.org/CorpusID:3738749

[11] Alessandro Barenghi, Luca Breveglieri, Israel Koren, and David Naccache. 2012. Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures. *Proc. IEEE* 100, 11 (2012), 3056–3076.

[12] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. 2015. SIMON and SPECK: Block Ciphers for the Internet of Things. *Cryptology ePrint Archive* (2015).

[13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. 2015. The SIMON and SPECK lightweight block ciphers. In *Proceedings of the 52nd annual design automation conference*. 1–6.

[14] Adrien Benamira, David Gerault, Thomas Peyrin, and Quan Quan Tan. 2021. A Deeper Look at Machine Learning-Based Cryptanalysis. Cryptology ePrint Archive, Paper 2021/287. https://eprint.iacr.org/2021/287 https://eprint.iacr.org/2021/287.

[15] Yoshua Bengio. 2009. Learning Deep Architectures for AI. *Found. Trends Mach. Learn.* 2, 1 (jan 2009), 1–127. https://doi.org/10.1561/2200000006

[16] Alex Biryukov, Vesselin Velichkov, and Yann Le Corre. 2016. Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck. In *Fast Software Encryption Workshop*. https://api.semanticscholar.org/CorpusID:6841381

[17] Céline Blondeau and Benoît Gérard. 2011. Multiple Differential Cryptanalysis: Theory and Practice. In *Fast Software Encryption Workshop*. https://api.semanticscholar.org/CorpusID:8036578

[18] Cisco. 2024. Cisco Internet of Things (IoT) Study. https://softwarestrategiesblog.com/tag/cisco-internet-of-things-iot-study/#:~:text=The%20global%20Internet%20of%20Things,B%20in%20global%20IoT%20spending

[19] John A Clark and Jeremy L Jacob. 2002. Fault injection and a timing channel on an analysis technique. In *Advances in Cryptology—EUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28–May 2, 2002 Proceedings 21*. Springer, 181–196.

[20] Itai Dinur. 2014. Improved Differential Cryptanalysis of Round-Reduced Speck. *IACR Cryptol. ePrint Arch.* 2014 (2014), 320. https://api.semanticscholar.org/CorpusID:17324074

[21] Jean-Max Dutertre, Jacques J.A. Fournier, Amir-Pasha Mirbaha, David Naccache, Jean-Baptiste Rigaud, Bruno Robisson, and Assia Tria. 2011. Review of fault injection mechanisms and consequences on countermeasures design. In *2011 6th International Conference on Design Technology of Integrated Systems in Nanoscale Era (DTIS)*. 1–6. https://doi.org/10.1109/DTIS.2011.5941421

[22] Aron Gohr. 2019. Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II* (<conf-loc>Santa Barbara, CA, USA</conf-loc>). Springer-Verlag, Berlin, Heidelberg, 150–179. https://doi.org/10.1007/978-3-030-26951-7_6

[23] Aron Gohr. 2022. Brute Force Cryptanalysis. Cryptology ePrint Archive, Paper 2022/053. https://eprint.iacr.org/2022/053 https://eprint.iacr.org/2022/053.

[24] Dongyoon Han, Jiwhan Kim, and Junmo Kim. 2016. Deep Pyramidal Residual Networks. *CoRR* abs/1610.02915 (2016). arXiv:1610.02915 http://arxiv.org/abs/1610.02915

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[26] Xinyi Hu and Yaqun Zhao. 2019. Block ciphers classification based on random forest. In *Journal of Physics: Conference Series*, Vol. 1168. IOP Publishing, 032015.

[27] John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. 1998. Side channel cryptanalysis of product ciphers. In *Computer Security — ESORICS 98*, Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 97–110.

[28] Motilal Singh Khoirom, Dolendro Singh Laiphrakpam, and Tuithung Themrichon. 2018. Cryptanalysis of multimedia encryption using elliptic curve cryptography. *Optik* 168 (2018), 370–375.

[29] Parveiz Nazir Lone, Deep Singh, Veronika Stoffová, Deep Chandra Mishra, Umar Hussain Mir, and Neerendra Kumar. 2022. Cryptanalysis and improved image encryption scheme using elliptic curve and affine hill cipher. *Mathematics* 10, 20 (2022), 3878.

[30] Zeeshan Mehmood, Aiman Sultan, Fawad Khan, and Shahzaib Tahir. 2023. Machine Learning Based Encrypted Content Type Identification. In *2023 International Conference on Communication Technologies (ComTech)*. 117–122. https://doi.org/10.1109/ComTech57708.2023.10164955

[31] Chu Jiann Mok and Chai Wen Chuah. 2019. An Intelligence Brute Force Attack on RSA Cryptosystem. *Communications in Computational and Applied Mathematics* 1, 1 (2019).

[32] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

[33] Raphael C W Phan and Sung-Ming Yen. 2006. Amplifying side-channel attacks with techniques from block cipher cryptanalysis. In *Smart Card Research and Advanced Applications: 7th IFIP WG 8.8/11.2 International Conference, CARDIS 2006, Tarragona, Spain, April 19-21, 2006. Proceedings 7*. Springer, 135–150.

[34] Mark Randolph and William Diehl. 2020. Power side-channel attack analysis: A review of 20 years of study for the layman. *Cryptography* 4, 2 (2020), 15.

[35] Gunupudi Sai Chaitanya Kumar, Reddi Kiran Kumar, Kuricheti Parish Venkata Kumar, Nallagatla Raghavendra Sai, and Madamachi Brahmaiah. 2024. Deep residual convolutional neural Network: An efficient technique for intrusion detection system. *Expert Syst. Appl.* 238, PB (feb 2024), 16 pages. https://doi.org/10.1016/j.eswa.2023.121912

[36] Cuiping Shao, Dongyan Zhao, Huiyun Li, Song Cheng, Shunxian Gao, and Liuqing Yang. 2023. Detection of security vulnerabilities in cryptographic ICs against fault injection attacks based on compressed sensing and basis pursuit. *Journal of Cryptographic Engineering* (2023), 1–14.

[37] Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. 2021. Towards Out-Of-Distribution Generalization: A Survey. *CoRR* abs/2108.13624 (2021). arXiv:2108.13624 https://arxiv.org/abs/2108.13624

[38] Yue Shu, Renchao Qin, Yaying He, Ya Li, Rui Jiang, and Zhiyuan Wu. 2022. Deep Residual Neural Networks with Attention Mechanism for Spatial Image Steganalysis. *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)* (2022), 1721–1727. https://api.semanticscholar.org/CorpusID:257808542

[39] Subinoy Sikdar and Malay Kule. 2022. Recent Trends in Cryptanalysis Techniques: A Review. In *International Conference on Frontiers in Computing and Systems*. Springer, 209–222.

[40] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), 2818–2826. https://api.semanticscholar.org/CorpusID:206593880

[41] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 2018. A Survey on Deep Transfer Learning. *CoRR* abs/1808.01974 (2018). arXiv:1808.01974 http://arxiv.org/abs/1808.01974

[42] Rajat Verma, Namrata Dhanda, and Vishal Nagar. 2022. Enhancing security with in-depth analysis of brute-force attack on secure hashing algorithms. In *Proceedings of Trends in Electronics and Health Informatics: TEHI 2021*. Springer, 513–522.

[43] Ya Xiao, Qingying Hao, and Danfeng Daphne Yao. 2019. Neural cryptanalysis: metrics, methodology, and applications in CPS ciphers. In *2019 IEEE conference on dependable and secure computing (DSC)*. IEEE, 1–8.

[44] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *ArXiv* abs/1411.1792 (2014). https://api.semanticscholar.org/CorpusID:362467

[45] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. *CoRR* abs/1605.07146 (2016). arXiv:1605.07146 http://arxiv.org/abs/1605.07146

[46] YongBin Zhou and DengGuo Feng. 2005. Side-channel attacks: Ten years after its publication and the impacts on cryptographic module security testing. *Cryptology*

*ePrint Archive* (2005).
[47] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2019. A Comprehensive Survey on Transfer Learning. *CoRR* abs/1911.02685 (2019). arXiv:1911.02685 http://arxiv.org/abs/1911.02685

# A ENCRYPTION KEYS

In this section, we provide information about the encryption keys used in the generating the dataset for training and evaluating the DL and TL model across various experimental scenarios considered in our study. Table 7 provided hexadecimal representation of the keys $\mathcal{K}_1$ and $\mathcal{K}_2$.

Table 7: Keys used for generating datasets for training and evaluating MIND-Crypt.

| | |
|---|---|
| $\mathcal{K}_1$ | 0x59, fd, 06, 41, 5f, 53, db, 99 |
| $\mathcal{K}_2$ | 0xfd, fe, 9c, a6, 10, 5c, b9, c7 |

# B EXTENDED LITERATURE REVIEW

In this section, we detailed additional studies and findings pertaining to application of ML in cryptanalysis.

Classical cryptanalysis methods, deeply rooted in the mathematical underpinnings of cryptographic algorithms and ciphertexts, Sabaawi et al. [6] extended these traditional techniques by surveying cryptanalysis implementation on ciphers like Caesar, transposition, and Hill. Their work showcases the simplicity and effectiveness of classical methods, contributing to a deeper understanding of cryptanalysis. Simultaneously, Khoirom et al. [28] proposed an image encryption scheme based on elliptic curve cryptography and chaotic maps. Their work identified vulnerabilities in the original scheme, leading to an improved version resilient to chosen-plaintext attacks, differential attacks, and statistical attacks, thereby enhancing security and performance in image encryption. This comprehensive exploration spans classical and contemporary approaches, highlighting the evolving landscape of cryptographic techniques for heightened security across diverse applications.

Despite the computational resource and time intensity associated with brute-force methods, they stand as a last resort, involving an exhaustive search through all possible keys or plaintexts until a match is found. Sikdar et al. [39] conducted a survey on recent cryptanalysis techniques, including brute-force attacks, exploring the growing influence of machine learning in cryptographic methods and suggesting future research directions. Verma et al. [42] delved into the historical significance of brute-force attacks in cybersecurity, emphasizing their enduring relevance for unauthorized data access. The paper extensively covers the evolution of brute-force variants employed by cybercriminals and discusses the corresponding evolution of hashing techniques. Specifically, the focus is on SHA-0, SHA-1, SHA-512, and SHA-256, with the objective of achieving enhanced security through a detailed analysis of brute-force attacks on secure hashing algorithms. Additionally, Mok et al. [31] proposed an intelligent brute-force attack targeting the RSA cryptosystem, simulating and evaluating the effectiveness of their approach in terms of time required for RSA key recovery. Collectively, these works contribute to the understanding and evolution of brute-force cryptanalysis, addressing its challenges and exploring avenues for improved security measures.

While considering side-channel cryptanalysis methods, which focus on the physical characteristics and behaviors of cryptographic devices or implementations, Zhou et al. [46] provided a comprehensive survey covering methods, techniques, and countermeasures in side-channel attacks, evaluating their feasibility and applicability. The paper also discusses the potential adoption of physical security testing in the development of the FIPS 140-3 standard. In a complementary study, Randolph et al. [34] present an in-depth tutorial on power side-channel analysis, spanning the past two decades. The paper elucidates fundamental concepts and practical applications of various attacks, such as Simple Power Analysis (SPA), Differential Power Analysis (DPA), Template Attacks (TA), Correlation Power Analysis (CPA), Mutual Information Analysis (MIA), and Test Vector Leakage Assessment (TVLA), along with the underlying theories. Additionally, the introduction of test statistics as a measure of confidence in detecting side-channel leakage adds depth to the understanding of these analyses. Together, these works contribute to a nuanced exploration of side-channel cryptanalysis, offering insights into methods, countermeasures, and potential advancements in security standards.

In summary, while the reviewed literature presents a comprehensive understanding of various cryptanalysis methods, it is noteworthy that the majority of the approaches explores differential attacks, statistical attacks, chosen-plaintext attacks, etc. In contrast, our study focuses specifically on indistinguishability of lightweight block cipher SPECK32/64 encryption scheme under KPA settings, addressing a critical gap in the literature and providing a more comprehensive evaluation of cryptographic indistinguishability of SPECK32/64.

# C OVERVIEW OF DL AND TL MODELS

Figure 3 depicts the architectures of both the DL and TL models utilized to analyze the indistinguishability of the SPECK32/64 cipher. The models process binary ciphertexts through multiple 1D convolutional layers paired with batch normalization and ReLU activation, designed to extract critical features. While the DL model extends these features through fully connected layers ending with a sigmoid output, while the TL model uses these features as inputs into an ensemble classifier XGBoost, enhancing adaptability and robustness. This approach highlights the effectiveness of feature extraction and reusability across different encryption scenarios, demonstrating the models' capacity to adapt to variations in encryption keys and rounds.

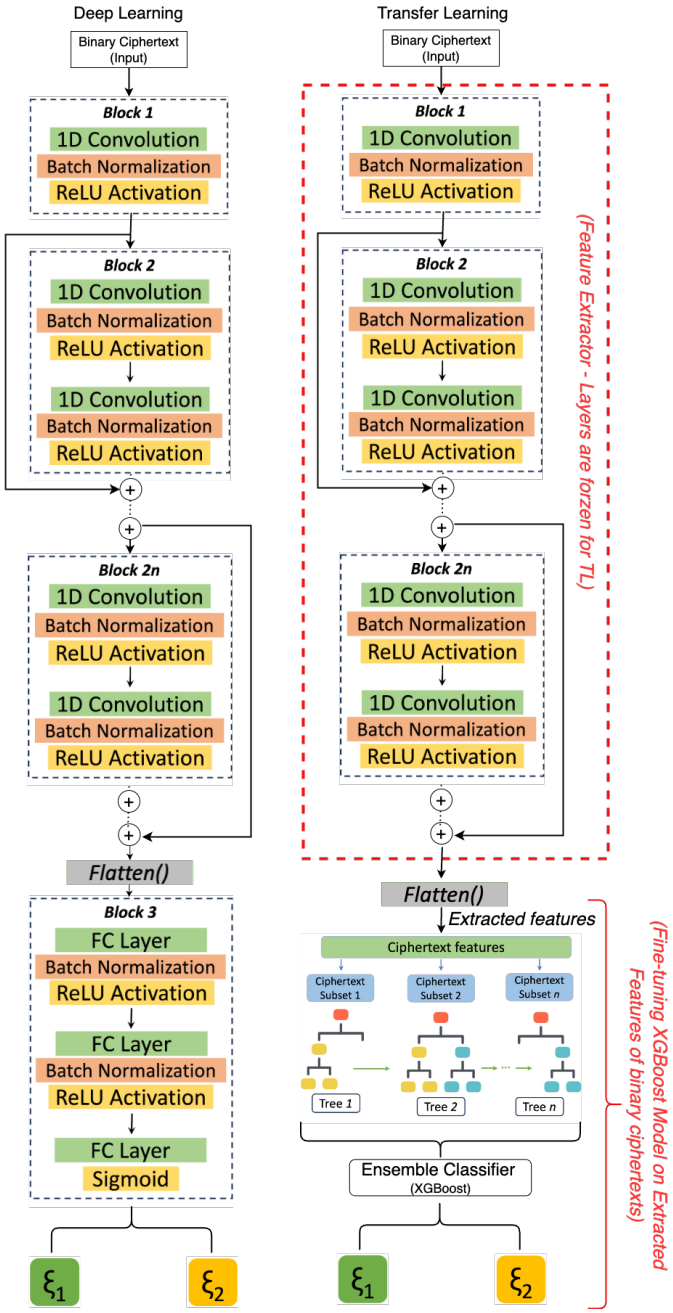


Figure 3: The architecture of the DL model adapted from [14, 22] employed in the MIND-Crypt (left). The feature extractor and the shallow ML model, XGBoost (right) used in the TL approach.

## D HYPERPARAMETERS SEARCH SPACE

In this section, we present the hyperparameter search space utilized for optimization of our XGBoost model. Table 8 details various hyperparameters which significantly influence the model's learning process and predictive performance. We utilized the `MedianPruner` to terminate unpromising trials early, thereby enhancing the efficiency of the search process. The tuning was configured to maximize the objective function, with existing studies being loaded if available, ensuring no repetition in experiments. The outcome is a more robust model that can perform classification efficiently on the challenge ciphertext.

Table 8: Hyperparameter search space for XGBoost model optimized using Optuna with Median Pruning strategy.

| Hyperparameter | Type of Distribution | Search Space |
|---|---|---|
| lambda | Log-uniform | [0.001 .. 10.0] |
| alpha | Log-uniform | [0.001 .. 10.0] |
| max_depth | Integer | [3 .. 9] |
| n_estimators | Integer | [50 .. 1000] |
| learning_rate | Log-uniform | [0.01 .. 1.0] |
| subsample | Uniform | [0.5 .. 1.0] |
| colsample_bytree | Uniform | [0.5 .. 1.0] |
| gamma | Log-uniform | [1e-8 .. 1.0] |
| min_child_weight | Log-uniform | [1 .. 10] |
| reg_alpha | Log-uniform | [1e-8 .. 1.0] |
| reg_lambda | Log-uniform | [1e-8 .. 1.0] |