JCTA Journal of Computing Theories and Applications ISSN:3024-9104

Review Article

Machine Learning and Cryptanalysis: An In-Depth Exploration of Current Practices and Future Potential

Ajeet Singh[1,*], Kaushik Bhargav Sivangi[2], and Appala Naidu Tentu[3]

[1] School of Computing Science and Engineering (SCSE), Vellore Institute of Technology – VIT Bhopal University, Bhopal-Indore Highway, Kothrikalan, Sehore, Madhya Pradesh – 466114, India;
e-mail : ajeetsingh@vitbhopal.ac.in
[2] School of Computing Science, University of Glasgow, United Kingdom;
e-mail : k.sivangi.1@research.gla.ac.uk
[3] C.R.Rao Advanced Institute of Mathematics, Statistics and Computer Science, University of Hyderabad Campus, Prof CR Rao Road, Hyderabad - 500 046, (Telangana) India;
e-mail : naidunit@gmail.com
* Corresponding Author : Ajeet Singh

Abstract: The rapidly evolving landscape of cryptanalysis necessitates an urgent and detailed exploration of the high-degree non-linear functions that govern the relationships between plaintext, key, and encrypted text. Historically, the complexity of these functions has posed formidable challenges to cryptanalysis. However, the advent of deep learning, supported by advanced computational resources, has revolutionized the potential for analyzing encrypted data in its raw form. This is a crucial development, given that the core principle of cryptosystem design is to eliminate discernible patterns, thereby necessitating the analysis of unprocessed encrypted data. Despite its critical importance, the integration of machine learning, and specifically deep learning, into cryptanalysis has been relatively unexplored. Deep learning algorithms stand out from traditional machine learning approaches by directly processing raw data, thus eliminating the need for predefined feature selection or extraction. This research underscores the transformative role of neural networks in aiding cryptanalysts in pinpointing vulnerabilities in ciphers by training these networks with data that accentuates inherent weaknesses alongside corresponding encryption keys. Our study represents an investigation into the feasibility and effectiveness of employing machine learning, deep learning, and innovative random optimization techniques in cryptanalysis. Furthermore, it provides a comprehensive overview of the state-of-the-art advancements in this field over the past few years. The findings of this research are not only pivotal for the field of cryptanalysis but also hold significant implications for the broader realm of data security.

Keywords: Block Ciphers; Cryptographic Algorithms Identification; Deep Learning; Machine Intelligence; Neural Cryptanalysis; Stream Ciphers.

Received: January, 7th 2024
Revised: January, 26th 2024
Accepted: February, 1st 2024
Published: February, 3rd 2024

1. Introduction

Cryptography largely deals with the algorithms and protocols that safeguard the privacy and integrity of data. Typically, cryptographic procedures are characterized as software or Turing machines, and attackers are also defined in these terms, constrained by their complexity (for example, restricted to polynomial time) and their likelihood of success (such as confined to an insignificant probability). A system is considered secure if it can successfully defend against all types of attackers. For example, an encryption algorithm is regarded as secure if no attacker can retrieve information about the original text from the encrypted text. The cryptanalysis of both block and stream ciphers has consistently been a focus of considerable interest. Many ciphertexts are still unsolved, and we have no idea what sort of cipher was employed to encode them. Each cryptanalyst begins by attempting to determine their cipher types using various (statistical) approaches. In principle and practice, cryptographic differentiating attacks allow for sophisticated cryptanalysis, in which an attacker extracts enough "information" from an encrypted message to differentiate it from random data. Deep learning, Machine Learning, and other statistical approaches have significantly impacted various

disciplines[1], [2]. It has a lot of possibilities for cryptanalysis as well. Traditional cryptanalysis approaches demand a thorough examination of non-linear functions that define the correlations among plaintext, key, and ciphertext [3]. These functions contain exceptionally high degree terms, making cryptanalysis exceedingly difficult. In 1991, Rivest [4] highlighted the interconnection between cryptography and machine learning, focusing on how these two fields have exchanged ideas and methodologies with each other. The introduction of deep learning methods, together with more powerful and efficient computing resources, has opened up new avenues for analysing encrypted data in its raw form. The fundamental concept behind creating a cipher involves injecting randomness into it, ensuring that there are no discernible patterns in the encrypted data.

Machine Learning (ML) and Deep Learning (DL) based cryptanalysis represent a cutting-edge intersection of cybersecurity and artificial intelligence, where these computational techniques are applied to the analysis and breaking of encrypted information. This modern approach to cryptanalysis leverages ML and DL algorithms' pattern recognition and predictive capabilities to detect underlying structures and vulnerabilities within cryptographic systems that traditional methods may overlook. In real-time problem-solving, ML and DL can significantly accelerate the decryption process, handling vast and complex datasets efficiently, which is crucial in a landscape where encryption algorithms are increasingly sophisticated. The significance of ML and DL in cryptanalysis is profound in various real-world applications. For instance, in cybersecurity, they can be used to test the strength of encryption algorithms by attempting to find weaknesses or potential backdoors. In digital forensics, these methods can assist in uncovering hidden information [5] in large sets of data, which might be critical in solving crimes. Furthermore, in the realm of secure communications, understanding the potential vulnerabilities of cryptographic techniques through ML and DL helps in enhancing security measures to counter advanced cyber threats. Moreover, the threat to current cryptographic standards is imminent as the world moves towards quantum computing. ML and DL-based cryptanalysis play a pivotal role in preparing for post-quantum cryptography by evaluating which algorithms could withstand quantum attacks. This preemptive analysis is vital for ensuring the future security of digital infrastructure, financial systems, and sensitive communication channels. Thus, ML and DL-based cryptanalysis is not only significant for solving current encryption-related challenges but also indispensable for anticipating and mitigating the cybersecurity threats of tomorrow. The main contribution highlights of this paper are as follows:

- This paper investigates the scope and feasibility of machine learning, deep learning, and some other stochastic optimization methods for cryptanalysis. This study also looks into state-of-the-art advances in this field during the last few years.
- Some experimental results of the discussed computational intelligence methods are also summarised in later sections of this paper.

The remaining paper is structured as follows: Section 2 discusses the impact of stochastic optimization methods in ciphertext-only cryptanalysis. The neural distinguisher is described in section 3. Various approaches for machine learning and deep learning-based cryptanalysis are summarized in section 4. Some experimental scenarios are given in section 5. In conclusion, section 6 wraps up the paper.

2. Impact of Stochastic Optimization Methods in Ciphertext only Cryptanalysis

This section discusses the impact of stochastic optimization methods in ciphertext only cryptanalysis.

2.1. Generalized and Slippery Hill Climbing

Essential Primitives:

Some of the essential primitives i.e., N-Gram Statistics, Cost Function, and Index of Coincidence, are summarized here.

N-Gram Statistics and Cost Function:

An n-gram refers to a continuous series of n elements from a specific text sample. The Fitness Function, sometimes called the Evaluation Function or Cost Function, assesses the proximity

of a particular solution to the optimal solution of the intended problem, thereby determining the solution's suitability. Every computational problem possesses its unique fitness function, and the choice of which fitness function to use is dictated by the specific problem at hand. Designing an appropriate fitness function for a given problem stands as the most challenging aspect in framing a problem through the use of genetic algorithms.

Index of Coincidence (IoC):

The period of the polyalphabetic cipher can be ascertained using the index of coincidence (IoC), which gauges the likelihood that any two characters in a text will match. Owing to the absence of randomness in natural languages, the IoC for meaningful texts is noticeably higher compared to that of random strings. English text has an IoC $\approx$1.75.

Methodology:

T. Kaeding, in his work[6], has discussed the slippery Hill-climbing technique for ciphertext-only cryptanalysis of periodic polyalphabetic substitution ciphers. In a stochastic ciphertext-only attack on a monoalphabetic substitution cipher, a child key is created from a parent key by randomly exchanging elements. The child takes the parent's place if the decrypted text's fitness is an improvement, with fitness being a gauge of how closely the text mirrors a specific language. After many child candidate evaluations, this process continues until no further enhancement is observed. The periodic polyalphabetic substitution cipher is an extension of the monoalphabetic cipher involving several key alphabets. During encryption, the selection of the key alphabet rotates through them, with the key choice determined by the text position modulo the total number of keys. To prevent being stuck at a local peak of textual fitness, keys are sporadically randomized, and the hill is ascended again, a step referred to as "slippery". The total number of key alphabets can often be found by dividing the ciphertext into parts of every n^{th} character and computing the index of coincidence (IoC) for each fragment. The average IoC is taken over slices for each n, and the smallest n close to the IoC of English text is used to determine the number of key alphabets. After setting the keys to initial values, the process loops through them, randomizing each one before using the stochastic method to climb back up. Figure 1 depicts example pseudocode instance of a genetic algorithm for decrypting a cipher using fitness evaluation to select optimal character mappings.

```
parent = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
plaintext = decrypt (ciphertext, parent)
parentfit = fitness (plaintext)
count = 0

while count < LIMIT do
        child = parent
        i = random() mod 26
        j = random() mod 26
        child[i], child[j] = child[j], child[i]
        plaintext = decrypt (ciphertext, child)
        childfit = fitness (plaintext)
        if childfit > parentfit then
                parent = child
                parentfit = childfit
                count = 0
        else
                count++

output parent, decrypt (ciphertext, parent)
```

Figure 1. Pseudocode of genetic algorithm for decrypting a cipher using fitness evaluation to select optimal character mappings

The entire decrypted text's fitness is utilized, not just that of the individual slice. This looping continues until no improvement is seen over a substantial number of steps. To assess whether one decryption is superior to another, there must be a method to evaluate the decryption quality. A fitness function is defined for this purpose, providing a numerical value

to indicate how closely a text resembles English. Various definitions are possible for this function. The chosen approach in this instance is to use the average of the logarithms of the tetragram (four-letter) frequencies, calculated from a significant corpus of English text. Consider a standard periodic polyalphabetic substitution cipher. The process starts by utilizing the index of coincidence to identify the period, followed by selecting an initial set of key alphabets. The discovery is that the running time is reduced, and the likelihood of a successful decryption is augmented if the initial keys are configured to align as closely as possible with the single-letter frequencies of English text. For instance, if 'M' is the most frequent character in a section of the text, the corresponding key element that encrypts 'E' (the most common letter in English) would be assigned to 'M'. Next, the process enters a loop where each key alphabet is randomized, and the hill-climbing method is applied as in the monoalphabetic ciphers, but with a modification: the fitness is computed for the entire plaintext, not just the specific slice the key relates to. This looping through the set of key alphabets continues until there is no improvement in fitness over a certain number of iterations. The decrypted plaintext with the highest fitness is then selected.

2.2. Particle Swarm Optimization

Particle swarm optimization (PSO) is a self-adaptive optimization strategy based on populations. Similar to genetic algorithms, PSO begins with a randomly initialized population of solutions in the search space and iteratively refines them to locate the optimal solution. Within PSO, each possible solution is termed a particle.

Every particle maintains its coordinates in the search space, which correlate with its best-found solution, referred to as the particle best or $pbest$. Additionally, the population's overall best value is recorded, known as the global best or $gbest$. After these two values are identified, each particle's velocity ($v_{i,j}$)and position ($P_{i,j}$) are updated towards its $pbest$ and $gbest$ using the Equations (1).

$$v_{i,j} = c_0 v_{i,j} + c_1 r_1 \left(Ppbest_{i,j} - p_{i,j}\right) + c_2 r_2 \left(Pgbest_{i,j} - P_{i,j}\right), \quad P_{i,j} = P_{i,j} + v_{i,j} \tag{1}$$

Where $Ppbest_{i,j}$ and $Pgbest_{i,j}$ signify the particle best and global best positions, respectively, and $1 \leq r_1, r_2 \leq 0$ are randomly distributed variables. The learning factors c_0, c_1, c_2 to PSO's appeal, as adjusting these allows for diverse applications.

In classic PSO, particles are encoded as strings of positions representing multidimensional space, with each dimension acting independently. But in the context of cryptanalysis[7], each solution in the parameter space symbolizes a permutation of alphabetic characters corresponding to a key, and thus the elements are interdependent.

2.3. Cryptanalysis using Phantom Gradient Attack

A.A. Sommervoll [8] proposed a novel cryptanalytical assault known as the phantom gradient attack. This technique, rooted in machine learning and backpropagation, seeks to recover the encryption key. While traditional neural networks depend on differentiable function gradients, encryption and decryption rely on discrete functions. The unique contribution of this work is to substitute these discrete functions with piecewise differentiable ones, facilitating neural network-based key recovery.

In 2015, Google's DeepDream made the concept of input "training" famous using a pretrained network. The phantom gradient attack extends this idea, modeling a cryptosystem as a neural network. It substitutes the original discrete functions with piecewise differentiable ones that have gradients, termed the phantom gradients.

In symmetric key encryption, a secret key (k) is employed for both encryption and decryption (shown as Equation (2)). The encryption function is symbolized as f_k, and its inverse, used for decryption, asf_k^{-1}. Though easy to find when k is known, uncovering f_k^{-1} is deliberately challenging when k is concealed. The phantom gradient attack aims to retrieve this hidden k, or more specifically, an input x^* for a known function f and output y, so that $f(x^*) = y$.

$$Enck(p) = f_k(p) = f(k,p) = c \tag{2}$$

Here, both k^* and p^* must be recovered, though the recovered k^* is likely to differ from k since $|k| + |p|$ is usually much larger than $|c|$. The authors represent the function f_p as a neural network by replacing the functions involved in the encryption with piecewise differentiable ones. These substitutes are crucial, as their derivatives are utilized to retrieve the key. The loss function serves as a critical indicator of the fidelity between the model's predictions and the actual target values during training. However, it does not inherently account for the validity of the predicted values with respect to the desired range, which is particularly crucial in scenarios like bit recovery, where the expected outputs are binary. Predictions that fall outside the binary threshold of 0 to 1 hold no practical relevance. To address this, an enhanced loss function is implemented, which integrates a penalization term, referred to as the "confinement penalty." This term imposes an increasing penalty on predictions as they deviate further below 0 or rise above 1, effectively steering the model's outputs towards the acceptable binary range. The mathematical representation of this concept introduces a penalty ridge in the loss landscape, penalizing out-of-bound values to ensure the fidelity of the model's predictions not just in accuracy but also in adherence to the defined binary constraints. This is encapsulated by Equation (3), which quantifies the additional penalty for predictions situated outside the [0,1] interval.

$$\text{punish}_{\text{ridge}}(x) = \begin{cases} \frac{1}{2}(x-1)^2 & \text{, for } x > 1 \\ 0 & \text{, for } 0 \leq x \leq 1 \\ \frac{1}{2}x^2 & \text{, for x} < 0 \end{cases} \quad (3)$$

This extra punishment helps in keeping the values within the desired range.

3. Neural Distinguisher (ND)

In the field of machine learning, the practice of including additional features is often employed to enhance the precision of neural networks. This idea is particularly pertinent to the complex domain of neural-aided cryptanalysis, where the ND plays a crucial role. The underlying motivation is that the distinguishing accuracy can be increased by utilizing features obtained from several ciphertext pairs. The efficacy of the ND is pivotal for the success of neural-aided cryptanalysis[5], [9]. A neural network is considered a valid ND if it can achieve a distinguishing accuracy greater than 0.5 with randomly chosen ciphertext pairs. An ND that processes only a single ciphertext pair can be used to identify k incorrectly classified pairs, a method that is considered acceptable based on the following rationale:

- In constructing ND_k, all the ciphertext pairs are generated from different keys, guaranteeing the availability of only two kinds of features: those hidden within a single ciphertext pair and those derived from numerous ciphertext pairs.
- Suppose the ND, when given a single ciphertext pair, demonstrates high accuracy. In that case, this implies that the concealed features within the ciphertext pair offer substantial evidence that can lead to incorrect classifications. If the new ND_k continues to classify such k pairs accurately, it lends credence to the idea that this effectiveness is attributable to features extracted from multiple ciphertext pairs.

4. Machine Learning and Deep Learning based Cryptanalysis

The scope of Machine Learning and Deep Learning based cryptanalysis is summarized here in the perspectives of Block Ciphers and Stream Ciphers.

4.1. In the perspective of Block Ciphers

In principle and practise, cryptographic differentiating attacks allow for sophisticated cryptanalysis, in which an attacker extracts enough "information" from an encrypted message to differentiate it from random data. Expertise using cutting-edge machine learning algorithms to launch cryptographic distinguishing attacks on a variety of public datasets is described by Chou et al. [10]. On these datasets, authors test a variety of current and novel characteristics and discover that the ciphers' "modes of operation" influence classification task performance. When CBC mode is used with a random starting vector for each plaintext, performance is quite poor, however when ECB mode is used, performance is relatively

excellent for specific datasets. Finally, authors conclude that, contrary to previous research, current machine learning approaches cannot extract valuable information from ciphertexts generated by modern ciphers running in a sufficiently safe mode, such as CBC, much alone distinguish them from random data.

Danziger et al. [11], explore the use of a neural cryptanalysis method on S-DES input-output-key data to examine its ability to understand the connections between these components. Their findings reveal that the neural network successfully mapped the relationship between inputs, keys, and outputs, managing to ascertain the correct values for the key bits even with a small sample size (approximately 0.8% of the total data). By employing differential cryptanalysis on the key space, the authors were able to rationalize the neural network's partial success with specific key bits. However, after introducing new s-boxes that were more resilient to the differential attack, the neural network lost the ability to identify any parts of the key.

With the advancement of neural networks in increasingly intricate tasks, they are being trained for end-to-end objectives that transcend mere functional specifications. Generally, neural networks aren't designed to excel in cryptography, as evidenced by the fact that even the simplest ones can't compute XOR, a fundamental component in many cryptographic algorithms. However, as presented by Martin A. et al. (2016). [12], neural networks are capable of learning encryption and decryption methods to protect the confidentiality of their data against other neural networks, even without explicit instructions for these specific tasks. This work not only shows the ability of neural networks to comprehend and selectively apply encryption and decryption but also posits that, although neural networks may never become exceptional at cryptanalysis, they could prove to be valuable in interpreting metadata and traffic analysis.

Cryptanalysis focuses on finding and evaluating the vulnerabilities of ciphers and explores ways to utilize these weaknesses to decipher the plaintext or reveal the concealed cipher key. Taking advantage of these weaknesses is often complex and, in numerous instances, has proven to be effective only on diminished versions of the ciphers. Focardi et al. [13] employ artificial neural networks to augment the cryptanalysts' efforts in exploiting these cipher frailties automatically. By training the networks with data that highlights the weaknesses alongside the corresponding encryption key, the network eventually learns to predict or assess the likelihood of the key for any given ciphertext. Through the utilization of simple classical ciphers, this research introduces the first ciphertext-only attack on substitution ciphers using neural networks, thereby demonstrating the efficacy of this novel approach.

Aidan N. Gomez et al. [14] introduce CipherGAN, an innovative architecture modeled on CycleGAN designed to deduce the underlying cipher mapping from unpaired ciphertext and plaintext collections. This work illustrates the ability of CipherGAN to successfully decipher language data encoded with shift and Vigenere ciphers, achieving a high level of accuracy and handling vocabularies of sizes previously unattained. The authors also outline how to adapt CycleGAN for compatibility with discrete data and how to stabilize its training process. Furthermore, they establish that the methodology employed in CipherGAN effectively circumvents the typical issue of uninformative discrimination that often plagues GANs when applied to discrete data.

Mello D. et al. [15] investigated the use of machine learning approaches for identifying encryption schemes based only on ciphertexts. The experiment included plain text corpora in seven distinct languages, seven encryption techniques, each in ECB and CBC modes, and six categorization algorithms. Each cryptographic technique was used to encrypt plain text files in both cipher modes. The ciphertexts were then processed to provide metadata, which the classification algorithms subsequently employed. The total experiment included not just a large number of ciphertexts but also time-consuming metadata compilation and identification methods.

The examination of block ciphers has been a continual focus of interest. In Crypto 2019, Gohr [16] put forth a multiple differential cryptanalysis of the SPECK cipher, centering on the input difference $\Delta_{\text{in}} = 0\text{x}0040/0000$. This concept is centered on converting a distinguishing challenge into a classification task. The goal here is to differentiate genuine pairs, specifically encryptions of plaintext pairs P, P' where $P \oplus P' = \Delta_{\text{in}}$, from arbitrary pairs encrypted with no consistent input difference. Gohr contrasted a conventional differential distinguisher with a deep neural network (DNN)-based distinguisher for 5 to 8 rounds of SPECK-32/64 [17], demonstrating that the DNN approach was superior. Gohr calculated

the full DDT for the input difference Δ_{in}, based on the Markov assumption, and subsequently used probabilities to classify a ciphertext pair (C, C'). The following classification scheme in form of Equation (4) was adopted.

$$\text{classification/categorization} = \begin{cases} \text{real}, & \text{if DDT}(\Delta_{in} \rightarrow \Delta_{out}) > \dfrac{1}{2^{32}-1} \\ \text{random}, & \text{otherwise} \end{cases} \quad (4)$$

Here, reduced-round SPECK-32/64 distinguishers are symbolized as D_{nr}, and the neural distinguishers as N_{nr}, with $nr \in \{5, 6, 7, 8\}$ indicating the number of rounds. The DNN, illustrated in Figures 2 to 4, is composed of:

- **Block 1:** A one-dimensional convolutional neural network (1D-CNN) that employs a kernel of size 1, utilizes batch normalization, and incorporates a ReLU activation function.
- **Blocks 2:** From one to ten layers, each consisting of two 1D-CNN with a kernel size of 3, followed by batch normalization and a ReLU activation function.
- **Block 3:** A complex final classification block made up of three perceptron layers, interspersed with two batch normalization and ReLU functions and capped with a Sigmoid function.

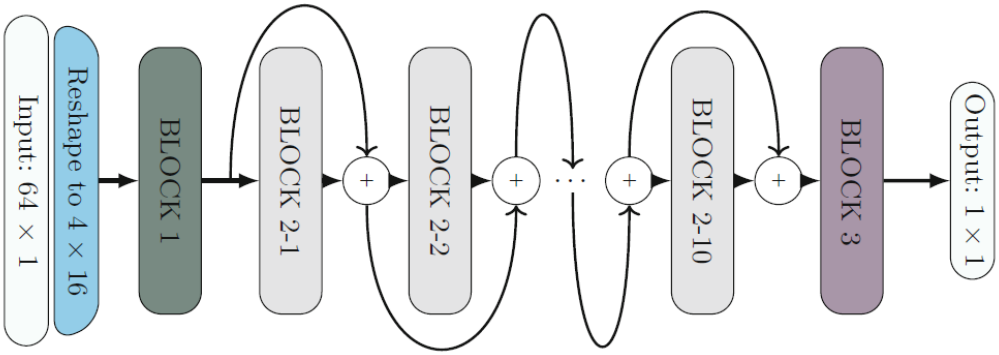

Figure 2. Gohr's deep neural network's entire pipeline consists of different blocks. Block 1 is the starting convolution block. Blocks 2-1 to 2-10 correspond to the residual block, and Block 3 is designated as the classification block.

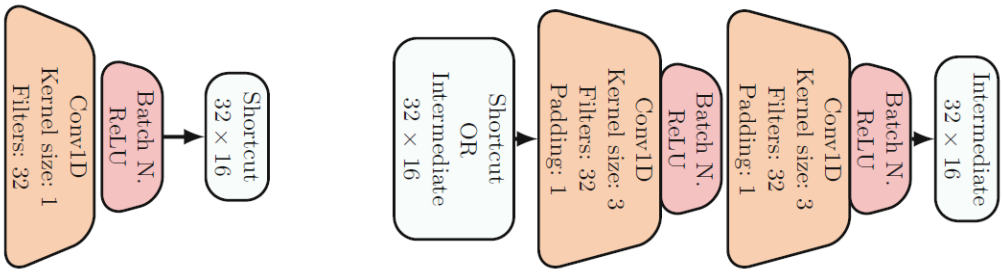

Figure 3. Initial convolution block (Block 1).

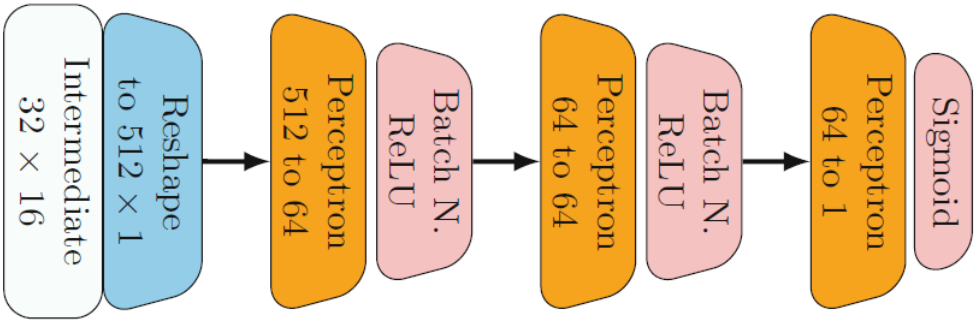

Figure 4. The residual block (Blocks 2-i).

The neural distinguishers produce a real-valued score ranging from 0 to 1 for every pair. If the score is 0.5 or higher, the sample is identified as a real pair; if it's less than 0.5, it is classified as a random pair.

Jaewoo So[2] introduces a universal cryptanalysis model that leverages DL to uncover the keys of block ciphers using known plaintext-ciphertext pairs. By attacking lightweight block ciphers like simplified DES, Simon, and Speck, he demonstrates the viability of employing DL for cryptanalysis. The outcomes reveal that the DL-based method can effectively retrieve key bits if the keyspace is limited to 64 ASCII characters. While traditional cryptanalysis typically occurs without restricting the keyspace, only reduced-round versions of Simon and Speck have been successfully assailed in this way. Even when a text-based key is utilized, the suggested DL-driven cryptanalysis can successfully breach the full rounds of Simon32/64 and Speck32/64. These findings suggest that DL technology may serve as a valuable instrument in the cryptanalysis of block ciphers, especially when the keyspace is confined

Kopal et al. [9] built a preliminary version of an artificial neural network capable of distinguishing between five classic ciphers: basic monoalphabetic substitution, Vigenere, Playfair, Hill, and transposition. The network is built using Keras and Google's TensorFlow framework. The current state of research into utilizing such networks to detect encryption type is presented in this publication. Stamp launched a new MysteryTwister C3 challenge named "Cipher ID" in 2019, and they tried to categorize all of the ciphers in it. About 90% of the ciphertexts in the challenge are properly classified by the network. In addition, the study discusses the current state of encryption type detection.

A. Baksi et al. [1] find that all-in-one differential cryptanalysis is typically more effective than utilizing just a single differential trail. However, it often becomes incredibly challenging to completely compute when the cipher is non-Markov or possesses a large block size. Drawing inspiration from Gohr's work, the authors attempt to imitate the all-in-one differentials for non-Markov ciphers in their research by employing machine learning. They gauge the results using the 8-round distinguisher of GIMLI-PERMUTATION as a standard. For the training process, 218 data samples were utilized, and the number of epochs was capped at 20 to prevent overfitting with higher numbers. The authors explored various neural network architectures, encompassing the basic Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), and Long Short-Term Memory Network (LSTM).

Benamira A. et al. [5] provide an in-depth analysis and comprehensive elucidation of the neural distinguisher's inner mechanics, as initially proposed by [16]. They begin by examining the classified sets, attempting to discern patterns that might lead to a better comprehension of Gohr's findings. Through experimental evidence, they demonstrate that the neural distinguisher's reliance is predominantly on the differential distribution concerning the ciphertext pairs. It also depends on the differential distribution in both the penultimate and antepenultimate rounds. To corroborate these insights, the authors create a distinguisher for the SPECK cipher relying purely on cryptanalysis without utilizing any neural network. This approach essentially achieves the same accuracy and efficiency as Gohr's neural distinguisher, as referenced in [16].

Yi Chen et al. [18] discuss the ND introduced by Gohr at CRYPTO'19 [16], which is founded on plaintext difference. The ND evaluates a ciphertext pair, classifying it as either a real or random pair. While Benamira et al. offered a more detailed examination of how two specific NDs function against Speck32/64 [5], three significant research questions remain, sparking interest among researchers. These questions concern (i) the characteristics of a ciphertext pair that the ND learns, (ii) the way to elucidate various phenomena connected to NDs, and (iii) other potential applications of machine learning within traditional cryptanalysis. In their work, the authors address some of these research gaps. They first introduce the Extended Differential-Linear Connectivity Table (EDLCT), a universal tool that characterizes a cipher. They then design features that correspond to the EDLCT to describe a ciphertext pair. Based on these features, they construct different machine learning-based distinguishers, including the ND. Additionally, the authors employ the EDLCT to clarify the phenomena associated with NDs.

The usage of deep learning (DL) models in a known-plaintext context is investigated in B.Y.Chong et al.[19]. The models' purpose is to use DL approaches to forecast a cipher's secret key. Simplified Data Encryption Standard (S-DES), Speck, Simeck, and Katan are used to test the DL approaches against different cyphers. The authors look at the categorization of the entire key set in S-DES, and the results are better than a random guess. However,

authors discovered that applying the same categorization approach beyond the 2-round Speck is problematic. They also show that deep learning models trained on known-plaintext data can recover the S-DES random key effectively. However, the same strategy has had less success when used with newer ciphers such as Speck, Simeck, and Katan.

Zezhou Hou et al. [20] outline a method for constructing the ciphertext pairs needed for differential cryptanalysis using deep learning. Utilizing this approach, they develop differential distinguishers for SIMON32 with deep residual neural networks for both 9-round and 8-round configurations. Furthermore, they investigate how the input difference patterns influence the accuracy of the deep learning-based distinguisher. Specifically, they find that for input differences with a Hamming weight of 1, the accuracy of the 9-round distinguisher varies between the first 16 bits and the last 16 bits when dealing with non-zero bit positions.

Jain et al. [3] construct a Deep Neural Network-based differential distinguisher for the round reduced lightweight block ciphers PRESENT and Simeck, which is influenced by previous work on differential distinguishers. The state-of-the-art technique is improved and used to two fundamentally dissimilar block ciphers, i.e., PRESENT-80 and Simeck64/128. The acquired findings show that our cryptanalysis approach is ubiquitous. With excellent accuracy, the suggested approach can differentiate random data from encrypted data acquired after 6 rounds of PRESENT and 7 rounds of Simeck encryption. In addition, we investigate a novel strategy for selecting excellent input differentials that, to our knowledge, has never been investigated before.

In the study by Kimura et al. [21], deep learning-based output prediction attacks are introduced in a blackbox setting. The researchers considered two toy Substitution-Permutation Network (SPN) block ciphers (small PRESENT and small AES) and one toy Feistel block cipher (small TWINE) as part of their preliminary investigation. Deep learning models can be constructed using the maximum amount of plaintext/ciphertext pairs, allowing for the precise calculation of the rounds where full diffusion occurs. Subsequently, the paper explores whether the evaluation results obtained from attacks against these three toy block ciphers can be generalized to block ciphers with larger block sizes, such as 32 and 64 bits, building on the initial investigations.

Tarun Y. et al. [22] reveal how machine learning applications offer favorable outcomes for differential cryptanalysis. The authors introduce a novel method that enhances the traditional differential distinguisher by incorporating machine learning (ML). They utilize an r-round classical differential distinguisher to form an s-round ML-based differential distinguisher. This newly devised s-round ML distinguisher aids in the creation of an (r+s)-round differential-ML distinguisher. Notably, this approach achieves a more complex distinguisher while simultaneously reducing the data complexity.

Wenqiang T. et al., [23], examined the safety of the SIMON cipher under neural differential cryptanalysis. They first show that SIMON is a non-Markov cypher theoretically, implying that typical differential cryptanalysis conclusions may be erroneous. Then, for SIMON32/64, we train a residual neural network to generate the 7, 8, and 9-round neural distinguishers. Furthermore, authors briefly discussed the effect of different residual network structures on the training results of neural distinguishers.

Chen et al. [24] present a novel Neural Distinguisher (ND) that takes into account several ciphertext pairings at the same time. Furthermore, separate keys are used to create several ciphertext pairings. The idea is that using characteristics acquired from several ciphertext pairings can increase distinguishing accuracy. This work used this novel ND on five different cyphers to validate this motivation. Experiments have shown that using numerous ciphertext pairs as input improves accuracy.

On Speck32/64 and Simon32/64, Bao et al. [25] designed the first practical 13-round and enhanced 12-round neural-distinguisher-based key-recovery attacks, as well as 16-round key-recovery assaults. The findings support the use of machine-learning methods in cryptanalysis. The major reason, however, is due to the enhancements made to the original components. The output difference of a differential route is important to the ND, whereas the input difference is unimportant. As a result, several differentials can be prepended to ND as long as the output difference is the same. Surprisingly, many such differentials can share some neutral bits. Using such differentials might allow data to be reused, reducing data complexity marginally.

Against Simon and Simeck, Jinyu et al. [26] created neural distinguishers (NDs) and related-key neural distinguishers (RKNDs). Simon32/64's ND and RKND achieve 11-, and 11-

round accuracy of 59.55 percent and 97.90 percent, respectively. The ND achieves 60.32 percent accuracy in 13 rounds for Simon64/128, whereas the RKND achieves 95.49 percent. Simeck32/64 produces ND and RKND of 11- and 14-round, with accuracy of 63.32 percent and 87.06 percent, respectively. For Simeck64/128, authors create 17-round ND and 21-round RKND with accuracy of 64.24 percent and 62.96 percent, respectively. For Simon32/64, Simon64/128, Simeck32/64, and Simeck64/128, these are currently the longest (related-key) neural distinguishers with better precision.

Zhang et al.,[27] employed multiple parallel convolutional layers at the core of their network structure to train a neural distinguisher. This approach was inspired by the Inception Blocks in GoogLeNet and aimed to capture higher dimensional information. Through this method, they enhanced the accuracy of the (5-8)-round neural distinguisher and developed a new 9-round neural distinguisher specifically for Speck32/64. They also increased the accuracy of the (7-11)-round neural distinguisher and trained a new 12-round neural distinguisher for Simon32/64. In addition, the authors extended Gohr's concept of neutral bits. This extension was implemented to meet the need for a uniform distribution among various plaintext pairings, particularly in key recovery attacks.

Traditional symmetric cryptanalysis has seen incremental gains over time, prompting individuals to examine other options. Nicoleta-Norica et al., (2022) [28] investigates Gohr's[16] depth-10 and depth-1 neural distinguishers in order to see if smaller or better-performing distinguishers for Speck32/64 exist. Authors investigate if the performance of a smaller network that accomplishes the same results may be enhanced as well. They also investigate if manipulating the input prior to feeding it to the pruned depth-1 network improves its performance. Convolutional autoencoders were discovered to be capable of effectively reconstructing the ciphertext pairs, and their trained encoders were utilized as a preprocessor before training the pruned depth-1 network.

4.2. In the perspective of Stream Ciphers

Girish M. et al. [28] developed a deep learning-based technique for detecting biases in stream ciphers in the black-box analysis model with the idea that deep learning algorithms would be suitable for cipher data. The goal of the suggested approach is to forecast the appearance of an output bit/byte at a specified place in the keystream created by a stream cipher. The authors tested their approach on the RC4 stream cipher and its upgraded form RC4A and then went over the findings in depth. The approach is also applied to two other stream ciphers, Trivium and TRIAD. The suggested approach can detect bias in RC4 and demonstrate that it does not exist in its enhanced counterpart or the other two ciphers. Focusing on RC4, the authors compare methodology and observations with various current approaches, demonstrating that their procedure is clearer and less difficult than the others.

As this domain being an emergent and challenging interdisciplinary research field, the substantial body of significant scholarly work in the area of ML-based cryptanalysis began to emerge prominently around 2019. Since that juncture, a significant amount of the research has been oriented toward ML applications within the realm of block cipher cryptanalysis. Conversely, the literature concerning the application of ML to stream cipher cryptanalysis remains relatively sparse. The application of Machine Learning (ML) to the cryptanalysis of stream ciphers, as opposed to block ciphers, encounters distinct obstacles arising from the fundamental structural variances between these two cipher categories. A technical exposition of the computational challenges and analysis is presented as follows:

- Stream ciphers operate on the principle of encrypting plaintext sequentially - bit by bit or byte by byte - thereby producing a keystream of equivalent length to the plaintext. This continuous output stream lacks the block-wise structure that might provide discrete and consistent patterns for ML models to learn and exploit.
- The security of stream ciphers is highly contingent on initial conditions, such as the secret key and initialization vector. Even minute alterations in these initial parameters can result in a drastically altered keystream, a phenomenon analogous to the 'butterfly effect' in chaos theory. This sensitivity introduces an additional layer of complexity for ML models, which would require an enormous amount of data to learn the nuances of such a highly dynamic system.
- Stream ciphers are designed to generate highly complex keystreams that approximate the properties of a random sequence. The high entropy of the output - intended to ensure

cryptographic security - poses a significant challenge for ML algorithms, which typically require discernible patterns or statistical anomalies to make accurate predictions. The apparent randomness of the keystream complicates the task of identifying features that are indicative of the underlying encryption process.

These computational idiosyncrasies of stream ciphers make them a less tractable target for ML-based cryptanalysis compared to block ciphers, whose fixed-size blocks and iterative rounds of well-defined transformations present a more patterned and structured environment for ML algorithms to analyze and learn from.

5. Some Experimental Scenarios

Experimental scenarios on some state-of-the-art methodologies are provided in this section. The experiments are performed in perspectives of Monoalphabetic ciphers' cryptanalysis and polyalphabetic ciphers' cryptanalysis using slippery hill climbing, Particle Swarm Optimization (PSO) based cryptanalysis, and Deep Neural Networks.

5.1. Monoalphabetic ciphers' cryptanalysis

The frequency distribution of individual characters remains constant, meaning that a specific letter in the ciphertext appears the same number of times as it does in the corresponding plaintext. Repeated patterns in the plaintext also show up in the ciphertext. Often, the cryptanalyst combines two approaches, i.e. (Statistical analysis and Pattern Recognition), and supplements them with systematic guesses. One of such experimental scenarios is depicted below.

#	Message
0	Hwduythzwwjshnjx wzs ts f inxywngzyji uzgqnh qjiljw hfqqji gqthphmfns
1	Gvctxsgyvvirgmiw vyr sr e hmwxvmfyxih tyfpmg pihkiv geppih fpsgoglemr
2	Fubswrfxuuhqflhv uxq rq d glvwulexwhg sxeolf ohgjhu fdoohg eorfnfkdlq
3	Etarvqewttgpekgu twp qp c fkuvtkdwvgf rwdnke ngfigt ecnngf dnqemejckp
4	Dszqupdvssfodjft svo po b ejtusjcvufe qvcmjd mfehfs dbmmfe cmpdldibjo
5	**Cryptocurrencies run on a distributed public ledger called blockchain**
6	Bqxosnbtqqdmbhdr qtm nm z chrsqhatsdc otakhb kdcfdq bzkkdc aknbjbgzhm
7	Apwnrmasppclagcq psl ml y bgqrpgzsrcb nszjga jcbecp ayjjcb zjmaiafygl
8	Zovmqlzroobkzfbp ork lk x afpqofyrqba mryifz ibadbo zxiiba yilzhzexfk
9	Ynulpkyqnnajyeao nqj kj w zeopnexqpaz lqxhey hazcan ywhhaz xhkygydwej
10	Xmtkojxpmmzixdzn mpi ji v ydnomdwpozy kpwgdx gzybzm xvggzy wgjxfxcvdi
11	Wlsjniwollyhwcym loh ih u xcmnlcvonyx jovfcw fyxayl wuffyx vfiwewbuch
12	Vkrimhvnkkxgvbxl kng hg t wblmkbunmxw inuebv exwzxk vteexw uehvdvatbg
13	Ujqhlgumjjwfuawk jmf gf s vakljatmlwv hmtdau dwvywj usddwv tdgucuzsaf
14	Tipgkftliivetzvj ile fe r uzjkizslkvu glsczt cvuxvi trccvu scftbtyrze
15	Shofjeskhhudsyui hkd ed q tyijhyrkjut fkrbys butwuh sqbbut rbesasxqyd
16	Rgneidrjggtcrxth gjc dc p sxhigxqjits ejqaxr atsvtg rpaats qadrzrwpxc
17	Qfmdhcqiffsbqwsg fib cb o rwghfwpihsr dipzw zsrusf qozzsr pzcqyqvowb
18	Pelcgbpheerapvrf eha ba n qvfgevohgrq choyvp yrqtre pnyyrq oybpxpunva
19	Odkbfaogddqzouqe dgz az m puefdungfqp bgnxuo xqpsqd omxxqp nxaowotmuz
20	Ncjaeznfccpyntpd cfy zy l otdectmfepo afmwtn wporpc nlwwpo mwznvnslty
21	Mbizdymebboxmsoc bex yx k nscdbsledon zelvsm vonqob mkvvon lvymumrksx
22	Lahycxldaanwlrnb adw xw j mrbcarkdcnm ydkurl unmpna ljuunm kuxltlqjrw
23	Kzgxbwkczzmvkqma zcv wv i lqabzqjcbml xcjtqk tmlomz kittml jtwkskpiqv
24	Jyfwavjbyylujplz ybu vu h kpzaypibalk wbispj slknly jhsslk isvjr johpu
25	Ixevzuiaxxktioky xat ut g joyzxohazkj vahroi rkjmkx igrrkj hruiqingot
26	Hwduythzwwjshnjx wzs ts f inxywngzyji uzgqnh qjiljw hfaqji gathphmfns

5.2. Polyalphabetic ciphers' cryptanalysis using slippery hill climbing

T. Kaeding[6], discussed slippery Hill-climbing technique for ciphertext-only cryptanalysis of periodic polyalphabetic substitution ciphers. The explanation of this approach is given in section 2.1. The obtained experimental results are shown as follows.

Input Message String:

```
CILPUOYGKEQXLACGYVNATJCWIVFDELUKGICXMDMXNULIULQHGEJIPMLPLYRQEKLR
WAISGXUMZVYYIALNOCELTWAMFNRGBMOCWCSITYVPKKIMUYPAHUFFTLBEJGIRTWDC
UGDMKXAFHORITKZLDOVNATACDPUBGBSJWIVPTBAUDUASHMHQZGCIWXKGGVYCSXTM
OKKGDGIFDTXCLTZILPBCSMVYZGSMUFLAKCEGRTSBHVFLPMVPVCEBPETMVVRLNHAF
HTKMLTYBVYFSAWOYYGUGHMBPEGUYIELYVVFLTHMRKGDRWXTYFJZLTKFUDUFJSTUB
```

```
WIVQPIWCUUNCGXURKCGNNPVPNKEELBAFLVSSIFLAKCEGRTSKHEYYCBZKVERLQXLY
VKVPIHDMUMNGIAAFDPVITVAPRPZAHTZWRWTYCLLCKQNCKXYWWIZLVBZARPECRMLB
DLLBXVPMXUUMHXVDVWGCGZSSHHZVTWTMVVFDIALKHEYYCBZKVKENATJCMWIRAXHT
LPXSHPPRKVYCEKVZOGDMUPOYWVFBDPPRKCYCPIVDRNUYCWBLVVRZAXLVSNFQXOLQ
QQIKPESWZGNMIEKFDXVNATJCGQECDKAURQWMIKVUQFVRDGHRRTJYCWWYFMVBDNAR
KGGIPVLULVYQPGKZDIJYCWIPHGQCQEVANUKMRHURDKERWXLVSNFQXHUYQFZRWBUI
HXVLWTYPBOZEWMOYYGSCTGIMQXZLRXKRKCKRWTAGVYYYIPLQKQLISWVGQVYGHVHQ
HDLRIGMMUVLLPMLIBVYCTQWIRUZTTLDCUGEMIMOCRPCWIAPLJYVADNSBVGVGCLPB
HVYCHTMCVKKRXGNSQFVPIALKZCJIYHEPKENRAZUVMNYVADNSBQVJCTTUWGGKYXEZ
ZXVYYKBUEJQKRWBZDDTZRHXLKHFTPPSFRRFVQIKVWLVRLNMOGQIKFXLDCONGPDML
AWGUKILAFDXVZTXUGPRFPITURDVIMBXAGPGRLSPLLHGUCSMVIQQNUWTARKKIKXZO
RDNCFPOLZHGEYQHBRDHKCGTXSLEBAPESZDEBRDALYGSLYGMLPVIRPGRNMWCGNGHC
YOVFADGAGQWVRGRPLIVFAGTJIWIVQPYLFHMECLFLRRQNCAEAMDUBKTMVIHCMCQNA
RKGUMRNTCQVIRTTTPHVICPMLBWQRQPYLBLUKYCVLBRYERWXIMUTZBDKDGWIDMHMV
DWIVEJITYBVNVYKBUEDUDYAEACDOKMLTAAKQMCGNZYVYVUDKRCGQECDYAFHOYYSC
VGQGURWXZYVHIMBMOCVCGNTKZYQFNYHTUCARVPXXUAHFSMBUKGVRFQPELVSGIRHH
OCZQIITWDGWILQLTAAKKEETALCARCMHBCCVHFPPGFAKCEETLTMQKKMGBUEYKSPPM
PMQCEBIXTNHTRRIKLYQFXCCXYYONPITXWGQILQRTSKLVKYZXZYVRVAXTSRHOGCGT
TCQVKMSHAFDVNMGDMSONKGBXHLGYVUTKLEOCUMUMOCFQDNPGFYQFKFTXENHTKGHX
HDWGIQTOLLWGVLWHBPVYVUTKLEHVKGCZAGUGUYCWHQHKXFIXLLDRGPDTIFHFZZTZ
HLWQNMCWLPLHNCLHBJGORLPZLRRURTTMOCSCGCGLISWGMCCMBYO
```

Best Plaintext:

```
ARNABHADINSISTEDWEPLACETHEHOLEWHEREITWOULDNTBESEENLATERIWASPRETT
YSURENOWTHATHEPLANNEDTOCLAIMTHETABLEFORHISOFFICESOHESUGGESTEDWER
EMOVETHEMAKERSNAMEPLATEANDDRILLTHEREITWASJUSTASWELLHEDTDTHEDEMOL
ITIONCHARGEWASLINKEDTOAWEBOFMECHANICALDETONATORSANDALMOSTANYOTHE
RTOWARDSWOULDHAVEDISTURBEDATLEASTONEOFTHEMTHEMACHINERYWASOLDANDT
HESAPPERSWERENTHAPPYWORKINGWITHITBUTMECHANICALMECHANISMSCANBEEAS
IERTOWORKWITHTHANELECTRONICSASYOUCANSEEHOWEVERYTHINGISCONNECTEDA
JUDICIOUSDOSEOFSUPERGLUEFIXEDMOSTOFTHEMECHANISMSINPLACEFUSTLEAVI
NGUSWITHTHEPROBLEMOFWHATTODOWITHAHEAPOFOLDANDUNSTABLEEXPLOSIVESN
ORMALLYWEWOULDHAVEPLACEDONEORTWOOFOUROWNDETONATORSANDPACKEDOUTTH
EPLACEWITHSANDBAGSANDBREEZEBLOCKSTOCONTAINTHEEXPLOSIONANDITHINKE
VENHARRYMIGHTHAVEBEENCONVINCEDTHATTHATISWHATWESHOULDDOINTHISCASE
BUTUNFORTUNATELYTHEEXPLOSIVESWERENOTTHEONLYTHINGWECOULDSEEINSIDE
THESAFESITTINGUNDERTHEMWASASLIMBLACKBOOKWECOULDNTSEEANYDETAILSBU
THAVINGGOTTHISFARITSEEMEDCRAZYTODESTROYITANYTHINGTHISWELLPROTECT
EDMUSTHAVEBEENIMPORTANTATSOMETIMEANDWENEEDEDTOKNOWWHATTHISMIGHTA
LLHAVEBEENABOUTAFTERAQUICKCALLBACKTOHEADQUARTERSHARRYGOTAPPROVAL
TOCONTINUETRYINGTOCRACKTHESAFEHEKNEWMETOOWELLTOASKMETOLEAVEBUTTH
EDOCUMENTSTEAMRETREATEDTOASAFEDISTANCEDOWNTHECORRIDORWITHMOSTOFT
HEGUARDSLEAVINGASMALLTEAMTOWATCHOVERUSASWEWORKEDONEOFTHEMHADJOIN
EDTHESASFROMTHESAPPERSANDWASANEXPERTENCEDBOMBDISPOSALEXPERTSOHEW
ORKEDWITHUSWATCHINGTHEEXPLOSIVESFORANYCHANGESMONITORINGVIBRATION
ANDTEMPERATUREANDGENERALLYKEEPINGUSCALMITTAKESASPECIALTEMPERAMEN
TTODOTHATWORKFULLTIMEANDWEWEREGLADOFTHECOMPANYANDTHEEXPERTISEAFT
ERSEVENTEENHOURSWEWEREGETTINGTIREDANDASEIGHTEENAPPROACHEDIBEGANT
OWONDERIFWEWOULDMANAGETOSAVETHEPAPERSBUTEVENTUAL
```

Best key alphabets:

```
[CDEFGHIJKLMNOPQRSTUVWXYZAB]
[RSTUVWXYZABCDEFGOIJKLMNHPQ]
[YZABCDEFGOIJKLMNPQRSTUVHWX]
[PQRSTUVWXFZABCDEMGHIJKLONY]
[TUVWXYZABCDEFGHIJKLMNOPQRS]
[HIJKLMNOPGRSTUVWXYZABCDEFQ]
[YZABCDEFGHIJKLMNXPQRSTUVWO]
[DEFGHMIKLINOPQRSTUVWXYZABC]
```

fitness: -9.5534

5.3 Particle Swarm Optimization (PSO) based cryptanalysis

Particle swarm optimization (PSO) [7] is a population-based, self-adaptive search optimization technique. The approach, in detail, is presented in section 2.2. An experimental instance of the obtained output is depicted as follows.

```
Key=lssahznjcja Fitness=49
Key=lssahznacja Fitness=61
Key=lssahzkacja Fitness=74
Key=lsschzkacja Fitness=84
Key=lsschzkahja Fitness=94
Key=lsschzkahca Fitness=106
Key=lsschzkahch Fitness=115
Key=nsschzkahch Fitness=122
Key=nsschxkahch Fitness=127
Key=nsschxdahch Fitness=133
Key=nsschxdwhch Fitness=151
Key=nssccxdwhch Fitness=157
Key=nssccxdwhyh Fitness=163
Key=nssccxdwhyd Fitness=171
Key=assccxdwhyd Fitness=185
Key=azsccxdwhyd Fitness=198
Key=azbccxdwhyd Fitness=214
Key=azbycxdwhyd Fitness=248

Coputationally Empirical Key=azbycxdwhyd

Plain Results: your summarization should include a brief definition of the
type of problem being solved by the swarm intelligence algorithm as well as
discussing how a candidate solution is represented by a particle in the case
of pso or an ant in the case of aco and how the evaluation of a candidate
solution is compute d by a fitness quality function. in the case of aco using
a local heuristic function, this should be discussed too. your essay must
also discuss the stre ngths and weaknesses of the discussed types of swarm
intelligence algorithms in the context of the application or type of problems
addressed in your essa y. note that there is no need to explain, in this
short essay, concepts or applications that were already explained in the
lectures.
```

5.4 Attack on Round-Reduced Speck32/64 Using Deep Learning (Gohr's Approach)

In Crypto 2019, Gohr [16] introduced a method for performing multiple differential cryptanalysis on the SPECK cipher, specifically focusing on an input difference of $\Delta_{\text{in}}= 0\text{x}0040/0000$. This approach aimed to simplify a distinguishing problem by converting it into a classification issue. The details of this methodology are elaborated in section 4.1. To generate training and validation data, Gohr used the Linux random number generator (/dev/urandom), ensuring uniformly distributed keys K_i and plaintext pairs P_i with the specified input difference $\Delta= 0\text{x}0040/0000$. The data also included a vector of binary-valued real/random labels Y_i. Through this process, a dataset of 107 samples was created for training purposes. The optimization of this method was conducted against mean square error loss, with a minor penalty added for L2 weights regularization. This was achieved using the Adam optimization algorithm.

Computational Libraries and Simulation Setup:

Python3, TensorFlow library, and Keras were used in experimentation as an interface for TensorFlow. We have executed the experiments on a high-end workstation machine having 128 Gb RAM and (Nvidia 2080 T_i ×4) GPU cards present.

Input format for Speck 32/64:

In this neural network architecture, the first convolution block (Block 1) accepts a 4×16 matrix as input, where each row represents a 16-bit value arranged in the order: C_l, C_r, C_l', C_r'. A convolution layer with 32 filters is applied to this input. Given the kernel size of one in this 1D CNN, it converts the values C_l, C_r, C_l', C_r' into a series of filters ($filter_1$, $filter_2$,…, $filter_{32}$). These filters represent non-linear combinations of the original features, following the ReLU activation function, with the specific combinations determined by the inputs and weights learned by the 1D-CNN. The output from the first block is then connected to the input and added to the output of the next layer within the residual block. The 1D-CNNs in these residual blocks (referred to as Blocks 2-i) feature a kernel of size 3, padding of size 1, and a stride of size 1, maintaining the temporal dimension consistent across different layers.

To preserve significant input signals and prevent them from being lost between layers, the output from each layer is connected to its input and added to the output of the subsequent layer. A residual block produces a feature tensor of 32×16, which is then flattened into a 512×1 vector. This flattened vector is forwarded to the final classification block. It undergoes processing through three perceptron layers, also known as MLP. Batch normalization and ReLU activation functions are applied to the first two layers, while a final sigmoid activation function is utilized for binary classification.

ResNet module:

The initial layer of the ResNet Architecture proposed by Gohr consists of an input layer. The input layer takes in the size of (number of blocks × word_size × 2). The input is next directed to the first block, where it undergoes processing through a 1D-CNN with a kernel size of 1. This is followed by batch normalization and a ReLU activation function. Next, the output of the Block 1 is passed on to the second block. The second block is the modular unit of which we can set the number of layers. For our experiments, we set one to ten layers, each consisting of two 1D-CNN with a kernel size of 3, each followed by batch normalization and a ReLU activation function. The final classification layers are included in block 3. It comprises three perceptron layers separated by two batch normalization and ReLU functions and finished with a Sigmoid function.

Accuracy and efficiency of the Neural Distinguishers:

The neural distinguisher assigns each pair a real-valued score ranging from 0 to 1. If the score is 0.5 or higher, the sample is identified as a real pair; if it's less, the sample is labeled as a random pair. For five rounds, the ND outputs the classification accuracy of ≈ 92.61%. The performance and results are depicted in Figure 5.

```
Epoch 32/115
2000/2000 [==============================] - 34s 17ms/step - loss: 0.0574 - acc: 0.9258 - val_loss: 0.0573 - val_acc: 0.9261
Epoch 33/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0574 - acc: 0.9259 - val_loss: 0.0574 - val_acc: 0.9257
Epoch 34/115
2000/2000 [==============================] - 34s 17ms/step - loss: 0.0574 - acc: 0.9259 - val_loss: 0.0573 - val_acc: 0.9260
Epoch 35/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0573 - acc: 0.9259 - val_loss: 0.0577 - val_acc: 0.9255
Epoch 36/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0573 - acc: 0.9259 - val_loss: 0.0574 - val_acc: 0.9257
Epoch 37/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0573 - acc: 0.9259 - val_loss: 0.0573 - val_acc: 0.9259
Epoch 38/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0573 - acc: 0.9260 - val_loss: 0.0572 - val_acc: 0.9261
Epoch 39/115
2000/2000 [==============================] - 33s 17ms/step - loss: 0.0573 - acc: 0.9259 - val_loss: 0.0572 - val_acc: 0.9261
Epoch 40/115
2000/2000 [==============================] - 33s 16ms/step - loss: 0.0573 - acc: 0.9259 - val_loss: 0.0572 - val_acc: 0.9261
Epoch 41/115
```

Figure 5. An instance screenshot of results.

6. Conclusion and Consolidated Observations

Traditional symmetric cryptanalysis has seen incremental gains over time, prompting individuals to examine other options. Deep learning has lately gained a lot of attention as a result of substantial improvements in study fields like computer vision and speech recognition, thus it wasn't long before cryptography experts started to think about Deep Neural Networks (DNNs). The scope and practicality of machine learning, deep learning, and other stochastic optimization approaches for cryptanalysis are investigated in this work. This research also looks at the most recent state-of-the-art advancements in this subject.

6.1. Future Scope

The research in this direction has an enough possibility to look into. Most of the state-of-the-art approaches have considered Simon and Speck lightweight block ciphers in their experiments. So, apart from this, other ciphers' generated training data can also be considered for particular cryptanalysis exploiting DNNs. In addition to this, the language processing and translation models, Inverse Reinforcement Learning, Neural Machine Translation modeling mechanisms are also having a good potential for ciphertext-only attack.

Author Contributions: Ajeet Singh – Conceptualization, methodology, writing original draft, writing-review and editing; Kaushik Bhargav Sivangi – implementation, visualization, investigation; Appala Naidu Tentu - Formal analysis, supervision.

Funding: No funding was received for this research work.

Conflicts of Interest: Authors have no conflict of interest in this research work.

References

[1] A. Baksi, J. Breier, Y. Chen, and X. Dong, "Machine Learning Assisted Differential Distinguishers For Lightweight Ciphers," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Feb. 2021, pp. 176–181. doi: 10.23919/DATE51398.2021.9474092.

[2] J. So, "Deep Learning-Based Cryptanalysis of Lightweight Block Ciphers," *Secur. Commun. Networks*, vol. 2020, pp. 1–11, Jul. 2020, doi: 10.1155/2020/3701067.

[3] A. Jain, V. Kohli, and G. Mishra, "Deep Learning based Differential Distinguisher for Lightweight Block Ciphers," Dec. 2021, [Online]. Available: http://arxiv.org/abs/2112.05061

[4] R. L. Rivest, "Cryptography and machine learning," 1993, pp. 427–439. doi: 10.1007/3-540-57332-1_36.

[5] A. Benamira, D. Gerault, T. Peyrin, and Q. Q. Tan, "A Deeper Look at Machine Learning-Based Cryptanalysis," 2021, pp. 805–835. doi: 10.1007/978-3-030-77870-5_28.

[6] T. Kaeding, "Slippery hill-climbing technique for ciphertext-only cryptanalysis of periodic polyalphabetic substitution ciphers," *Cryptologia*, vol. 44, no. 3, pp. 205–222, May 2020, doi: 10.1080/01611194.2019.1655504.

[7] A. T. Sadiq, A. A. Ahmed, Master, S. M. Ali, and Software, "Attacking classical cryptography method using PSO based on variable neighborhood search," *Int. J. Comput. Eng. Technol.*, vol. 5, no. 3, pp. 34–49, 2014.

[8] Å. Sommervoll, "Dreaming of Keys: Introducing the Phantom Gradient Attack," in *Proceedings of the 7th International Conference on Information Systems Security and Privacy*, 2021, pp. 619–627. doi: 10.5220/0010317806190627.

[9] N. Kopal, "Of Ciphers and Neurons – Detecting the Type of Ciphers Using Artificial Neural Networks," in *Proceedings of the 3rd International Conference on Historical Cryptology HistoCrypt 2020*, May 2020, pp. 77–86. doi: 10.3384/ecp2020171011.

[10] J.-W. Chou, S.-D. Lin, and C.-M. Cheng, "On the effectiveness of using state-of-the-art machine learning techniques to launch cryptographic distinguishing attacks," in *Proceedings of the 5th ACM workshop on Security and artificial intelligence*, Oct. 2012, pp. 105–110. doi: 10.1145/2381896.2381912.

[11] M. Danziger and M. A. Amaral Henriques, "Improved cryptanalysis combining differential and artificial neural network schemes," in *2014 International Telecommunications Symposium (ITS)*, Aug. 2014, pp. 1–5. doi: 10.1109/ITS.2014.6948008.

[12] M. Abadi and D. G. Andersen, "Learning to Protect Communications with Adversarial Neural Cryptography," Oct. 2016, [Online]. Available: http://arxiv.org/abs/1610.06918

[13] R. Focardi and F. L. Luccio, "Neural cryptanalysis of classical ciphers?," in *CEUR Workshop Proceedings*, 2018, vol. 2243, pp. 104 – 115. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056878857&partnerID=40&md5=e8dfd3c9ccddf72a42ca8ea5164dfbc1

[14] A. N. Gomez, S. Huang, I. Zhang, B. M. Li, M. Osama, and L. Kaiser, "Unsupervised Cipher Cracking Using Discrete GANs," in *International Conference on Learning Representations*, Jan. 2018. [Online]. Available: http://arxiv.org/abs/1801.04883

[15] F. L. de Mello and J. A. M. Xexéo, "Identifying encryption algorithms in ECB and CBC modes using computational intelligence," *J. Univers. Comput. Sci.*, vol. 24, no. 1, pp. 25–42, 2018.

[16] A. Gohr, "Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning," 2019, pp. 150–179. doi: 10.1007/978-3-030-26951-7_6.

[17] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proceedings of the 52nd Annual Design Automation Conference*, Jun. 2015, pp. 1–6. doi: 10.1145/2744769.2747946.

[18] Y. Chen and H. Yu, "Bridging Machine Learning and Cryptanalysis via EDLCT." 2021. [Online]. Available: https://eprint.iacr.org/2021/705

[19] B. Y. Chong and I. Salam, "Investigating Deep Learning Approaches on the Security Analysis of Cryptographic Algorithms," *Cryptography*, vol. 5, no. 4, p. 30, Oct. 2021, doi: 10.3390/cryptography5040030.

[20] Z. Hou, J. Ren, and S. Chen, "Cryptanalysis of Round-Reduced SIMON32 Based on Deep Learning." 2021. [Online]. Available: https://eprint.iacr.org/2021/362

[21] H. Kimura, K. Emura, T. Isobe, R. Ito, K. Ogawa, and T. Ohigashi, "Output Prediction Attacks on Block Ciphers Using Deep Learning," 2022, pp. 248–276. doi: 10.1007/978-3-031-16815-4_15.

[22] T. Yadav and M. Kumar, "Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Cryptanalysis," 2021, pp. 191–212. doi: 10.1007/978-3-030-88238-9_10.

[23] W. Tian and H. Bin, "Deep Learning Assisted Differential Cryptanalysis for the Lightweight Cipher SIMON," *KSII Trans. Internet Inf. Syst.*, vol. 15, no. 2, Feb. 2021, doi: 10.3837/tiis.2021.02.012.

[24] Y. Chen, Y. Shen, H. Yu, and S. Yuan, "A New Neural Distinguisher Considering Features Derived From Multiple Ciphertext Pairs," *Comput. J.*, vol. 66, no. 6, pp. 1419–1433, Jun. 2023, doi: 10.1093/comjnl/bxac019.

[25] Z. Bao, J. Guo, M. Liu, L. Ma, and Y. Tu, "Enhancing Differential-Neural Cryptanalysis." 2021. [Online]. Available: https://eprint.iacr.org/2021/719

[26] J. Lu, G. Liu, B. Sun, C. Li, and L. Liu, "Improved (Related-key) Differential-based Neural Distinguishers for SIMON and SIMECK Block Ciphers." Jan. 10, 2022. [Online]. Available: http://arxiv.org/abs/2201.03767

[27] L. Zhang, Z. Wang, and B. wang, "Improving Differential-Neural Cryptanalysis." 2022. [Online]. Available: https://eprint.iacr.org/2022/183

[28] N. Băcuieţi, L. Batina, and S. Picek, "Deep Neural Networks Aiding Cryptanalysis: A Case Study of the Speck Distinguisher," in *Applied Cryptography and Network Security*, 2022, pp. 809–829. doi: 10.1007/978-3-031-09234-3_40.