ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

Simplified AES Key Bits Classification using Machine Learning Techniques

Abdurazzokov Javokhir Rustamovich

P.G. Student, Research Institute for the Development of Digital Technologies and Artificial Intelligence, Tashkent, Uzbekistan

ABSTRACT: Cryptanalysis is the field of science that determines the secret key or level of stability of a cryptographic algorithm. Currently, there are several methods for cryptanalysis of modern cryptographic algorithms. At the same time, deep learning-based cryptanalysis is being actively researched following the development of machine learning technologies. Today, cryptanalysis techniques such as plaintext attacks on light block ciphers are known. In this paper, we consider a deep learning method for key classification in machine learning techniques using a separate model for each key byte. We use this method to predict the key of the light block S-AES algorithm. And we propose a method based on machine learning techniques based on each byte of the key. The accuracy of machine learning using this method has been observed to vary by different values depending on the data set and the parameters included in it.

KEY WORDS: Simplified AES, Block Cipher, k-NN, Random Forest, Support Vector Machine, Ciphertext, Recovery Key, Cryptanalysis.

I.INTRODUCTION

The increasingly important field of cryptanalysis has its place in cryptology, which is crucial for evaluating the strength and effectiveness of encryption methods in an environment of increasing demand for secure communications and privacy. Traditional cryptanalysis, based on mathematical and statistical methods, is used to find flaws in cryptographic algorithms[1]. Advances in artificial intelligence technologies have led to deep learning-based techniques that provide new methods for cryptanalysis of encrypted data. The goal of cryptanalysis is to decrypt encrypted data without a decryption key, which reveals the weaknesses of cryptographic systems[2], [3]. The incorporation of deep learning into cryptanalysis has led to promising advances in attack strategies, increasing the accuracy and speed of breaking encrypted information. Research in this area focuses on the use of neural networks for known plaintext attacks on various encryption algorithms, where the attacker knows parts of the plaintext and the corresponding ciphertext, and attempts to discover the secret key to decrypt the remaining ciphertext [4], [5]. Research in this area also includes side-channel attacks, which exploit unintentional information leaks during the encryption process, such as power consumption or time differences[6]. The use of deep learning models has shown the potential for effective side-channel attacks against complex encryption algorithms, highlighting the importance of strong security measures to protect against these vulnerabilities [7]. Research is also being carried out into the classification and identification of encryption algorithms using neural networks. Training models on ciphertext from different algorithms has led to high accuracy in identifying the encryption techniques used [8], [9]. This ability helps you understand the strengths and weaknesses of different encryption methods and identify potential vulnerabilities. As deep learning develops in many fields, its role in cryptanalysis opens up new opportunities and challenges[10]. The results of these studies highlight the impact of deep learning models on the development of cryptanalysis, raising questions about the future of cryptographic security and the need for stronger encryption solutions. The convergence of deep learning and cryptanalysis marks a dynamic, evolving frontier with significant implications for the future of encryption and data security.

II. MACHINE LEARNING CLASSIFIERS

A) Random Forest Algorithm

Given a training dataset $D = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$, where x_i denotes the i th feature vector and y_i is the associated target value or class label:

1. For *t=1* to *T*:

Generate a random training subset D_T by applying bootstrapping to D.

Randomly select a subset of features F_T for each split.

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

Construct a decision tree T_t using D_T and F_T by recursively optimizing node splits based on impurity reduction criteria (e.g., Gini impurity or mean squared error).
2. For a given input feature vector x:
Compute predictions from each tree: $y_{tree_1}(x), y_{tree_2}(x), \ldots, y_{tree_T}(x).$
For classification tasks, predict the class label by determining the mode of the predictions (1):

$$\hat{y} = mode\{y_{tree_1}(x), y_{tree_2}(x), \ldots, y_{tree_T}(x)\} \quad (1)$$

For regression tasks, predict the target value by calculating the average (2):

$$\hat{y} = \frac{1}{T}\sum_{i=1}^{T} y_{tree}(x) \quad (2)$$

The Random Forest algorithm amalgamates the individual outputs of multiple decision trees to yield a more robust and accurate prediction. The incorporation of random data subsampling and feature selection, coupled with ensemble averaging, contributes to the control of overfitting and enhancement of generalization ability. Furthermore, the utilization of out-of-bag samples can facilitate internal validation during the training phase[11].
The ensemble nature of Random Forest renders it well-suited for diverse domains encompassing classification and regression tasks. It excels in addressing intricate data relationships while concurrently mitigating the perils of overfitting. Notably, the algorithm's efficacy can be influenced by adjustable parameters such as the number of trees T and the depth of each individual tree.

B) k Nearest Neighbours

Given a training dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where x_i represents the i th feature vector and y_i is the corresponding target value or class label:
1. For a given input feature vector x:
Calculate the distance between x and all x_i in the training dataset using a chosen distance metric (e.g., Euclidean distance).
Select the k nearest neighbors based on the calculated distances.
2. For classification tasks:

- Determine the class labels of the k nearest neighbors.
- Predict the class label for x using majority voting among the class labels of the k neighbors.

3. For regression tasks:
Retrieve the target values of the k nearest neighbors.
Predict the target value for x by calculating the average (or weighted average) of the k neighbors' target values.
The KNN algorithm assigns predictions based on the majority class among the k nearest neighbors for classification, or the average of their target values for regression. It relies on the assumption that similar data points tend to have similar outcomes. The choice of k affects the trade-off between bias and variance: smaller k values increase variance and reduce bias, while larger k values have the opposite effect.
KNN is known for its simplicity and ability to capture complex relationships in data. However, it can be sensitive to the choice of distance metric and the curse of dimensionality when applied to high-dimensional spaces. Parameter tuning, distance metric selection, and data normalization are essential considerations when employing KNN.
In summary, the KNN algorithm exemplifies the essence of local pattern recognition and can be adapted to various applications across classification and regression tasks. Its strength lies in its intuitive nature and adaptability, although thoughtful parameter choices are crucial for achieving optimal results [12] .

C) Support Vector Machine (SVM) Algorithm

Given a training dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where x_i denotes the i th feature vector and y_i is the associated class label $(y_i \in \{-1, +1\})$:

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

Optimal Hyperplane Search:

- Find the hyperplane with the largest margin that separates the classes.
- The optimal hyperplane is determined by maximizing the margin while minimizing classification error.

Support Vectors:

- Identify the data points, referred to as support vectors, that lie closest to the optimal hyperplane.
- These support vectors play a pivotal role in defining the hyperplane and influencing the classification decision.

Classification:

- For a new input feature vector x, determine the side of the hyperplane it lies on.
- Predict the class label $(-1 \; or \; +1)$ based on which side of the hyperplane x falls.

Mathematically, the optimal hyperplane is represented as: $w \cdot x + b = 0$
where w is the weight vector perpendicular to the hyperplane, x is the input feature vector, and b is the bias term. The classification decision is made based on the sign of the discriminant function (3):

$$f(x) = sign(w \cdot x + b) \tag{3}$$

where $sign$ returns −1 or +1 depending on whether the argument is negative or positive.
SVM aims to find the optimal hyperplane by solving the following optimization problem (4):

$$\min_{w,b} \frac{1}{2}\|w\|^2 \tag{4}$$

subject to $y_i(w \cdot x_i + b) \geq 1$ for all $i = 1, 2, ..., N$.

SVM can also be extended for non-linearly separable data using the kernel trick, where data is mapped to a higher-dimensional space to make it linearly separable.
In summary, the SVM algorithm seeks to find the optimal hyperplane that maximizes the margin between classes while considering support vectors. It offers a powerful framework for binary classification, even in complex scenarios, and its effectiveness is attributed to its ability to handle high-dimensional data and nonlinear relationships [13].

III. SIMPLIFIED AES ALGORITHM

Simplified AES (S-AES) is a variant of the Advanced Encryption Standard (AES) algorithm developed by Professor Edward Schaefer of Santa Clara University, primarily as an educational tool. It is designed to help students understand the complex structure of AES by using smaller cipher block sizes and key lengths. S-AES uses a uniform block size of 16 bits for both plaintext and ciphertext, accompanied by a uniform key size of 16 bits. In the encryption process, S-AES generates a 16-bit ciphertext from a 16-bit plaintext and a 16-bit key. Similarly, in decryption, it converts a 16-bit ciphertext back into a 16-bit plaintext using the identical 16-bit key. This process is illustrated in Figure 1.
S-AES is an encryption method designed to work with 16-bit data blocks. It performs both encryption and decryption using a single swap step and a simplified process with only two rounds. The key used for encryption is set to 16 bits. The structures of the encryption and decryption processes are similar, but the subkeys are used in reverse order in the encryption steps. An encryption operation takes a 16-bit plaintext and a 16-bit key, resulting in a 16-bit ciphertext. On the other hand, the decryption process, often called reverse encryption, works with the 16-bit ciphertext and the same 16-bit key, resulting in the original 16-bit plaintext. Figure 2 presents the comprehensive structure of the encryption algorithm. In the domain of S-AES, the auxiliary keys K_0, K_1, and K_2 are generated through the key expansion process, playing a crucial role in the encryption rounds. These auxiliary keys always have a 16-bit size, matching the dimensions of the plaintext or ciphertext block. During the encryption phase, the plaintext seamlessly integrates with these auxiliary keys through a series of substitution and shuffling steps, culminating in the ciphertext. Similarly, the decryption process inversely combines the ciphertext with the auxiliary keys, reconstructing the original plaintext [14].

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

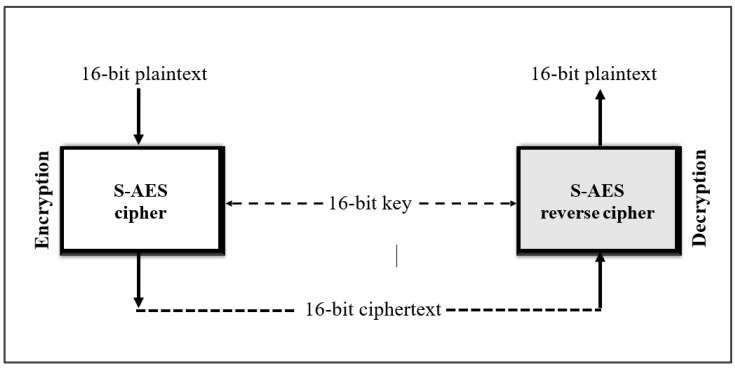

Figure 1: Encryption and decryption process of S-AES

Although S-AES strictly adheres to smaller block and key sizes, making it less formidable compared to AES, it stands out as an invaluable teaching resource. It sheds light on the fundamental architecture and principles of AES, thereby aiding in a deeper understanding of cryptographic concepts.

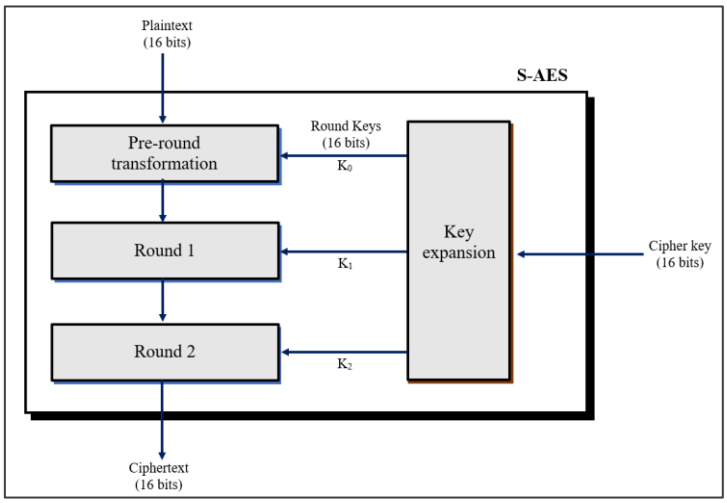

Figure 2: General design of S-AES encryption cipher

The S-AES encryption procedure meticulously adheres to a series of steps encompassing four key elements: *Nibble Substitution*, *Row Shifting*, *Column Mixing*, and *Key Addition*. These elements are skillfully executed across two rounds, with the Column Mixing step being excluded in the final round. An extra *Key Addition* phase is incorporated at the beginning, before the first round commences. Figure 3 provides a graphical depiction of the complete S-AES encryption and decryption process, encapsulated within a diagram or flowchart. This graphical illustration clearly portrays the unique role and orderly progression of each component in the sequence.

The opposite action to encryption, commonly referred to as decryption, is performed expertly to recover the original plaintext. Decryption acts as a counterbalance to encryption, using the same key but in reverse. This inversion involves reversing the series of steps taken in the encryption phase. The act of decryption successfully unravels the intertwined effects of encryption, thus providing a counter mechanism to the encryption process. Examples of S-AES encryption and decryption processes are thoughtfully presented in Table 1.

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

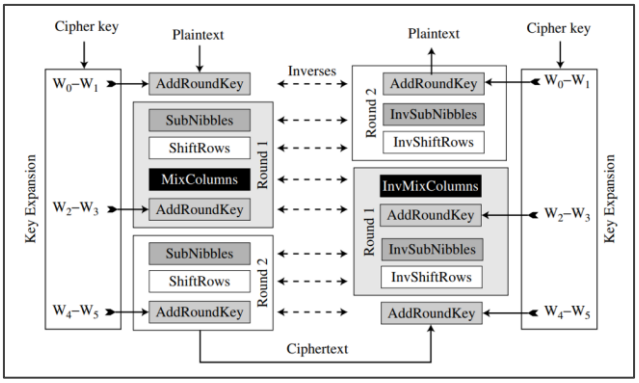

Figure 3: The S-AES Encryption and decryption process

Table 1: S-AES encryption values obtained using a randomly selected plaintext and key

№	Plaintext	Key	Ciphertext
1	0111011101110111	0011011001111001	0100000010011000
2	1110100110110100	0100000111111011	1011111110100110
3	1110100000000000	0011010100011010	1010100110100110
4	0110100010111001	0010010111110110	0101111111110101
5	0100100010111001	0001000111110100	1011000110011011

IV. METHODOLOGY

In this paper, we present an approach to extract key information of S-AES based on machine learning. For this we chose the KNN, Random Forest and SVM methods. Plaintext and ciphertext pairs are taken as input and each key bit is taken as a target bit. Data set selected: 70% for training and 30% for testing. To discover the keys, the dataset size was changed and these algorithms were trained with different sizes. At each stage of the process, the same parameters were used for each machine learning method. Figure 4 shows the scheme for generating the S-AES dataset by random selection. The process begins by randomly selecting plaintext and a key, which is then used to generate ciphertext using the S-AES encryption algorithm. In this case, the key in the algorithm is designated as *K*.

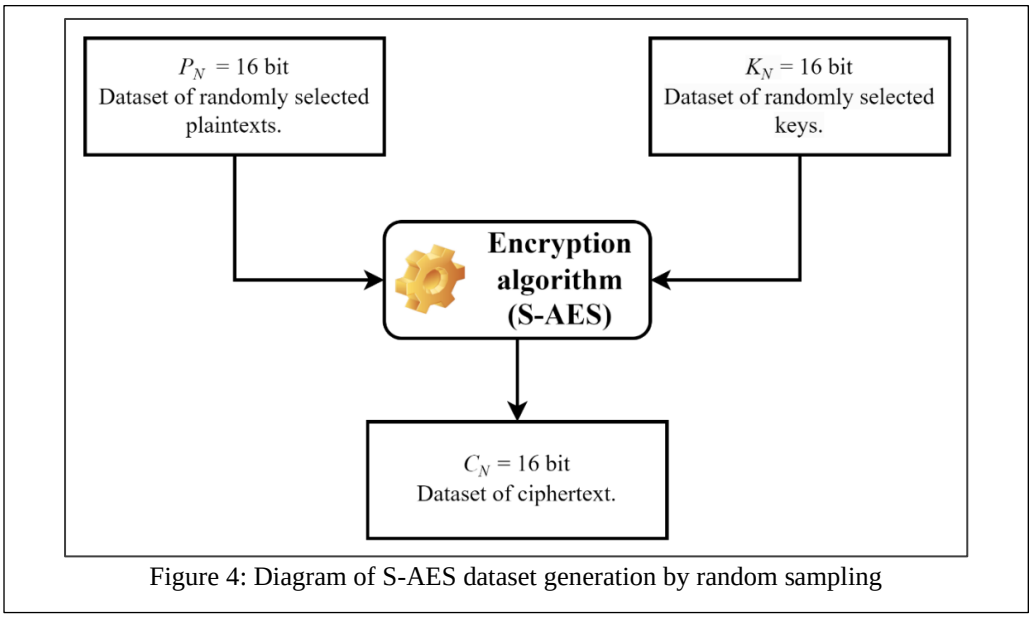

Figure 4: Diagram of S-AES dataset generation by random sampling

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

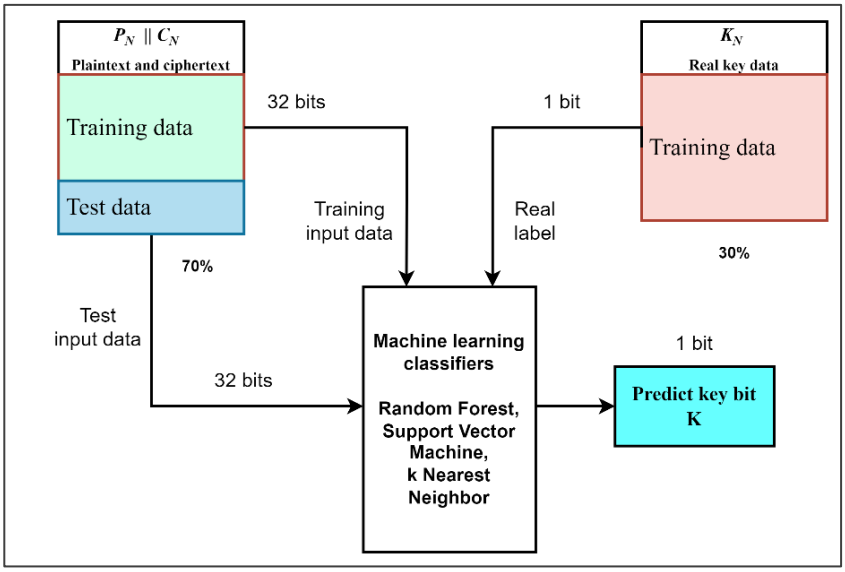

Figure 5: Diagram of the use of machine learning techniques in S-AES key classification

V. EXPERIMENTAL RESULTS

This article provides details on the S-AES key classification method, using the Python programming language and the Scikit-Learn library. Calculations were performed on an Intel i5 microprocessor computer with 24 GB of RAM and an NVIDIA RTX 3050 video card. During the execution of the attack, various training methods and parameters were used. These parameters were used sequentially in the classification process of all keys. Three different classification methods - K-NN (K-Nearest Neighbors), Random Forest and SVM (Support Vector Machine) were selected for this experiment. Each method was tested with different sizes of data sets. The results of the experiment are presented in Tables 2-6.

Table2. Key bits classification accuracy when the number of k neighbours is 2 in the k-NN method

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,473	0,500	0,553	0,487	0,487	0,487	0,487	0,513	0,460	0,527	0,520	0,527	0,460	0,553	0,480	0,453
1000	0,503	0,453	0,523	0,457	0,503	0,510	0,443	0,503	0,483	0,493	0,533	0,477	0,483	0,497	0,560	0,513
5000	0,515	0,507	0,498	0,497	0,511	0,492	0,522	0,488	0,477	0,489	0,513	0,494	0,506	0,486	0,497	0,498
10000	0,509	0,509	0,492	0,504	0,501	0,494	0,495	0,494	0,484	0,512	0,512	0,509	0,503	0,487	0,493	0,490
20000	0,496	0,505	0,493	0,497	0,496	0,504	0,504	0,510	0,501	0,504	0,498	0,490	0,496	0,495	0,504	0,508

Table3. Key bits classification accuracy when an estimator is 100 in the Random Forest method

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,520	0,493	0,493	0,547	0,480	0,513	0,547	0,467	0,533	0,500	0,500	0,460	0,413	0,520	0,533	0,493
1000	0,437	0,443	0,517	0,463	0,503	0,487	0,540	0,540	0,500	0,507	0,517	0,563	0,457	0,490	0,540	0,547
5000	0,510	0,506	0,501	0,483	0,496	0,486	0,513	0,516	0,506	0,497	0,508	0,505	0,499	0,494	0,483	0,492
10000	0,506	0,516	0,505	0,501	0,512	0,501	0,505	0,495	0,495	0,496	0,515	0,503	0,509	0,490	0,494	0,508
20000	0,508	0,498	0,512	0,513	0,501	0,499	0,500	0,489	0,496	0,497	0,502	0,484	0,500	0,513	0,493	0,508

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

Table4. Key bits classification accuracy when Kernel is linear in SVM method

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,490	0,550	0,510	0,600	0,560	0,450	0,480	0,540	0,430	0,580	0,570	0,530	0,360	0,460	0,520	0,530
1000	0,525	0,530	0,495	0,510	0,480	0,575	0,495	0,495	0,445	0,540	0,485	0,485	0,480	0,465	0,520	0,465
5000	0,533	0,492	0,493	0,513	0,498	0,503	0,502	0,481	0,512	0,518	0,496	0,508	0,524	0,500	0,520	0,477
10000	0,511	0,507	0,504	0,479	0,525	0,512	0,497	0,500	0,474	0,499	0,504	0,492	0,520	0,500	0,523	0,482
20000	0,521	0,494	0,505	0,492	0,503	0,503	0,505	0,496	0,490	0,506	0,493	0,495	0,496	0,507	0,506	0,506

Table4. Key bits classification accuracy in SVM method when Kernel RBF and gamma = 0.1

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,513	0,487	0,473	0,533	0,487	0,480	0,533	0,473	0,560	0,507	0,547	0,480	0,400	0,567	0,473	0,480
1000	0,447	0,433	0,527	0,507	0,450	0,510	0,513	0,460	0,497	0,523	0,483	0,587	0,487	0,463	0,507	0,520
5000	0,497	0,504	0,495	0,491	0,509	0,486	0,491	0,516	0,504	0,503	0,517	0,501	0,494	0,513	0,512	0,487
10000	0,506	0,514	0,507	0,504	0,496	0,507	0,510	0,493	0,485	0,497	0,511	0,500	0,506	0,505	0,508	0,498
20000	0,496	0,506	0,508	0,500	0,508	0,487	0,491	0,489	0,502	0,498	0,490	0,485	0,499	0,504	0,511	0,504

Table5. Key bits classification accuracy when Kernel Polynomial degree=4 in SVM method

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,500	0,540	0,493	0,520	0,433	0,487	0,487	0,500	0,573	0,533	0,473	0,527	0,413	0,580	0,467	0,487
1000	0,430	0,437	0,520	0,530	0,473	0,510	0,500	0,467	0,527	0,523	0,507	0,550	0,457	0,500	0,500	0,503
5000	0,511	0,503	0,482	0,499	0,527	0,503	0,493	0,502	0,480	0,497	0,515	0,503	0,513	0,511	0,495	0,487
10000	0,492	0,504	0,490	0,515	0,509	0,507	0,503	0,492	0,473	0,504	0,512	0,489	0,510	0,497	0,503	0,495
20000	0,493	0,497	0,496	0,505	0,497	0,503	0,503	0,500	0,498	0,505	0,502	0,492	0,494	0,501	0,501	0,504

Table6. Key bits classification accuracy when Kernel is sigmoid in SVM method

Sample	Accuracy															
	k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
500	0,533	0,513	0,507	0,520	0,500	0,513	0,520	0,520	0,560	0,467	0,493	0,507	0,447	0,520	0,487	0,480
1000	0,533	0,507	0,510	0,513	0,523	0,490	0,513	0,490	0,513	0,507	0,517	0,540	0,513	0,483	0,523	0,487
5000	0,499	0,509	0,507	0,474	0,500	0,501	0,505	0,511	0,490	0,497	0,494	0,472	0,491	0,486	0,505	0,479
10000	0,510	0,513	0,502	0,491	0,494	0,516	0,512	0,509	0,492	0,514	0,510	0,511	0,495	0,499	0,510	0,497
20000	0,492	0,505	0,500	0,495	0,495	0,507	0,503	0,498	0,495	0,506	0,508	0,505	0,500	0,500	0,489	0,500

VI. CONCLUSION AND FUTURE WORK

The paper discusses the use of machine learning techniques for classifying keys in the Simplified Advanced Encryption Standard (S-AES) algorithm. Three different classification methods were tested with varying parameters. The study revealed that when using a Support Vector Machine (SVM) with 500 data sets, the prediction accuracy varied for different keys: 0.60 for k3, 0.58 for k9, and 0.57 for k10. This indicates that different parameters yield different classification accuracies for each key. A notable observation from the study is that the accuracy of machine learning methods tends to decrease as the size of the data sets increases. Despite this, the authors suggest that future work could improve key classification by optimizing machine learning hyperparameters. They also emphasize the importance of having adequate training and testing data, as well as sufficient computing power, to achieve better results in key classification using machine learning techniques.

ISSN: 2350-0328
International Journal of AdvancedResearch in Science, Engineering and Technology
Vol. 10, Issue 12, December 2023

REFERENCES

[1] B. Abdurakhimov, I. Boykuziyev, and J. Abdurazzokov, "ENCRYPTION SYSTEMS AND THE HISTORY OF THEIR DEVELOPMENT," in *InterConf*, 2022. doi: 10.51582/interconf.19-20.01.2022.085.

[2] B. Abdurakhimov, O. Allanov, I. Boykuziev, and J. Abdurazzokov, "Application of artificial neural networks in the classification of classical encryption algorithms," in *2022 International Conference on Information Science and Communications Technologies (ICISCT)*, 2022, pp. 1–5. doi: 10.1109/ICISCT55600.2022.10146796.

[3] B. Abdurakhimov, J. Abdurazzokov, and L. Lingyun, "Analysis of the use of artificial neural networks in the cryptanalysis of the SM4 block encryption algorithm," *AIP Conf Proc*, vol. 2812, no. 1, p. 020048, Aug. 2023, doi: 10.1063/5.0161859.

[4] C. de Canniere, A. Biryukov, and B. Preneel, "An introduction to Block Cipher Cryptanalysis," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 346–356, Feb. 2006, doi: 10.1109/JPROC.2005.862300.

[5] J. Kim, S. Hong, J. Sung, S. Lee, J. Lim, and S. Sung, "Impossible Differential Cryptanalysis for Block Cipher Structures," 2003, pp. 82–96. doi: 10.1007/978-3-540-24582-7_6.

[6] Y. Nozaki, S. Takemoto, Y. Ikezaki, and M. Yoshikawa, "Deep Learning Based Side-Channel Analysis for Lightweight Cipher PRESENT," in *2023 15th International Conference on Computer and Automation Engineering (ICCAE)*, IEEE, Mar. 2023, pp. 570–574. doi: 10.1109/ICCAE56788.2023.10111444.

[7] X. Jin, J. Feng, and B. Huang, "Side Channel Attack on SM4 Algorithm with Deep Learning-Based Analysis," in *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, IEEE, Aug. 2022, pp. 749–752. doi: 10.1109/AEECA55500.2022.9919093.

[8] 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.

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

[10] H. Liu and B. Lang, "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey," *Applied Sciences*, vol. 9, no. 20, p. 4396, Oct. 2019, doi: 10.3390/app9204396.

[11] K. Yuan, D. Yu, J. Feng, L. Yang, C. Jia, and Y. Huang, "A block cipher algorithm identification scheme based on hybrid k-nearest neighbor and random forest algorithm," *PeerJ Comput Sci*, vol. 8, p. e1110, Oct. 2022, doi: 10.7717/peerj-cs.1110.

[12] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN Model-Based Approach in Classification," 2003, pp. 986–996. doi: 10.1007/978-3-540-39964-3_62.

[13] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, Sep. 2020, doi: 10.1016/j.neucom.2019.10.118.

[14] Ü. Çavuşoğlu, S. Kaçar, A. Zengin, and I. Pehlivan, "A novel hybrid encryption algorithm based on chaos and S-AES algorithm," *Nonlinear Dyn*, vol. 92, no. 4, pp. 1745–1759, Jun. 2018, doi: 10.1007/s11071-018-4159-4.