# CLASSIFICATION OF S-AES ENCRYPTION KEY BITS USING MULTILAYER NEURAL NETWORKS

**Boykuziev Ilkhom Mardanokulovich**
Tashkent University of Information Technologies named after Muhammad al-Khwarizmi
salyut2017@gmail.com

**Abstract.** Symmetric-key encryption plays a central role in contemporary cybersecurity, providing a fast and reliable means of protecting digital information. In this work, we explore whether the individual key bits of the Simplified Advanced Encryption Standard (S-AES) can be identified through machine learning, focusing on multilayer perceptron (MLP) neural networks. Using a dataset composed of plaintext–ciphertext pairs generated from randomly selected 16-bit keys, several neural models were trained under different hyperparameter configurations. The experiments reveal that some key bits are easier for the models to learn than others, indicating uneven sensitivity across the key space. These observations underscore the significance of proper hyperparameter tuning and point to potential applications in cryptanalysis studies.

**Keywords:** symmetric encryption, S-AES, key bit classification, neural networks, multilayer perceptron (MLP), machine learning cryptanalysis, plaintext–ciphertext pairs, deep learning, activation functions, key prediction, cryptographic security.

Symmetric encryption algorithms are widely adopted in digital security due to their computational efficiency and strong confidentiality guarantees. Their security, however, depends critically on the unpredictability and secrecy of the encryption key. The Advanced Encryption Standard (AES) is the most ubiquitous of these algorithms and serves as the foundation for numerous cryptographic protocols worldwide [1], [2].

For educational and analytical purposes, the Simplified AES (S-AES) model proposed by Schafer provides a compact yet structurally meaningful representation of AES operations [3]. Despite its reduced block size and simplified transformations, S-AES preserves essential substitution–permutation characteristics, making it suitable for examining fundamental cryptographic properties, including key scheduling, S-box behaviour, and round-based transformations [4].

Recent advancements in machine learning have motivated researchers to explore neural-network-based cryptanalysis as a means of uncovering hidden patterns in encryption systems [5], [6]. Neural models, especially deep architectures, are capable of approximating complex non-linear relationships—which raises an important question: can such models learn structural dependencies between plaintext–ciphertext pairs and the underlying secret key bits?

This study addresses that question in the context of S-AES, constructing supervised learning models designed to classify individual key bits from encrypted data. The contributions of this work are as follows:

We generate large-scale S-AES datasets of plaintext, ciphertext, and corresponding key bits for supervised learning.

We train multiple neural models with diverse activation functions, loss functions, and depths to evaluate performance sensitivity.

We analyze the extent to which specific key bits are more or less predictable by neural networks.

S-AES operates on 16 -bit plaintexts and 16 -bit keys, producing 16 -bit ciphertext outputs via two primary rounds and a preliminary key addition step. Each round makes use of subkeys derived through a simplified key schedule that mirrors the structural intent of AES while maintaining pedagogical clarity [3]. The encryption and decryption processes apply their respective subkeys in forward and reverse order, preserving symmetry in the transformation flow [4].

To construct a dataset suitable for neural classification, random 16-bit keys $K_i$ were generated as in (3.1). Likewise, random plaintexts $P_i$ were produced following expression (3.2). Each plaintext was encrypted using its corresponding key through the S-AES algorithm, yielding ciphertext values $C_i$. The final dataset thus consisted of tuples:

$$(P_i, C_i, K_i).$$

In total, 10,000 such triplets were generated and partitioned into training (70%) and testing (30%) subsets. Each individual bit $k_j$ of the key was treated as an independent binary classification target. This approach allowed us to evaluate whether certain bits were inherently easier to predict from the available data.

A series of multilayer perceptron (MLP) networks were implemented using Python, TensorFlow, and Keras. Training was performed on Google Colab with Tesla T4 GPU acceleration.

In the first experimental approach, a relatively compact multilayer perceptron was designed to evaluate whether a moderate-depth network could capture relationships between plaintext–ciphertext pairs and the underlying S-AES key bit. The model consisted of five hidden layers, containing 32, 512, 512, 256, and 128 neurons, respectively. Each hidden layer employed the ReLU activation function, while the output layer used SoftPlus, which provided smoother gradients during training. The Mean Squared Error (MSE) was selected as the loss function, and ADAM served as the optimizer. The network was trained over 200 epochs.

Despite its relatively simple structure, the model achieved exceptionally high training accuracy—often reaching a perfect score—across all key bits. However, test-set accuracy varied considerably, generally falling between 0.52 and 0.61 for the 500-sample datasets. When the training set was expanded to 20,000 samples, training accuracy remained high, yet the test accuracy still hovered near chance level, indicating that the model tended to memorize the training data rather than learn generalizable patterns within the cipher's transformations.

A second family of models was constructed to test whether deeper networks with smoother non-linearities might better approximate the underlying structure of the S-AES mapping. This architecture consisted of seven hidden layers, with neuron counts arranged

as 256, 512, 1024, 1024, 512, 128, and 64, forming a much deeper computational pipeline than in the first approach. The hidden layers used the PReLU (Parametric ReLU) activation function, chosen for its ability to adaptively learn activation slopes. The output layer employed the Sigmoid function, while MSE again served as the loss function and ADAMAX was used as the optimizer. Training was performed for 200 epochs.

This deeper architecture demonstrated noticeably better generalization. Using 500 training samples, the best results were observed for key bit k13, which achieved 99.95% training accuracy and 64.7% test accuracy, substantially outperforming the first approach. When the dataset size increased to 10,000 samples, the most predictable bit was k10, which reached 99.86% training accuracy and 57.8% test accuracy—still significantly higher than the performance of the shallower network.

These findings suggest that deeper architectures, especially those using smooth adaptive activation functions, are better suited for detecting subtle statistical residues left by simplified cryptographic transformations.

Combined interpretation of the two approaches. Across both experimental configurations, all 16 key bits were evaluated using multiple datasets and hyperparameter settings. The comparative performance shows a clear trend:

Shallow models quickly overfit and fail to generalize effectively, even with larger training sets.

Deeper models with PReLU activations capture more meaningful structure and consistently achieve higher test accuracy.

Certain key bits—particularly k13 in the 500-sample case and k10 in the 10,000-sample case—appear more learnable, implying that different parts of the S-AES key schedule may leak varying degrees of information through the encryption mapping.

Overall, the experiments confirm that the choice of neural architecture and hyperparameters has a substantial effect on key-bit predictability, emphasizing the importance of model design in machine-learning-driven cryptanalysis [7].

The experimental outcomes reveal several noteworthy patterns:

**Non-uniform bit learnability.** Certain key bits exhibit significantly higher predictability than others, suggesting that the S-AES structure does not uniformly obscure all bit positions from neural approximation.

**Model depth and activation choice matter.** Deeper networks and smoother activations (GeLU, PReLU) demonstrated consistently superior performance over shallower or ReLU-based designs.

**Training–testing divergence.** Despite extremely high training accuracy—even near-perfect—the moderate test accuracy indicates that networks learn some exploitable structure, but not enough to fully reconstruct the key.

**Implications for cryptanalysis.** While these results do not compromise S-AES, they demonstrate that machine learning can capture non-random relationships in reduced-scale ciphers. For full AES, such patterns would likely be far more subtle or nonexistent due to vastly larger key and state spaces.

**Conclusion**

This study explored the classification of S-AES key bits through supervised deep learning. Using large datasets of plaintext–ciphertext pairs, we trained multiple neural architectures and observed that specific key bits could be predicted with notable success, especially when using deeper networks and advanced activation functions.

Although S-AES is a pedagogical cipher, the findings reinforce the growing relevance of machine learning in cryptanalysis research. They also highlight the necessity of thoughtful hyperparameter optimization when conducting such studies. Future work may extend these experiments to incorporate convolutional or recurrent neural architectures, adversarial training methods, or transfer learning techniques to determine whether more refined key extraction strategies exist.

## REFERENCES:

[1] J. Daemen and V. Rijmen, *The Design of Rijndael: AES — The Advanced Encryption Standard*. Springer, 2002.

[2] NIST, "FIPS-197: Advanced Encryption Standard (AES)," National Institute of Standards and Technology, 2001.

[3] E. Schaefer, "A Simplified AES Algorithm for Educational Use," Santa Clara University, 2003.

[4] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. Pearson, 2017.

[5] G. Benamira et al., "Neural Network-Based Approaches for Cryptanalysis: A Survey," *IEEE Access*, vol. 8, pp. 145–162, 2020.

[6] S. Dubois and M. Robshaw, "Applying Machine Learning Techniques to Side-Channel Attacks," in *Proc. CHES*, 2019.

[7] A. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv:1606.08415*, 2020.