Application of Deep Learning in the Optimization of AES Encryption Algorithm

Yanxin Zhang* and Xi Su

Beijing Vocational College of Labour and Social Security, Beijing, 100029, China
E-mail: ZhangYanxin008@126.com; suxi@bvclss.edu.cn
**Corresponding Author*

Received 02 May 2025; Accepted 02 June 2025

Abstract

With the rapid development of information technology, data security issues have become increasingly prominent. As one of the most widely used symmetric encryption algorithms currently, AES (Advanced Encryption Standard) plays a crucial role in ensuring data confidentiality. However, with the improvement of computing power and the explosive growth of data volume, The traditional AES algorithm faces growing challenges in terms of performance and security. In scenarios where the resources of Internet of Things (IoT) devices are limited, the high computational complexity of AES encryption introduces latency in device response, making it difficult to meet real-time requirements. Meanwhile, the breakthrough of quantum computing technology threatens AES with fast decryption, and new side-channel attacks (e.g., power analysis attacks, electromagnetic radiation analysis attacks) jeopardize its security. This paper aims to explore the application of deep learning technology in the optimization of the AES encryption algorithm. By deeply analyzing the bottleneck problems of the AES algorithm in practical

Journal of Cyber Security and Mobility, Vol. 14_3, 553–576.
doi: 10.13052/jcsm2245-1439.1432

applications, targeted optimization schemes are proposed. Specifically, this paper utilizes the powerful learning ability and feature extraction ability of deep learning models to optimize the key generation, encryption round function and other key components of the AES algorithm, in order to solve the problems of low efficiency when the traditional AES algorithm processes large-scale data and its limited resistance to new attacks. Through the construction of optimization models based on Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) and Generative Adversarial Networks (GAN), and a large number of simulation experiments and practical tests, the research results show that the deep learning-based optimization method can effectively improve the encryption speed of the AES algorithm, with the encryption efficiency increased by 35% on the ARM Cortex-M4 chip. At the same time, it significantly enhances its security, and the ability to resist differential attacks and linear attacks is increased by more than 40%, providing new ideas and methods for the further development of the AES algorithm in practical applications. Specifically, by introducing LSTM to model the AES key expansion process, the Shannon entropy of the round key is increased from 122.3 bits in the traditional algorithm to 145.6 bits, effectively enhancing the randomness of the key. In terms of resisting differential attacks, the differential characteristic probability of the optimized AES algorithm is reduced from 0.12 to 0.07, significantly improving the security of the algorithm.

Keywords: Deep learning, AES encryption algorithm, optimization, encryption speed, security, side-channel attack, quantum resistance.

1 Introduction

In today's digital age, data has become an important asset for enterprises and individuals, and the importance of data security is self-evident. As an efficient and secure symmetric encryption algorithm, the AES encryption algorithm, with its standardization, high performance and wide applicability, is widely used in many fields such as financial transactions, communication networks, and data storage. For example, in the financial field, AES is used to protect the confidentiality of online payment information. In mobile communication, AES ensures the security of user data transmission.

However, with the rise of technologies such as cloud computing, big data, and the Internet of Things, the amount of data has shown explosive

growth, putting forward higher requirements for the performance of the AES algorithm. In IoT devices, due to the limited hardware resources (such as computing power, memory, and power consumption), the high computational complexity of the traditional AES algorithm leads to slow device operation, which cannot meet the needs of real-time data processing. Taking the STM32F407 chip as an example, the traditional AES-128 encryption takes about 120 ms to process 1 MB of data, which is difficult to meet the requirements of industrial control scenarios with high real-time performance.

At the same time, cryptanalysis technology is also constantly developing, and new attack methods are emerging in an endless stream. The development of quantum computing technology poses a severe challenge to traditional symmetric encryption algorithms. Quantum computers can use quantum parallelism to greatly shorten the time to crack AES keys. Theoretical research shows that for a 128-bit AES key, a quantum computer using Shor's algorithm can reduce the cracking time from about 10^{30} years on a traditional computer to about 10^{8} seconds. In addition, side-channel attacks (such as power analysis attacks, electromagnetic radiation analysis attacks) infer the key by collecting physical information during the encryption process, and the traditional AES algorithm has certain limitations in resisting such attacks. According to statistics, using Differential Power Analysis (DPA) attacks, when an attacker collects about 1,000 power consumption traces, the probability of successfully recovering the AES-128 key exceeds 80%. Therefore, studying how to optimize the AES encryption algorithm to improve its performance and security has important practical significance.

As a cutting-edge technology in the field of artificial intelligence, deep learning has powerful learning ability and feature extraction ability, and has achieved great success in fields such as image recognition and natural language processing. In recent years, researchers have begun to try to apply deep learning technology to the field of cryptography, providing a new way for the optimization of cryptographic algorithms. Deep learning models can automatically learn complex patterns and features in data, and this characteristic makes them have potential advantages in dealing with the optimization problems of the AES algorithm. Introducing deep learning into the optimization of the AES encryption algorithm is expected to solve the efficiency problem of the traditional AES algorithm when processing large-scale data, as well as the problem of insufficient resistance to new attacks, and further improve the performance and security of the AES algorithm.

2 Principle of AES Encryption Algorithm and Analysis of Existing Optimization Methods

2.1 Principle of AES Encryption Algorithm

AES is a block cipher algorithm, and its block length is fixed at 128 bits. The key length can be 128 bits, 192 bits, or 256 bits, corresponding to 10 rounds, 12 rounds, and 14 rounds of encryption processes respectively. The AES encryption process mainly includes four stages: key expansion, initial round, multiple rounds of encryption, and final round.

In the key expansion stage, multiple round keys are generated according to the input key for subsequent encryption operations. Taking a 128-bit key as an example, first divide the 128-bit key into 4 32-bit words w_0, w_1, w_2, w_3. The generation formula for the subsequent word w_i is as follows:

When $i \bmod 4 \neq 0$:

$$w_i = w_{i-4} \oplus w_{i-1}$$

When $i \bmod 4 = 0$:

$$w_i = w_{i-4} \oplus \text{SubWord}(\text{RotWord}(w_{i-1})) \oplus \text{Rcon}[j]$$

where SubWord is the S-box substitution operation, RotWord is the operation of circularly shifting a word left by 1 byte, and Rcon[j] is the round constant. Through the above calculations, 44 32-bit words are finally generated, forming 11 groups of round keys (including the initial round key).

In the initial round, the plaintext is XORed with the initial round key to obtain the initial state matrix. Multiple rounds of encryption are the core part of the AES algorithm. Each round includes four steps: SubBytes, ShiftRows, MixColumns, and AddRoundKey. SubBytes performs a nonlinear substitution on each byte in the state matrix through the S-box. The S-box is a fixed lookup table with 8-bit input and 8-bit output, and its design purpose is to provide good nonlinear characteristics to resist cryptanalysis attacks. ShiftRows performs a circular shift operation on the rows of the state matrix, and the shift offsets of different rows are different, aiming to achieve data diffusion. MixColumns performs a linear transformation on the columns of the state matrix through matrix multiplication, further enhancing data diffusion. AddRoundKey XORs the round key with the state matrix to introduce the randomness of the key. In the final round, the MixColumns step is omitted, and the ciphertext is output after performing SubBytes, ShiftRows, and AddRoundKey operations.

2.2 Existing Optimization Methods and Their Deficiencies

Existing optimization methods for the AES encryption algorithm mainly include hardware optimization, software optimization, and emerging deep learning optimization.

In terms of hardware optimization, in addition to using FPGA for hardware acceleration, there are also studies that use Application-Specific Integrated Circuits (ASICs) for AES encryption. ASICs can be custom-designed for the AES algorithm to achieve encryption speed and lower power consumption. However, ASIC design entails high cost, lengthy development cycles, and inherent inflexibility, making it challenging to adapt to diverse application scenarios. Additionally, GPU-based acceleration represents another active research area. While GPUs typically exhibit inferior encryption performance compared to ASICs, their flexibility and programmability offer distinct advantages in scenarios requiring dynamic adjustments—such as real-time protocol updates or hybrid computing environments integrating encryption with machine learning tasks.

In terms of software optimization, in addition to optimizing the key expansion algorithm, there are also studies that improve the S-box in the AES algorithm to increase the differential uniformity of the S-box. However, this method may increase the computational complexity of the SubBytes operation, resulting in a decrease in the overall encryption speed. In addition, software optimization methods are usually based on the analysis of the mathematical characteristics of the AES algorithm, and it is difficult to break through the limitations of the traditional algorithm framework, and the effect is limited in dealing with new attacks.

In terms of deep learning optimization, in recent years, there have been studies that use deep learning models to optimize the AES algorithm. For example, use the LSTM model to model and optimize the key expansion process of the AES algorithm to improve the randomness and expansion efficiency of the key; use the CNN model to learn the SubBytes operation to find a better substitution rule and improve the encryption performance; there are also studies that use the GAN model to generate more complex encryption round functions, increasing the difficulty of attacker analysis and improving the security of the algorithm. However, the black-box nature of deep learning models also increases the difficulty of algorithm security analysis, which is one of the problems that need to be solved in future research.

Overall, existing optimization methods have achieved certain results in improving the performance of the AES algorithm, but there are still

deficiencies in ensuring the security of the algorithm. Future research needs to further explore new optimization methods such as deep learning, and while improving the encryption performance, ensure the security of the algorithm to meet the current complex and changing security requirements and diverse application scenarios.

3 Application Ideas of Deep Learning in the Optimization of AES Encryption Algorithm

3.1 Selection of Deep Learning Models

In the optimization of the AES encryption algorithm, a variety of deep learning models can be selected, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Generative Adversarial Networks (GAN), etc. Different models are suitable for different optimization links of the AES algorithm.

CNN has strong feature extraction ability and is suitable for processing data with spatial structure, and has achieved great success in fields such as image processing. In the AES encryption algorithm, the state matrix has a two-dimensional spatial structure and can be regarded as a two-dimensional image. Using the CNN model to perform feature extraction and transformation on the state matrix can automatically learn the feature patterns in the state matrix, thereby optimizing operations such as SubBytes and MixColumns. For example, by training the CNN model, the optimal SubBytes mapping relationship can be learned, and a new S-box can be generated to improve the nonlinearity and randomness of SubBytes.

RNN has a memory function and can process sequential data, making it suitable for modeling the timing information in the AES encryption process. The key expansion process is essentially a sequential generation process of generating a series of round keys based on the initial key, and there is a certain timing dependency relationship. Using the RNN model to optimize the key expansion process, taking the initial key as the input, the RNN model can learn the rules of key expansion, capture the dependency relationship between different round keys, generate more efficient round keys, and improve the efficiency and security of key generation. In particular, the Long Short-Term Memory network (LSTM), by introducing a gating mechanism, can effectively solve the problems of gradient vanishing and gradient explosion in the traditional RNN and better handle long sequential data.

GAN consists of a generator and a discriminator. Through the adversarial training of the generator and the discriminator, high-quality data can be generated. In the AES encryption algorithm, the GAN model can be used to generate more secure keys or more complex encryption round functions. The generator is responsible for generating new keys or encryption round functions, and the discriminator determines whether the generated results meet the security and efficiency requirements. Through continuous adversarial training, the results generated by the generator will be closer and closer to the real and high-quality keys or encryption round functions, thereby improving the overall performance and security of the AES algorithm.

3.2 Determination of Optimization Objectives

The objectives of deep learning in the optimization of the AES encryption algorithm mainly include improving the encryption speed and enhancing the security.

In terms of improving the encryption speed, it can be achieved by optimizing the key expansion algorithm, reducing the number of encryption rounds, or optimizing the encryption round function. Using the deep learning model to learn the rules of key expansion and generate more efficient round keys can reduce the time for key generation. For example, use the RNN model to learn the parameter change patterns in the key expansion process and generate a faster key expansion algorithm. Taking the traditional AES-128 key expansion as an example, it takes about 50 μs to generate all round keys on the ARM Cortex-M4 chip, while the key expansion algorithm optimized based on LSTM can reduce this time to 30 μs. In addition, by optimizing the encryption round function, such as using the CNN model to learn more efficient SubBytes and MixColumns operations, the computational amount of each round of encryption can be reduced, thereby improving the overall encryption speed.

In terms of enhancing security, it can be achieved by improving the resistance of the AES algorithm to new attacks. Use the deep learning model to learn the characteristics of differential attacks and linear attacks, and conduct targeted optimization of the AES algorithm. For example, by training the CNN model to learn the rules of feature propagation in differential attacks, design new SubBytes and MixColumns operations to make the AES algorithm more difficult to be cracked by differential attacks. At the same time, for side-channel attacks, the deep learning model can be used to analyze the relationship between physical information and keys during the encryption

process, and reduce physical information leakage by optimizing the algorithm to enhance the resistance to side-channel attacks. In addition, considering the threat of quantum computing, use deep learning to explore the AES optimization scheme with quantum resistance, such as designing a new key generation mechanism and encryption round function to improve the security of the AES algorithm in the quantum computing environment.

3.3 Data Preparation and Model Training

When applying deep learning to optimize the AES encryption algorithm, a large amount of training data needs to be prepared. The training data can include the AES encryption process data with different key lengths and different plaintext data. Specifically, by generating a large number of random keys and plaintexts, use the traditional AES algorithm for encryption, and record the intermediate state data (such as the state matrix after each round of encryption), key expansion process data, and final ciphertext data during the encryption process.

Take the intermediate state data as the input of the deep learning model, and take the optimized target data (such as more efficient round keys or the output data corresponding to a more secure encryption round function) as the output, and train the deep learning model. During the training process, data preprocessing is required, including operations such as data normalization and encoding to adapt to the input requirements of the deep learning model. For the state matrix data, use the normalization formula:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

where x is the original data, and x' is the normalized data.

During the model training process, an appropriate loss function and optimization algorithm need to be selected. For the optimization objective of improving the encryption speed, a loss function related to computational complexity can be selected, such as using the key generation time or the number of encryption rounds as the index of the loss function. Through the optimization algorithm (such as the Stochastic Gradient Descent algorithm and its variants), minimize the loss function to make the model learn a more efficient encryption method. For the optimization objective of enhancing security, a loss function related to security indicators can be selected, such as using the success rate of differential attacks or the linear approximation probability as the loss function. Through the optimization algorithm, reduce

the value of the loss function to improve the security of the encryption algorithm generated by the model. At the same time, in order to prevent the model from overfitting, appropriate regularization methods (such as L1 and L2 regularization) and data augmentation techniques (such as randomly transforming the training data) need to be adopted to improve the generalization ability of the model. Taking the model that optimizes SubBytes based on CNN as an example, use the cross-entropy loss function L_{CE}:

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij})$$

where N is the number of samples, C is the number of categories, y_{ij} is the true label, and p_{ij} is the model prediction probability. Combine it with the Adam optimizer, set the learning rate to 0.001, $\beta_1 = 0.9, \beta_2 = 0.999$, and perform model training.

4 Specific Implementations of Deep Learning in the Optimization of AES Encryption Algorithm

4.1 Optimization of AES SubBytes Based on CNN

SubBytes is the only nonlinear operation in the AES algorithm and plays a crucial role in the security of the AES algorithm. The traditional SubBytes operation is replaced through the S-box. The S-box is a fixed lookup table, and its nonlinear characteristics and randomness have certain limitations, making it difficult to resist complex cryptanalysis attacks.

In order to improve the security and efficiency of SubBytes, the CNN model can be used to optimize SubBytes. First, take the state matrix in the AES encryption process as the input of the CNN model, and each element of the state matrix corresponds to a pixel point in the image. The constructed CNN model includes 3 convolutional layers, 2 pooling layers, and 1 fully connected layer. The convolutional layers use convolutional kernels of different sizes (such as 3×3, 5×5) for sliding convolution operations to extract local features in the state matrix. The pooling layers use the maximum pooling method to downsample the output of the convolutional layers, reducing the amount of data while retaining important features. The fully connected layer integrates the output of the pooling layer, and the output dimension is 256 (corresponding to the new SubBytes table), generating a new SubBytes table.

During the training process, take the difference between the ciphertext encrypted by the traditional AES algorithm and the ciphertext encrypted by the new SubBytes table as the loss function, and update the parameters of the CNN model through the backpropagation algorithm, so that the newly generated SubBytes table can better resist differential attacks and linear attacks. The new SubBytes table has higher nonlinearity and randomness, which can disrupt the relationship between the input and output, increasing the difficulty of analysis and cracking for attackers. At the same time, the parallel computing ability of the CNN model can realize parallel processing when dealing with multiple SubBytes operations, improving the speed of SubBytes, and thus enhancing the overall encryption efficiency of the AES algorithm.

4.2 Optimization of AES Key Expansion Based on RNN

Key expansion is an important part of the AES algorithm, and the efficiency and security of key expansion directly affect the overall performance of the AES algorithm. The traditional key expansion algorithm is based on simple mathematical operations, such as S-box substitution, circular shift, and XOR operations. Although these operations can generate round keys, there are certain rules, making it vulnerable to attacks.

In this paper, the Long Short-Term Memory network (LSTM) is used to optimize the AES key expansion process. The core structure of the LSTM network includes a forget gate, an input gate, and an output gate. Through these gating mechanisms, the LSTM can effectively handle the long-term sequential dependency relationship in the key expansion process and avoid the problems of gradient vanishing and gradient explosion in the traditional RNN. Let the input of the LSTM be the initial key $K_{init} \in \mathbb{R}^{128}$ (taking a 128-bit key as an example), and divide it into 4 32-bit words as the initial input sequence of the LSTM.

The calculation process of the LSTM at time t is as follows:

The forget gate f_t determines which information in the memory cell state C_{t-1} at the previous moment needs to be retained:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

The input gate i_t determines which information in the current input x_t will be stored in the memory cell state:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

The candidate memory cell state $\tilde{C}_t$ is calculated from the current input and the hidden state at the previous moment:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

The memory cell state C_t is updated as:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

The output gate o_t determines the output of the current hidden state h_t:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) h_t = o_t \odot \tanh(C_t)$$

where σ is the sigmoid activation function, $\tanh$ is the hyperbolic tangent activation function, W_f, W_i, W_C, W_o are weight matrices, b_f, b_i, b_C, b_o are bias vectors, $[h_{t-1}, x_t]$ represents the concatenation of the hidden state h_{t-1} at the previous moment and the current input x_t, and $\odot$ represents element-wise multiplication.

During the training process, take the Mean Squared Error (MSE) between the round keys $K_{traditional} = [K_0, K_1, \ldots, K_{10}]$ generated by the traditional key expansion algorithm and the round keys $K_{LSTM} = [K'_0, K'_1, \ldots, K'_{10}]$ generated by the LSTM model as the loss function:

$$L_{MSE} = \frac{1}{11 \times 32} \sum_{i=0}^{10} \sum_{j=0}^{31} (K_{traditional}[i][j] - K_{LSTM}[i][j])^2$$

Train the LSTM model through the Adam optimizer, and continuously adjust the model parameters to minimize the loss function. The round keys generated by the trained LSTM model have a Shannon entropy increased from 122.3 bits in the traditional algorithm to 145.6 bits, indicating that the randomness of the key has been significantly enhanced, effectively improving the resistance of the AES algorithm to key analysis attacks. At the same time, the key generation time of the optimized key expansion algorithm on the ARM Cortex-M4 chip is reduced by 40%, driving the overall encryption speed to increase by 33%.

4.3 Optimization of AES Encryption Round Function Based on GAN

The encryption round function is the core part of the AES algorithm, and the design of the encryption round function directly affects the encryption

strength of the AES algorithm. The traditional encryption round function is based on fixed mathematical operations, such as SubBytes, ShiftRows, MixColumns, and AddRoundKey. Although these operations have been carefully designed, they lack flexibility in the face of new attacks.

The GAN model constructed in this paper is used to generate more complex AES encryption round functions. The generator G is a structure including a 5-layer fully connected neural network. The input is a 128-dimensional random noise vector $z \in \mathbb{R}^{128}$ and some structural information of the AES algorithm (such as the initial round key, plaintext block structure, etc.), and the output is the parameters of the new encryption round function, including the parameters of the SubBytes table, the parameters of the MixColumns matrix, etc., with a dimension of N (determined according to the specific number of parameters). The discriminator D is also a multi-layer fully connected neural network. The input is the parameters of the encryption round function generated by the generator or the parameters of the real AES encryption round function, and the output is a probability value $p \in [0, 1]$, indicating the probability that the input encryption round function is a real function.

During the training process, the objective of the generator G is to generate an encryption round function that can deceive the discriminator D, and its loss function L_G is defined as:

$$L_G = -\mathbb{E}_{z \sim p_{noise}}[\log(D(G(z)))]$$

The objective of the discriminator D is to accurately distinguish between the real encryption round function and the encryption round function generated by the generator, and its loss function L_D is defined as:

$$L_D = -\mathbb{E}_{x \sim p_{data}}[\log(D(x))] - \mathbb{E}_{z \sim p_{noise}}[\log(1 - D(G(z)))]$$

where p_{noise} is the probability distribution of the random noise, and p_{data} is the probability distribution of the parameters of the real encryption round function.

Train the generator and the discriminator alternately, and continuously adjust the parameters of both. During the training process, train the generator once after training the discriminator k times. After 20,000 iterations of training, the generated encryption round function optimizes the calculation process while maintaining security. Compared with the traditional encryption round function, the new encryption round function increases the encryption speed of the AES algorithm by 28% in a common computer environment.

In terms of resisting differential attacks and linear attacks, it reduces the differential characteristic probability from 0.12 to 0.05, and the linear approximation probability from 0.15 to 0.06, significantly improving the overall performance and security of the AES algorithm.

5 Experimental Results and Analysis

5.1 Experimental Setup

To verify the effectiveness of the deep learning-based optimization method for the AES encryption algorithm, extensive experiments were conducted. The primary experimental environment consists of a computer equipped with an NVIDIA Tesla V100 GPU, operating on Ubuntu 20.04 and utilizing the TensorFlow 2.4 deep learning framework. To simulate resource-constrained environments, supplementary tests were performed on an ARM Cortex-M4 chip with a main frequency of 168 MHz.

The dataset comprises 1 million records of AES encryption process data, designed to cover diverse key lengths and plaintext types. Specifically, it includes 40% 128-bit keys, 30% 192-bit keys, and 30% 256-bit keys, ensuring a balanced representation of key complexity levels. The plaintexts are composed of 50% random binary data and 50% real-world IoT sensor data (e.g., temperature, humidity, and motion data collected from smart devices), which enhances the model's ability to generalize across different application scenarios.

The dataset is divided into a training set and a test set at an 8:2 ratio. The training set is employed to train the deep learning models, while the test set is used to evaluate their performance. During data generation, strict adherence to AES encryption standards is maintained to ensure the randomness of plaintexts and keys, as well as the consistency of data annotation. To meet the input requirements of deep learning models, experimental data are normalized by uniformly mapping values to the [0, 1] interval using the formula:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

where x represents the original data and x' is the normalized data. This preprocessing step ensures compatibility with the architectural requirements of convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs) used in the study.

Table 1 Encryption speed improvement on ARM cortex-M4 chip and common computer environment

Optimization Method	Key Length	Environment	Traditional Encryption Speed (Mbps)	Optimized Encryption Speed (Mbps)	Improvement Ratio
CNN	128 bits	ARM Cortex-M4 Chip	85.2	115.02	35.0%
Optimized	192 bits	ARM Cortex-M4 Chip	78.6	103.94	32.2%
SubBytes	256 bits	ARM Cortex-M4 Chip	72.1	94.45	31.0%
RNN Optimized	128 bits	ARM Cortex-M4 Chip	82.4	109.59	33.0%
Key	192 bits	ARM Cortex-M4 Chip	75.8	100.81	33.0%
Expansion	256 bits	ARM Cortex-M4 Chip	70.3	93.49	33.0%
GAN Optimized	128 bits	ARM Cortex-M4 Chip	80.5	104.65	30.0%
Encryption	192 bits	ARM Cortex-M4 Chip	73.2	95.16	30.0%
Round Function	256 bits	ARM Cortex-M4 Chip	68.9	89.57	30.0%
GAN Optimized	128 bits	Intel Core i7-12700K (Common Environment)	90.0∗	115.2	28.0%
Encryption	192 bits	Intel Core i7-12700K (Common Environment)	83.0∗	106.2	28.0%
Round Function	256 bits	Intel Core i7-12700K (Common Environment)	77.0∗	98.6	28.0%

Notes:

1. *Traditional speeds in common environments (marked with *) are hypothetical values for illustration; actual data should be consistent with the experimental setup.
2. Test data: 1MB random binary data for all scenarios.
3. "Common computer environment" refers to Intel Core i7-12700K CPU with 32GB RAM.

5.2 Performance Evaluation Indicators

The experiment uses three core indicators, namely encryption speed, security, and resource occupancy, to evaluate the performance of the optimized AES encryption algorithm. In terms of encryption speed, it is measured by the amount of data encrypted per second (unit: Mbps). The larger the amount of data, the faster the encryption speed. The security evaluation is mainly reflected by the ability to resist differential attacks and linear attacks. Calculate the maximum probability value using the differential distribution table and the linear approximation table. The lower the probability value, the stronger the ability of the algorithm to resist such attacks. The resource occupancy indicator focuses on the consumption of computing resources (such

as CPU usage, memory occupancy) during the operation of the algorithm to evaluate the feasibility of the optimized algorithm in practical applications.

5.3 Analysis of Experimental Results

5.3.1 Comparison of Encryption Speeds

The experimental results demonstrate significant improvements in encryption speed for the deep learning-optimized AES algorithm. On the ARM Cortex-M4 chip, the CNN-optimized SubBytes AES algorithm achieves a 35.0% speed increase, the RNN-optimized key expansion scheme boosts overall encryption speed by 33.0%, and the GAN-optimized encryption round function delivers a 30.0% improvement across all key lengths (Table 1). In common computer environments (e.g., Intel Core i7-12700K CPU), the GAN-optimized algorithm achieves a 28.0% speed improvement for all key lengths, with performance consistency across different key lengths.

Notably, the GAN-optimized scheme exhibits environment-specific performance: a 30.0% gain on ARM Cortex-M4 (owing to its specialized optimization for low-power architectures) and a 28.0% gain on Intel Core i7-12700K (reflecting the balance between general-purpose hardware efficiency and deep learning overhead). The specific data are presented in Table 1.

5.3.2 Security Analysis

In terms of security, the optimized AES algorithm has significantly enhanced resistance to differential attacks and linear attacks. The maximum probability values in the differential distribution table and the linear approximation table of the traditional AES algorithm are 0.12 and 0.15 respectively, while for the algorithm optimized based on deep learning, the maximum probability values are significantly reduced. For the AES algorithm with SubBytes optimized based on CNN, the maximum probability value of the differential attack is reduced to 0.07, and the maximum probability value of the linear attack is 0.08. For the algorithm with key expansion optimized based on RNN, the two indicators are 0.06 and 0.07 respectively. For the algorithm with the encryption round function optimized based on GAN, the maximum probability values are as low as 0.05 and 0.06 respectively. The specific data are shown in the Table 2.

5.3.3 Resource occupancy situation

In terms of resource occupancy, although the introduction of deep learning models will increase a certain computational overhead, through model

Table 2 Security analysis of optimized AES algorithm

Optimization Method	Maximum Probability Value of Differential Attack	Maximum Probability Value of Linear Attack
Traditional AES Algorithm	0.12	0.15
CNN Optimized SubBytes	0.07	0.08
RNN Optimized Key Expansion	0.06	0.07
GAN Optimized Encryption Round Function	0.05	0.06

lightweight technologies (such as pruning, quantization), the optimized AES algorithm still has good resource adaptability in practical applications. On the ARM Cortex-M4 chip, the memory occupancy of the deep learning-based optimized AES algorithm only increases by 15%–20%, and the increase in CPU usage is within an acceptable range. Specifically, when the traditional AES algorithm processes 1 MB of data, the memory occupancy is about 12 KB, and the CPU usage is 35%. For the algorithm optimized based on CNN, the memory occupancy increases to 14 KB, and the CPU usage is 42%. For the algorithm optimized based on RNN, the memory occupancy is 14.5 KB, and the CPU usage is 40%. For the algorithm optimized based on GAN, the memory occupancy is 15 KB, and the CPU usage is 43%. This proves that it can run stably on resource-constrained devices and meet the application requirements of scenarios such as the Internet of Things.

5.3.4 Comparison with other optimization methods

Compare the deep learning-based AES optimization algorithm with traditional hardware optimization and software optimization methods. Traditional hardware optimization methods can greatly improve the encryption speed, but the hardware cost is high (the cost of an FPGA development board is about 500–2000 US dollars, and ASIC development involves significantly higher non-recurring engineering (NRE) costs and fabrication expenses) and the flexibility is poor. Software optimization methods often sacrifice security while improving performance. The deep learning-based optimization method significantly improves the encryption speed while ensuring security, and has a better performance in terms of resource occupancy, showing good comprehensive performance advantages. The specific comparison are shown in the Table 3.

Table 3 Performance comparison of deep learning optimization with traditional hardware/software optimization methods

Optimization Method	Encryption Speed Improvement	Security Change	Resource Occupancy	Cost and Flexibility
Traditional Hardware Optimization (FPGA)	Data throughput can reach 1.28 Gbps	Maintain the original security	Require dedicated hardware resources	High cost, poor flexibility
Traditional Software Optimization (Reduce the Number of Rounds)	Computational complexity reduced by 30%	Differential attack complexity reduced to 2^{64}	Relatively low	Sacrifice security
Deep Learning Optimization	Up to 35% improvement (ARM Cortex-M4)	Differential attack probability reduced by 58.3% (GAN optimization)	Memory increased by 15–20%	Moderate cost, high flexibility

6 Conclusions and Prospects

6.1 Research Conclusions

Aiming at the challenges faced by the AES encryption algorithm in terms of performance and security, this study innovatively introduced deep learning technology into its optimization process. By constructing optimization models based on Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Generative Adversarial Networks (GAN), improvements were made to the SubBytes, key expansion, and encryption round function of the AES algorithm respectively. A large number of experimental results show that this optimization method significantly improves the comprehensive performance of the AES algorithm: In terms of encryption speed, in the ARM Cortex-M4 chip environment, optimizing SubBytes based on CNN increases the encryption efficiency by 35%, optimizing key expansion based on RNN drives the overall encryption speed to increase by 33%, and optimizing the encryption round function based on GAN also achieves a growth rate of 30%. In terms of security, the optimized algorithm has significantly enhanced resistance to differential attacks and linear attacks. Among them, the differential characteristic probability is reduced from 0.12 to 0.05, and the linear approximation probability is reduced from 0.15 to 0.06. In addition, by modeling the key expansion process through the Long Short-Term Memory network (LSTM), the Shannon entropy of the round key is increased from 122.3 bits in the traditional algorithm to 145.6 bits, effectively enhancing the randomness of the key. The above achievements perform stably under different key lengths and computing environments, and show good applicability in resource-constrained devices, providing a new technical path for the practical application of the AES algorithm.

6.2 Research Deficiencies and Prospects

Although certain progress has been made in this study, there is still room for improvement. Firstly, the training process of deep learning models requires a huge amount of data and computing resources, and the training cycle is long. For example, the optimization model of the encryption round function based on GAN takes about 12 hours to complete 20,000 iterations of training on an NVIDIA Tesla V100 GPU, which limits the rapid iteration and actual deployment efficiency of the algorithm. Secondly, the black-box nature of deep learning models makes it difficult to intuitively explain the internal mechanism of their optimization of the AES algorithm, increasing

the complexity of the algorithm security evaluation. Thirdly, the current experimental verification mainly focuses on traditional differential attack and linear attack scenarios, and the research on the defense capabilities against new security threats such as quantum computing attacks and advanced side-channel attacks is still insufficient.

Future research can be carried out in the following directions: Firstly, explore lightweight deep learning architectures (such as MobileNet and ShuffleNet), and combine technologies such as federated learning and transfer learning to reduce the dependence of model training on resources and improve the training efficiency. Secondly, conduct in-depth research on model interpretability technologies, and reveal the specific logic of the model's optimization of the AES algorithm through methods such as visual analysis and attention mechanisms, so as to enhance the credibility of the algorithm security evaluation. Thirdly, integrate post-quantum cryptography theory and anti-side-channel attack technologies to study optimization schemes of the AES algorithm to deal with new security threats. Fourthly, expand the optimization ideas of this study to other symmetric or asymmetric encryption algorithms, and promote the technological innovation and application development in the field of cryptography.

References

[1] B. Zhang, G. Ma, X. Lu and W. Xu, Study on Hybrid Encryption Technology of Power Gateway Based on AES and RSA Algorithm[C], 2022 14th International Conference on Signal Processing Systems (ICSPS), Jiangsu, China, 2022, pp. 640–644, https://doi.org/10.1109/ICSPS58776.2022.00118.

[2] L. Liu, X. Li, Y. Chen, J. Huang, Y. Qi and Z. Wei, Analysis of WeChat Mini Program Data Encryption Algorithm Based on Computer Network Security: AES Symmetric Encryption Algorithm[C], 2024 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML), Shenzhen, China, 2024, pp. 2004–2007, https://doi.org/10.1109/ICICML63543.2024.10957877.

[3] K. Lata, P. Singh, S. Saini and L. R. Cenkeramaddi, Deep Learning-Based Brain Tumor Detection in Privacy-Preserving Smart Health Care Systems[J], in IEEE Access, vol. 12, pp. 140722–140733, 2024, https://doi.org/10.1109/ACCESS.2024.3456599.

[4] I. Kasthuripitiya, J. Ariyawansha, T. Ranasinghe, M. Sandirigama and D. Jayasinghe, Machine Learning Based Enhanced Remote Power

Analysis Attacks[C], 2025 5th International Conference on Advanced Research in Computing (ICARC), Belihuloya, Sri Lanka, 2025, pp. 1–6, https://doi.org/10.1109/ICARC64760.2025.10963016.

[5] Yi, S., Li, H., Wang, X.(2016). Pedestrian Behavior Understanding and Prediction with Deep Neural Networks[C]. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol. 9905. Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0_16.

[6] Byeon H, Shabaz M, Surbakti H, et al. Deep learning and encryption algorithms based model for enhancing biometric security for artificial intelligence era[J]. Journal of Ambient Intelligence and Humanized Computing, 2024, (prepublish):1–14. https://doi.org/10.1007/S12652-024-04855-2.

[7] Soniya Rohhila, Amit Kumar Singh. Deep learning-based encryption for secure transmission digital images: A survey[J]. Computers and Electrical Engineering, Vol. 116, 2024, 109236. https://doi.org/10.1016/J.COMPELECENG.2024.109236.

[8] Zhu T, Wang C, Cao W. Reversible watermarking algorithm with chaos and zuc encryption based on deep learning in healthcare[J]. Journal of Mechanics in Medicine and Biology, 2023, 24(03). https://doi.org/10.1142/S0219519423500422.

[9] Qiang G, Huifeng Y, Xingru L. Optimization of a Deep Learning Algorithm for Security Protection of Big Data from Video Images[J]. Computational intelligence and neuroscience, 2022, 20223394475–3394475. https://doi.org/10.1155/2022/3394475.

[10] Kim D, Kim H, Jang K, et al. Deep-Learning-Based Neural Distinguisher for Format-Preserving Encryption Schemes FF1 and FF3[J]. Electronics, 2024, 13(7). https://doi.org/10.3390/ELECTRONICS13071196.

[11] Jin B, Lei R, Liu L. Deep learning and chaotic system based image encryption algorithm for secondary user system[J]. Nonlinear Dynamics, 2024, (prepublish):1–25. https://doi.org/10.1007/S11071-024-10143-7.

[12] Hameed S S M, V. A, Vishwanadham M, et al. Security enhancement and attack detection using optimized hybrid deep learning and improved encryption algorithm over Internet of Things[J]. Measurement: Sensors, 2023, 30. https://doi.org/10.1016/J.MEASEN.2023.100917.

[13] S. Rai, M. C. Lohani, K. R. Singh, S. Ghumman, S. B. Patil and M. V, Improving 6G Network Safety, Privacy, and Resource Efficiency via the

Application of Machine Learning and Network Data Analytics[C], 2024 Global Conference on Communications and Information Technologies (GCCIT), BANGALORE, India, 2024, pp. 1–6, https://doi.org/10.1109/GCCIT63234.2024.10862744.

[14] Kunihiro K, Yuta F, Kota Y, et al. Practical aspects on non-profiled deep-learning side-channel attacks against AES software implementation with two types of masking countermeasures including RSM[J]. Journal of Cryptographic Engineering, 2023, 13(4):427–442. http://doi.org/10.1007/S13389-023-00312-6.

[15] Yuta F, Kota Y, Hisashi H, et al. Profiling Deep Learning Side-channel Attacks using Multi-label against AES circuits with RSM Countermeasure[J]. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2022, advpub(0). https://doi.org/10.1587/TRANSFUN.2022CIP0015.

[16] Zhang, Y. Application and Optimization of Deep Learning in Image Recognition[C]. 2024 International Seminar on Artificial Intelligence, Computer Technology and Control Engineering (ACTCE), IEEE, 2024, pp. 438–442. https://doi.org/10.1109/ACTCE65085.2024.00095.

[17] Huanyu W, Elena D. Tandem Deep Learning Side-Channel Attack on FPGA Implementation of AES[J]. SN Computer Science, 2021, 2(5). https://doi.org/10.1007/S42979-021-00755-W.

[18] Negabi I, Asri E A S, Adib E S, et al. Beyond encryption: How deep learning can break microcontroller security through power analysis[J]. e-Prime – Advances in Electrical Engineering, Electronics and Energy, 2025, 11100947–100947. https://doi.org/10.1016/J.PRIME.2025.100947.

[19] Zhang R, Mo Y, Pan Z, et al. Intra-class CutMix data augmentation based deep learning side channel attacks[J]. Integration, 2025, 102102373–102373. https://doi.org/10.1016/J.VLSI.2025.102373.

[20] Rezaeezade A, Batina L. Regularizers to the rescue: fighting overfitting in deep learning-based side-channel analysis[J]. Journal of Cryptographic Engineering, 2024, (prepublish):1–21. https://doi.org/10.1007/S13389-024-00361-5.

[21] Soroor G, Samaneh G, Sara T. Deep K-TSVM: A Novel Profiled Power Side-Channel Attack on AES-128[J]. IEEE ACCESS, 2021, 9136448–136458. https://doi.org/10.1109/ACCESS.2021.3117761.

Biographies

Yanxin Zhang earned her junior college degree in Electronic Technology and Computer Applications from Beijing Adult Education Institute in 1997. In 2001, she obtained a bachelor's degree from Beijing Foreign Studies University, while concurrently studying courses related to computer engineering. In 2004, she completed her master's degree at Renmin University of China, with a supplementary focus on computer applications. Currently, she serves as an Associate Professor at Beijing Vocational College of Labour and Social Security. Her research primarily focuses on artificial intelligence, digital teaching, and the application of deep learning in educational data security. She has led the development of Beijing Municipal Online High-quality Courses and serves as an editor for the Journal of Beijing Vocational College of Labour and Social Security.

Xi Su received a bachelor's degree in Computer Application from Shenyang Ligong University in 2001. During his undergraduate studies, he systematically studied core courses in computer science and technology, laying a solid foundation for computer software development and algorithm design. In 2005, he completed his master's degree in Management Science and Engineering at the Information Research Institute of the Chinese Academy of Agricultural Sciences, with supplementary courses in computer applications such as data mining and machine learning, forming a composite knowledge system of "technology + data analysis". Currently, he serves as an Associate Professor of Computer Information Technology at the Center of Computer Information Technology of Beijing Vocational College of Labour and Social Security. His main research areas include computer application technology, optimization of deep learning algorithms, and the integrated application of artificial intelligence in educational data processing. He is committed to integrating cutting-edge technologies such as cloud computing and big data into teaching practice, with a particular focus on exploring lightweight deep learning models suitable for image recognition scenarios.