# Convolutional neural network based cipher algorithm attack scheme

Yichuan Wang[1,2], Xin Wang[1,2], Wenbin Hu[1,2], Xinhong Hei[1,2,*], Lei Zhu[1,2],Wenjiang Ji[1,2]

[1] Xi'an University of Technology School of Computer Science and Engineering
[2] Shaanxi Key Laboratory for Network Computing and Security Technology

**Abstract**

With the leading of technology network, the whole world becomes informationized, and the problem of encryption and decryption of sensitive information has become a hot issue for research. In recent years, due to the continuous development of deep learning technology, various methods of deep learning have been gradually applied to the field of information decryption, but the current technology is only for simple encryption methods to crack, for complex logical encryption there are still some imperfections. In this paper, we design a new model based on convolutional neural networks to decipher ciphertexts encrypted by complex symmetric encryption algorithms, and apply the model to decrypt AES ciphers, i.e., we use three convolutional neural networks to simulate the Encryptor, Decryptor and Key Solver, and encode the plaintext, ciphertext and key into the combined model to solve the fake plaintext, fake ciphertext and fake key. The deciphered fake plaintext is converted to binary for bit-by-bit comparison with the binary of the original plaintext, while the binary generated by the random algorithm is used to represent the binary generated by random guessing. The experimental results show that the hit rate of the deciphered binary plaintexts of the convolutional neural network against the original binary plaintexts tends to increase with the increase of the number of model iterations, i.e., the increase of the number of training rounds, within a certain interval, while the hit rate of the binary generated by the random algorithm against the original binary plaintexts hovers steadily around 1/2, which means that this is an on-the-fly guess. The experimental results reflect that the model designed based on convolutional neural network is more sensitive to the results of complex password decryption, and the model is able to reduce the distance in the space between the original plaintext and the decrypted plaintext. To formalize the advantages of multi-module convolutional neural networks for deciphering AES ciphers, the results are analyzed using provable security theory in cryptography, and by setting up a challenger and two adversaries, it is demonstrated that multi-module convolutional neural networks outperform randomized algorithms by a non-negligible margin.

***keyword:*** Deep learning,Convolutional Neural Network,plaintexts,ciphertexts

## 1 Introduction

Humans have been encoding confidential information since the ancient Greeks. People started to try to encrypt sensitive data that needed to be transmitted using simple encryption methods[23,24] in order to use such encoding methods to prevent the leakage of sensitive information, and in the same time, humans have tried to break these encoding methods through brute force cracking, frequency analysis, plagiarism and even spying.Currently, many organizations use many cryptographic techniques to protect data, which may include:RSA, AES, DES, MD5, ECC, IDEA, Diffie-Hellman, etc. Modern encryption techniques can provide integrity, non-repudiation and authentication. Encryption technology can play an irreplaceable role in protecting the integrity and security of information between the two sides of communication in an environment with potential threats, so it is widely used in military communication and encryption of sensitive information. Plaintext and ciphertext are data pairs with complex mapping relationships, and the so-called code-breaking is to find the corresponding plaintext in the plaintext-ciphertext data pair based on the given ciphertext. Traditional password cracking is often based on a large number of dictionaries for brute force cracking, which requires a large amount of data to try to solve, and this cracking method requires a lot of time, resulting in very low efficiency. The emergence of deep learning techniques has introduced a whole new way of thinking about code-breaking. deep neural networks are often described as "black boxes" due to their complex internal structure, and one way to understand how neural networks work internally is to study the activation patterns of neurons.Deep neural network itself is equivalent to a very complex and rich nonlinear feature mapping function, and using neural network for symmetric cipher deciphering is essentially to find an approximate mapping relationship between plaintext space and ciphertext space in the cipher space, because all encryption algorithms are one-to-one correspondence between plaintext and ciphertext, so theoretically, as long as the neural network is complex enough, it must be able to find such a Therefore, theoretically, as long as the neural network is complex enough, it must be able to find such a mapping relationship, so that this complex mapping relationship can be used instead of the complex encryption process to achieve the purpose of breaking the password. Since the current cryptographic schemes are based on computational difficulty rather than algorithmic irreversibility, they are under threat with the advances in computing technology and the emergence of new generation computers such as quantum computers. Some new cryptosystems combined with basic science, such as AES cryptography, quantum cryptography, optical cryptography, DNA cryptography, etc., are still in the laboratory stage due to their high implementation cost, and are still far from practical use [3], and the main research hotspots are still focused on the problem of how to implement public key cryptography quickly. Neural networks with high nonlinearity have attracted much attention from the cryptographic community and have formed a new research field, neural network cryptography, in which some results have moved towards more mature applications and some new findings have provided many novel directions for researchers.

## 2 Related work

This section introduces the basics of the model construction process and explains some of the terminology used in the model, starting with the root causes of the model.

### 2.1 Sketch of symmetric encryption

Symmetric encryption algorithm is an earlier encryption[36,37,38] algorithm with mature technology. In the symmetric encryption algorithm, the data sender processes the plaintext (original data) and the encryption key together with a special encryption algorithm[22,26,27,29] to make it a complex encrypted ciphertext and send it out. In most symmetric algorithms, the encryption key and the decryption key are the same, so this encryption algorithm is also called a secret key algorithm or a single key algorithm. It requires the sender and receiver to agree on a key before communicating securely[28]. The security of the symmetric algorithm[41] depends on the key. Leaking the key means that anyone can decrypt the messages they send or receive. Therefore, the confidentiality of the key is essential to the security of communication. The Vigenère cipher is an encryption algorithm [8] that uses a series of Caesar cipher to form a cipher alphabet, which is a simple form of multi-table ciphers. The shift cipher, also known as the Caesar cipher, is a simple encryption method that moves each letter in the plaintext to a fixed length position in the alphabet. At that time, Caesar used the letter in the alphabet for each letter in the message. Replace the third letter afterwards.However, this kind of password can be cracked by frequency analysis, or if you know it is In the case of shifting ciphers, exhaustive search results in a maximum of 26 possibilities. Encryption, decryption and cracking are relatively simple.The Vigenère cipher is a key ki (i∈[1,d]) given by a sequence of d letters, and ki determines the number of shifts of the i+td (t is an integer) letter. Advanced Encryption Standard AES (Advanced Encryption Standard, AES[10,11,12,34]) is one of the best encryption technologies. AES is a modern encryption technology that surpasses the problems of DES, and its efficiency exceeds triple DES[33]. AES[39,40] uses symmetric cryptography. This means that the same key is used for encryption and decryption, while asymmetric cryptography uses two different keys for encryption and decryption [3]. At present, most applications use the AES model to solve security problems. Compared with other cryptographic algorithms, AES algorithm and symmetric cryptography are more complex and difficult to crack.In order to generate evenly distributed ciphertexts, a series of link operations are added to the design of the AES encryption system. The AES algorithm is an iterative algorithm. Each iteration is called a round. When the key length is 128, 192, or 256, the total number of rounds is 10, 12, or 14, respectively. The data block size is 128 bits and is divided into 16 bytes.

## 2.2 Deep Learning and Neural Networks

Deep learning[7,13,14,16] is a branch of machine learning, which can be thought of as special neural networks with multiple hidden layers and can be classified into supervised and unsupervised learning based on the presence or absence of sample labels. With the development and gradual maturity of deep learning technology, a variety of deep learning models now exist, such as storm-buckling convolutional neural network (CNN), auto encoder(AE), recurrent neural network (RNN), deep belief network (DBN), and so on. The study of artificial neurons originated from the brain neuron doctrine, which was created by Waldeger et al. in the late 19th century in the field of biology and physiology.Artificial neural network [15,17,18,35] is an artificial network composed of a large number of processing units extensively interconnected to simulate the structure and function of the brain nervous system. These processing units are referred to as artificial neurons. An artificial neural network can be viewed as a directed graph connected by directed weighted arcs with artificial neurons [9] as nodes. In this directed graph, artificial neurons are the simulation of biological neurons, and directed arcs are the simulation of axon-synapse-dendrite pairs. The weights of the directed arcs indicate the strength of the interaction between two artificial neurons connected to each other. .

Feedforward neural networks are the first artificial neural networks invented, and they are a mathematical model to achieve artificial intelligence by simulating the neural network of biological brain. The middle layer of the model also becomes the hidden layer, and the hidden layer can be composed of multiple layers. A typical neural network as shown in the figure has three neurons in the input layer, four neurons in the hidden layer, and two neurons in the output layer. Each layer of the neural network is connected to each other by different weights, and the neurons in the same layer are not connected to each other. The process of training the model is also the process of adjusting the weights and bias values in the model, so that the model is fitted to the best effect.

The information propagation formula of the neural network is shown in equations (1), (2)

$$z^l = W^l \cdot a^{l-1} + b^l \tag{1}$$

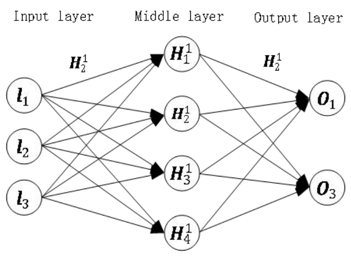


Figure 1: Artificial Neural Networks

$$a^l = f_l(z^l) \tag{2}$$

Equations (Equation (1), (2) above) can also be combined into Equation (3)

$$z^l = W^l \cdot f_{l-1}(z^{l-1}) + b^l \tag{3}$$

or combined into equations (4)

$$a^l = f_l(W^l \cdot a^{l-1} + b^l) \tag{4}$$

The neural network can be regarded as a composite function, as shown in Equation (5).

$$\emptyset(x; W, b) \tag{5}$$

The vector $x$ is used as the input a$^0$ of the first layer, and the output a$^l$ 1 of the $l$th layer is used as the output of the whole function, as shown in Equation (6).

$$x = a^0 \to z^1 \to a^1 \to z^2 ... \to a^{l-1} \to z^L \to a^L = \emptyset(x; W, b) \tag{6}$$

where $W$ and $b$ denote the connection weights and biases of all layers in the network.

## 2.3 Convolutional Neural Networks

Convolutional neural networks[1,2,3,19] were originally designed for images, so they have a unique advantage in the field of image processing, and by extending this advantage, they also have excellent performance in the field of target recognition. Convolutional neural networks can take images or process data as images and input them directly into the model, which not only eliminates the pre-processing of the input data, but also preserves all the features of the data. The feature extraction is left to the convolutional layers, and the optimal feature results are obtained by setting a reasonable number of convolutional layers. In 1980, Japanese scientist Kunihiko Fukushima initially proposed convolutional and pooling layers, which laid the foundation for the gradual layering of convolutional neural networks, and decomposed the input data into multiple features and connected them in layers, so that objects with translation, scaling, or any The convolutional neural network can be recognized even if there is any form of deformation. Since then, the convolutional neural network enters a completely new stage.

### 2.3.1 Structure of Convolutional Neural Network

The key techniques of convolutional neural networks[4] contain local perceptual fields, pooling, shared weights and biasing. For image processing pixels are used to represent that each local structure has its corresponding pixel matrix, and since matrices can be superimposed in a convolutional neural network, the pixel sub-matrices of adjacent input neurons are connected to form a local receptive field. After inputting an image, the same filter is used for scanning to extract features, and the values in the filter are the weights of the corresponding positions, a process known as weight sharing, which minimizes the number of parameters. After the convolutional layer, a pooling layer is set up to suppress noise, reduce information redundancy, and prevent overfitting phenomena.

### 2.3.2 Principle of Convolutional Neural Network

$$y = f(u) = f(\sum_{i=1}^{m} x_i \cdot w_i + \mathit{\Theta}) \tag{7}$$

As shown in Equation (7), where $X$ is the input feature vector, either the feature vector transmitted from the input layer or the feature values transmitted from other neurons. m is the number of dimensions of the feature value vector, i.e., the number of neurons, b is the bias value of the neuron, and $F()$ is the activation function, and the sigmoid function and tanh function are often chosen in convolutional neural networks. The error between the predicted output and the test output is gradually reduced by continuously correcting the magnitude of the weights, a process called back propagation. The backpropagation is implemented by the gradient descent algorithm, which is shown in Equations (8) and (9).

$$W^l_{new} = W^l_{old} - \eta \cdot \frac{\partial E}{\partial W^l_{old}} \tag{8}$$

$$b^l_{new} = b^l_{old} - \eta \cdot \frac{\partial E}{\partial b^l_{old}} \tag{9}$$

In Equation(8), (9) $\eta$ is the learning rate in the gradient descent algorithm and $E$ is the loss function. The squared error loss function is shown in Equation (10).

$$E^N = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{c} (t^n_k - y^n_k)^2 \tag{10}$$

The gradient descent algorithm solves the parameter gradients based on the loss function and then updates the weights and bias values. The gradient descent algorithm passes the gradient to each layer in turn, which in turn updates the parameters. $\delta$ is the rate of change of the error to the output as shown in Equation (11).

$$\delta = \frac{\partial E}{\partial u} \tag{11}$$

Equation (11), $\partial u$ is $W^{'} x^{l-1} + b^{'}$ , according to the chain rule can be obtained from the formula (12).

$$\frac{\partial E}{\partial b'} = \frac{\partial E}{\partial u'} \frac{\partial u}{\partial b'} = \delta \tag{12}$$

Substituting into equation (12) yields equation (13).

$$\delta^l = \frac{\partial E}{\partial b'} = \frac{\partial \frac{1}{2}(y-t)^2}{\partial b'} f(u^l)(y^n - t^n) \tag{13}$$

Equation (13) in y is a function of b,$y = f(u^{'}) = f(w^l x^{l-1} + b)$ . Further, it can be deduced that the loss function $E$ is biased against the weights $W$ in the parameters as shown in Equation (14).

$$\frac{\partial E}{\partial W'} = \frac{\partial E}{\partial u'} \frac{\partial u^{'}}{\partial W'} \tag{14}$$

The transfer equation of the reverse error is shown in Equation(15).

$$\delta^{'} = \delta^{l+1} W^{l+1} f^{'}(u^l) \tag{15}$$

In the derivation, the weights W can be replaced by the convolution kernel k, and the corresponding calculated sensitivity can be obtained by combining Equation(15) as shown in Equation(16).

$$\delta_j^l = \delta_j^{l+1} W_j^{l+1} f^{'}(u^l) = \beta_j^{l+1} up(\delta_j^{l+1}) f^{'}(u^l) \tag{16}$$

The partial derivative of the loss function with respect to b is obtained according to Equation (14) as shown in Equation (17).

$$\frac{\partial E}{\partial b_j} = \sum_{u,v} (\delta_j^l)_{u,v} \tag{17}$$

Further the bias derivative of the loss function to the convolution kernel can be obtained as shown in Equation (18).

$$\frac{\partial E}{\partial k_{ij}^l} = \sum_{u,v} (\delta_j^l)_{u,v} (p_i^{l-1})_{u,v} \tag{18}$$

The calculated sensitivity of the downsampling layer is shown in Equation (19).

$$\delta_j^l = \delta_j^{l+1} W_j^{l+1} f^{'}(u^l) = f^{'}(u_j^{l+1}) conv2(\delta_j^{l+1}, rot180(k_j^{l+1}), full) \tag{19}$$

The bias derivation of the loss function with respect to the bias value b is the same as that of the convolutional layer, and the above is the derivation process of the parameter update in the convolutional neural network.

## 2.4 Other Models

During the research of deep learning for code-breaking work, there are two other models, namely Seq2Seq model and PassGAN [6] model, which are also relevant for code-breaking aspects[42].

### 2.4.1 Deep learning code-breaking method based on seq2seq model

The seq2seq model[20,21] is a type of encoder-decoder codec, and the typical encoder-decoder structure essentially uses two RNN recurrent neural networks, one recurrent neural network as the encoder and the other recurrent neural network as the decoder. as shown in Figure 3-1 below, the encoder encoder's main The main task of encoder is to encode the input character vector into a vector of specified length, and the whole process is called encoding. The last hidden state C in Figure *-* is namely used as the semantic vector of the input sequence h0.

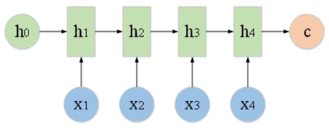


Figure 2: Seq2seq encoder

The encoder encoder and recurrent neural network are very similar. Both are computed for the hidden state C as shown in Equations (20), (21), (22).

$$c = h_N \tag{20}$$

$$c = q(h_N) \tag{21}$$

$$c = q(h_1, h_2, ...h_N) \tag{22}$$

The role of decoder is the opposite of encoder. The decoder decoder parses the hidden state C generated by the encoder encoding to generate the target sequence. The process of encoding and decoding is shown in Figure *-*. decoder takes the semantic vector C generated by encoder encoding as input, and according to the length of the target sequence, decoder parses out each character of the target sequence[30] in turn. And the output semantic vector in encoder is only used as the input of the initial state of decoder, and does not participate in the later calculation. The encoder operation converts the ciphertext sequence into a fixed-length semantic vector, and

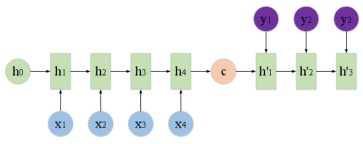


Figure 3: The Decoder structure

then the decoder operation converts the semantic vector into the target sequence, i.e., plaintext, which means that the ciphertext can be deciphered by this encoding and decoding method.

### 2.4.2 PassGAN: A Deep Learning Approach for Password Guessing

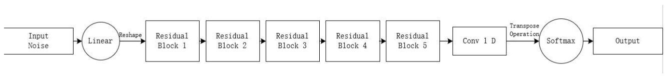


Figure 4: Generator Architecture,G

The model is based on the GAN infrastructure and consists of two deep neural networks: a generative deep neural network G and a discriminative deep neural network D. The goal of D is to distinguish between the "real samples" generated by G and the "pseudo-samples". These

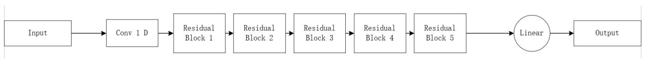


Figure 5: Discriminator Architecture,D

two neural networks influence each other by iterating over each other several times. In each iteration, the pseudo-samples from G are used as D. The output of D is provided to G, which uses the output of D as feedback to generate pseudo-samples that are closer and closer to the actual samples. After many iterations, the output of G becomes the output of the overall GAN. PassGAN motivates this technique to become the new password guessing, using real samples to train D so that the pseudo-samples of the output of the PassGAN model become close to the password distribution of the real samples, and such pseudo-samples will also match the passwords of the real users more closely. It has been experimentally demonstrated that the number of new cryptographic guesses grows solidly with the total number of ciphers generated by GAN.

## 3 The Model

Convolutional neural networks are chosen as the basis for model design in this chapter for two main reasons.

1) For simple ciphers, such as shift ciphers, the character space is very limited. The plaintext and ciphertext are concentrated on 26 English characters no matter how they are changed, and the encryption rules of simple passwords are relatively traceable.This leads to a very large mapping space for plaintext ciphertext, and the mapping rules are not available, which makes it very difficult to decipher complex passwords using machine translation model.

2) For simple cipher encryption algorithms, there may be character-to-character connections between the ciphertext sequences after plaintext encryption, because the encryption is simple and there are strong mapping rules between plaintext characters and ciphertext characters, while the convolutional kernel in convolutional neural networks, emphasizing the window in space, also need to consider the contextual issues, and a closer look at the advanced encryption standard AES shows that AES does encryption.Therefore, it is appropriate to use convolutional neural network to solve the complex encryption algorithm.

### 3.1 Model Description

TensorFlow is a relatively high-level machine learning library that allows users to easily design neural network structures with it without having to write C++ or CUDA code themselves in pursuit of efficient implementations. The code-breaking model in this chapter is nested by combining three convolutional neural networks, which are built through Tensorflow. The data set is processed using Pandas to normalize the data and input to the model in the form of a matrix, and the data is divided into a training set and a test set with a ratio of 0.9. The structure and parameters of the model are saved through the tf.train.Saver function every 10 rounds during the training process. Predictions are then made by a test machine and the results are output. The model structure is shown below, A, B and C are all three convolutional neural networks and their parameters are $\vartheta_A$,$\vartheta_B$,$\vartheta_C$ respectively. Before inputting data into the model prepare plaintext plain and key key, and generate ciphertext cipher by Virginia cipher encryption algorithm, we put plaintext plain and ciphertext cipher into the convolutional neural network A, and his calculation result will be key fake_key, we give the fake_key and ciphertext generated by A The cipher is decrypted by B to get the plaintext fake_plain, and the C convolutional neural network receives the key from A's network and the plaintext from B's network and tries to recover the ciphertext.

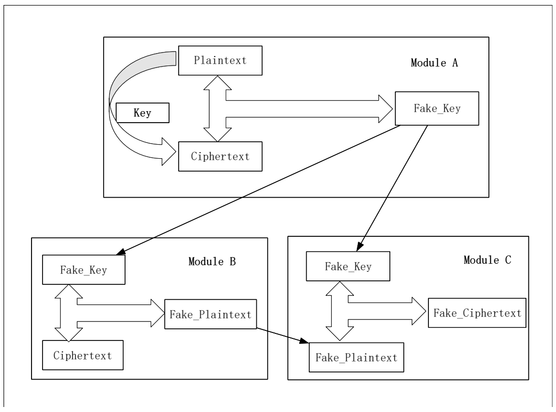


Figure 6: The Model

## 3.2 Specific Implementation

The specific steps for implementing a convolutional neural network-based method for encrypting and decrypting communication data of the present invention are as follows.

Step 1: making a plaintext dataset and processing the plaintext dataset, i.e. such dataset should have to contain all the data types that are likely to be involved.

Step 2: Choose a suitable encryption algorithm and use a fixed key key to encrypt the plaintext. This results in a number of plaintext pairs.

Step 3: Construct a convolutional neural network model; for the plaintext plain and ciphertext cipher, a pseudo-key fake_key is generated by convolution; also, two convolutional neural network models are constructed, and the pseudo-key fake_key and plaintext plain, fake_key and ciphertext cipher are used as inputs to the two neural networks, respectively. using the continuous iterative training of the convolutional neural network, and then generate the corresponding fake_cipher and fake_plain, respectively.

Step 4: Monte Carlo method is used to judge the loss function of plain and fake_plain and the loss function of cipher and fake_cipher, and make them reach the minimum i.e. the similarity of both is great, and finally the trained key can be obtained; and the encryptor and decryptor with good performance can be obtained to provide a data transmission high security[31,32] path for data transmission.

## 3.3 Model Architecture

The network architecture is as follows.(Take the B network as an example)

First a fully connected layer, the number of inputs is equal to the number of outputs, the input of the fully connected layer is the key fake_key and ciphertext cipher solved by Alice network, the fully connected layer is followed by 5 one-dimensional convolutional layers, the window size, input depth and output depth of each convolutional layer are [2, 1,4], [4,2,2], [4,1,1], and [1,1,1] respectively, in steps of 1, 2, 1, and 1. The number of vectors in the last layer after the convolutional layer is the length of the plaintext cipher. Except for the convolutional layer, the activation functions of all other layers are sigmoid functions, and the activation function of the last convolutional layer is tanh function, because the loss function is calculated with the normalized value (-1,1) unified. The specific processing flow is shown as follows.

(1) The input layer uses the fake_key+cipher solved by Alice network, and the format of Alice_output is Tensor("Alice/convlayers/conv_5/conv1d/Tanh:0", shape=(? , 8, 1), dtype=float32), you can see that the network output result is two-dimensional, in order to match the Bob network structure will output the key reshape into a one-dimensional, that is, reshape_Alice_output = tf.reshape(Alice_output, shape=[-1, keyLength]), where keyLength is the length of the key 8, and the dimension of reshape_Alice_output is Tensor("Reshape:0", shape=(? , 8), dtype=float32). Bob's input layer is a fully connected layer that splices reshape_Alice_output with cipher, i.e., Bob_input = tf.concat([reshape_Alice_output, cipher], axis=1 ). The structure of Bob_input is Tensor("Bob/concat:0", shape=(? , 33), dtype=float32).

(2) The middle convolutional layer is divided into five layers, and the parameters and structure of each layer are. The parameters of the "conv_1" layer are: fileters=2, stride=1, kernelSize=4, activation function is sigmoid, padding="SAME", and the structure is Tensor("Bob/convlayers/conv_1/conv1d/Sigmoid:0", shape=(?,33, 2), dtype=float32). The parameters of the "conv_2" layer are: fileters=4, stride=2, kernelSize=2, activation function is sigmoid, padding="valid", and the structure is Tensor("Bob/convlayers/conv_2/conv1d/Sigmoid:0", shape=(?,16, 4), dtype=float32). The parameters of the "conv_3" layer are: fileters=4, stride=1, kernelSize=1, activation function is sigmoid, padding="SAME", and the structure is Tensor("Bob/convlayers/conv_3/conv1d/Sigmoid:0", shape=(?,16, 4), dtype=float32).

(3) The parameters of the last convolutional layer "conv_4" are: filets=1, stride=1, kernelSize=1, because the value to be taken lies between (-1, 1), so the activation function is tanh, padding=" valid",structure as Tensor("Bob/convlayers/conv_4/conv1d/Tanh:0", shape=(? , 16, 1), dtype=float32), the output is the plaintext.

## 4 Model Analysis

### 4.1 Model Construction and Evaluation Metrics

This chapter is a code-breaking experiment based on convolutional neural network, and the experiment is divided into three modules, which are used to solve the key key, plain text plain and cipher text cipher, and the experiment will focus on cracking the cipher in the module of solving the plain text according to the cipher text, i.e., B module and C module, and in the process of training the model, we set the loss B_loss of B solving the plain text respectively, and the loss of plain text and loss of ciphertext are solved according to the mean square difference formula, i.e., the difference is calculated for each bit of the plaintext ciphertext, so the loss function of the B network is shown in Equation (23).

$$B\_loss = tf.reduce(tf.square(b\_out - plain)) \tag{23}$$

The C network module is based on the fake_key and fake_plain to solve the ciphertext, the same as the loss function of B network, the loss function of C network is shown in Equation (24).

$$C\_loss = tf.reduce(tf.square(c\_out - plain)) \tag{24}$$

In order to visualize the accuracy of the network of solving plaintext and the network of solving ciphertext, four evaluation indexes are set in the experiment, which are accuracy_B of B network, the number of bits_wrong of B network error, accuracy_C of C network, and the number of bits_wrong of C network error, and the calculation formula is given below, the accuracy of B network The accuracy of solving plaintext is shown in Equation (25).

$$accuracy\_B = tf.reduce(tf.cast(tf.abs(b\_out - plain) < 0.1)) \tag{25}$$

The accuracy of the C-network decryption text cipher is shown in Equation (26).

$$accuracy\_C = tf.reduce(tf.cast(tf.abs(c\_out - cipher) < 0.1)) \tag{26}$$

The statistics of the number of error bits of the B-network unplaintext are shown in Equation (27).

$$B\_bits\_wrong = plainTextLength - accuracy\_B * plainTextLength \tag{27}$$

The statistics of the number of error bits of the C-network decryption message are shown in Equation (28).

$$C\_bits\_wrong = cipherTextLength - accuracy_C * cipherTextLength \tag{28}$$

The following explains the details of the network model construction.

1) A, B and C have the same network structure, but the input, output and parameters are different.

2) The lengths of Plain, Cipher, and Key are 16, 25, and 8, respectively. They are floating-point numbers in the interval (0-84). After normalization, they are (-1, 1). In the final calculation of accuracy, Keep a significant number of floating-point numbers for comparison.

3) Each plaintext corresponds to a ciphertext.

4) Loss function: The three convolutional neural networks of A, B and C have the same calculation objective. The goal of A is to solve the fake_key as close to the real key as possible; the goal of B is to solve the fake_plain as close to the real plaintext as possible; the goal of C is to solve the fake_cipher as close as possible The real ciphertext Cipher; that is to say, the error between the result and the original data is reduced as much as possible, so the loss function is:

$$Loss = tf.reduce\_mean(tf.abs(out - text) < 0.01) \tag{29}$$

The input and output of A, B, and C neural networks are shown in the following Table**:

Table 1: A, B, and C neural network input and output

| **Network model** | Input | Output |
|---|---|---|
| **Module A** | Plaintext,Ciphertext | Fake_Key |
| **Module B** | Fake_Key,Ciphertext | Fake_Plaintext |
| **Module C** | Fake_Key,Fake_Plaintext | Fake_Ciphertext |

## 4.2 Demonstrable security theory analysis study

This experiment uses the adversary attack type of CPA, and the experiment selects the plaintext attack by encrypting the ciphertext with AES encryption over.The notion of semantic security of encryption schemes is inscribed by indistinguishability games, i.e., IND games, which are thought experiments in which there are two participants, one called the challenger and the other the adversary.

The IND game for the public-key encryption scheme under selective plaintext attack (CPA) is as follows.

1) Initialization. The challenger enters the plaintext pair and the adversary (denoted as A) gets the initial key Key for the experiment.

2) The adversary enters a plaintext message and gets a ciphertext message encrypted by the key key.

3) After a series of convolutional combinations and experimental algorithms under different choices, the adversary outputs two ciphertext pairs of the same length,$M_0$ and $M_1$. By generating a random selection of binary bits, $\beta \in$ {0, 1} encrypting $M_\beta$ and delivering the plaintext pair to the adversary after model deciphering.

4) Guessing. The adversary outputs $\beta^{'}$, and if $\beta^{'} = \beta$ , the cipher is successfully deciphered, i.e., the deciphered binary bits match the original plaintext message binary bits.

The adversary's advantage can be defined as a function of parameter K.

$$Adv_A^{CPA}(K) = \mid Pr[\beta^{'} = \beta] - \frac{1}{2} \mid \tag{30}$$

where K is the security parameter to determine the length of the encryption scheme key. Because any inactive adversary A can win the IND game by making a random guess about $\beta$, and the probability generated by the random algorithm goes to 1/2. And $\mid Pr[\beta^{'} = \beta] - \frac{1}{2} \mid$ is obtained by the adversary through his efforts, so this is called the advantage of the adversary. In the data plot of the experiment that is expressed as the value of the vertical coordinate of the point on that curve on any horizontal coordinate.

If an adversary goes through the binary bits of the ciphertext after $M_\beta$ , it is possible to distinguish whether $M_\beta$ is $M_0$ or $M_1$ , thus the IND game portrays the concept of semantic security.

**Notes.**

1) The adversary in the definition is polynomial round number, otherwise, because he has the original key of the system, he can get any multiple ciphertexts of $M_0$ and $M_1$ , and then compare them with the known ciphertext bit by bit to win the game and decipher successfully.

2) $M_0$ and $M_1$ of equal length, otherwise by the ciphertext, there exists the possibility to distinguish whether $M_\beta$ is $M_0$ or $M_1$ .

3) If the encryption algorithm is deterministic and there is only one ciphertext corresponding to each plaintext, the adversary only needs to re-encrypt $M_0$ and $M_1$ and compare them with the known ciphertext to finally win the game, i.e., successful decipherment.

4) In each encryption, a random algorithm is first selected and then the ciphertext is generated.

## 5 Experimental analysis and proof

### 5.1 Experimental environment

The environment for the experiments in this chapter is shown in Table *-*.

Table 2: Experimental environment

| | Details configuration | Resource usage |
|---|---|---|
| CPU | Intel(R) Core i7-8550U 2.0GHz | / |
| Memory | 16G | / |
| GPU | GeForce GTX 1060 | approximately 5G |
| System | 64-bit Win10 Professional Edition | / |
| Python | Python 3.6.0(64-bit) | / |
| IDE | Pycharm 2018.2.4 | / |

### 5.2 Analysis of experimental results

The experiments are mainly divided into three groups, and the difference of each group lies in the inconsistent length of the plaintext ciphertext, the reason is to compare the effect of increasing the length of the plaintext ciphertext on the cracking effect of the model. The lengths of the plaintexts in the experiments are 16, 32, and 128, corresponding to the lengths of the AES ciphertexts of 25, 45, and 175, respectively. the results of the experimental validation process are mainly verified for the plaintexts solved by the B network, so the bit lengths of the plaintexts are divided into 128, 256, and 1024 when the bit hit rate is verified. in the training process of each group of the model, the B network's The loss value B_loss_value, the error B_bits_wrong and accuracy_B_value between the plaintext and the real value, and the conversion of the solved plaintext and the real value into binary are compared to calculate the change curve of the number of bits hit with the iterative process of the model, and the comparison graph of the number of hits is used as an important basis for the model to decipher the password. Finally, the time curve of the model training is recorded.

The parameter settings for the first group of experiments are shown in Table *-*.

Table 3: The first group of parameter setting table

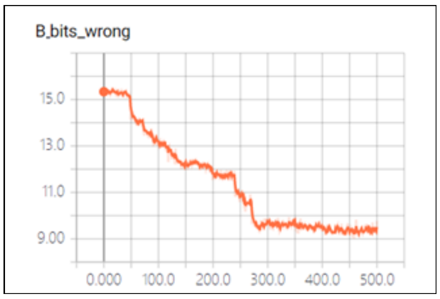
| **Parameter Name** | **Parameter Value** |
|---|---|
| plainTextLength | 16 |
| cipherTextLength | 25 |
| keyLength | 8 |

The experimental results are shown in Figure (9) below.

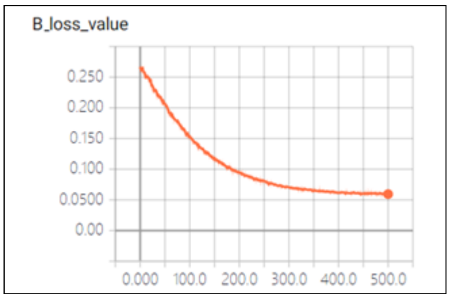
As shown in the analysis in Figure (9).

Figure (a) shows the number of error bits of the plaintext solved by the B-network in the course of 500 training iterations. length is able to solve for 7 bits.

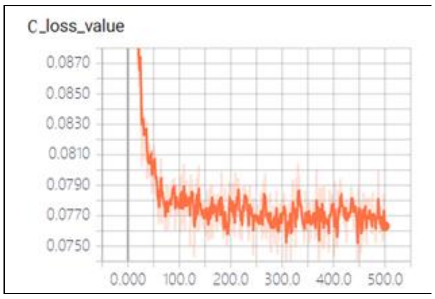
Figure (b) represents the process of gradually decreasing the loss value of the B-network, which finally converges to 0.05 or below, and there is no significant fluctuation in the decreasing process, which proves the relative stability of the network.



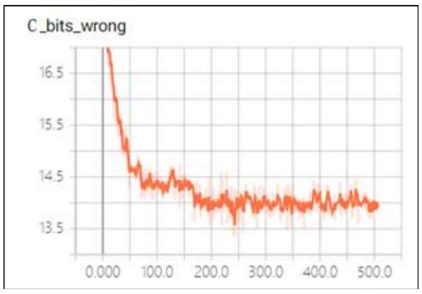
(a) Error Bit Curve of B Network Bits



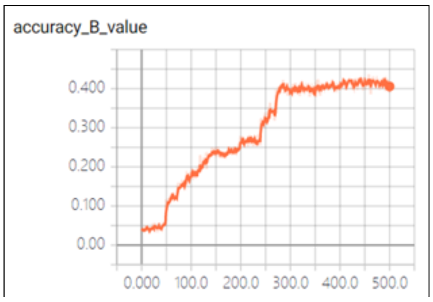
(b) B network loss curve



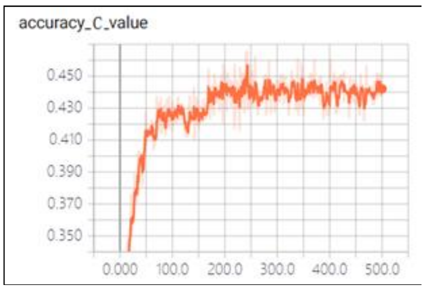
(c) C network loss curve



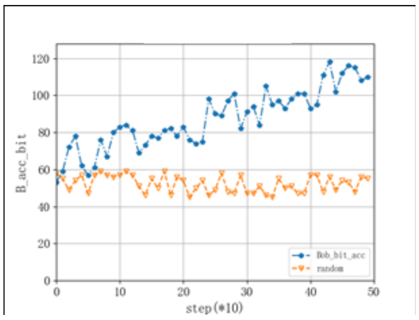
(d) Error Bit Curve of C Network Bits



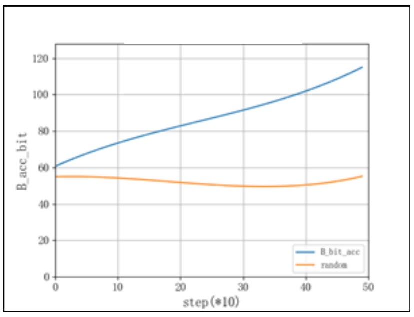
(e) B network correct rate curve



(f) C network correct rate curve



(g) Comparison of the number of bits hit by B-network and randomized algorithms



(h) Comparison of the number of bits hit by B-network and random algorithm fit

Figure 7: The first group of experimental results

Figure (c) represents the loss value between the solved ciphertext and the original ciphertext in 500 iterations of the C-network for solving the ciphertext, which can be seen to be gradually converging as well.

Figure (d) represents the curve of the number of error bits of the ciphertext solved by the C-network, the length of the ciphertext is 25 and the number of error bits finally converges to 13 bits or less.

Figures (e) and (f) show the correct rates of the plaintext solver and the ciphertext solver, respectively, looking to see that the highest correct rates of both are around 40%.

In Figure (g), there are two dashes, the yellow dash indicates the random algorithm generates 128-bit binary to represent the random guess of the binary plaintext, and it can be seen that as the number of model iterations increases, the hit rate of the 128-bit binary generated by the random algorithm on the original plaintext always hovers around 60 bits, which means that the random algorithm's cracking of the plaintext is equivalent to a blind guess, while the blue dash indicates that the B We can see that the number of hits increases gradually as the model is trained, and in the beginning of the model, the number of hits of both the model and the random algorithm is around 60 bits, while in 500 rounds, the number of hits of the model reaches up to 120 bits, which proves that the model can reduce the distance between the solved plaintexts and the original plaintexts. proving that the model is effective.

Figure (h) represents the fitted comparison plot of the number of bits hit by the B-network and the randomized algorithm, which is a fit to the two discounted items of Figure (g), and expresses the same information as Figure (g).

## 6 Conclusion

With the development of information technology, security has become a key concern. Information security is extraordinarily important to the privacy and interests of each of us, and cryptography is the core technology for protecting information security, so the problem of encryption and decryption of sensitive information has become a hot issue for research today. With the development of deep learning technology, deep neural networks have achieved excellent results in various fields. At present, deep learning technology is not widely involved in the field of information security, especially for the application of password encryption and decryption problems are few and far between, so this paper mainly tries to crack for complex encryption methods. Finally, the multi-module combined convolutional neural network model proposed and designed based on convolutional neural network is applied with AES advanced encryption standard, and the evaluation index of the experiment is set. It is proposed that by encoding the plaintexts into binary for bit-by-bit comparison, the number of bit hits of the plaintexts decrypted by the model is counted as the evaluation index of the experimental results, and the experiments are set to 128-bit, 256-bit, 1024-bit plaintexts, with the The experimental results reflect that the model based on convolutional neural network is more sensitive to the decryption results of complex passwords, and the model can reduce the distance between the original plaintext and the decrypted plaintext. This paper is an attempt to decrypt complex passwords by using deep learning techniques, and there are certain shortcomings and limitations in the experiments, both in terms of experimental data settings and model building, and there is still much to be studied in depth. The model proposed in this paper requires that the experimental data must be symmetric passwords, and there are certain required restrictions on the length of the passwords to be applicable, which relaxes many restrictions in the experimental evaluation index settings compared to simple passwords. Modern encryption algorithms are either based on a large number of discrete function transformations or on some hard-to-compute numbertheoretic problems, which make the output ciphertext indistinguishable from the real random data in a statistical sense. A good encryption algorithm, which destroys the laws and patterns between the input and output, is difficult to obtain with machine learning, and deep learning networks to simulate it, and its control parameters have to be comparable to its secret key set, so the decipherment uses moderate learning to decipher modern encryption algorithms, which has high requirements on the selection of the network structure and the setting of parameters, and the workload of this point is very large. We can try to analyze different encryption algorithms and design a targeted model structure and parameters that can continue to reduce the distance between the decryption space and the original space.

## Acknowledgments

This research work is supposed by the National Key R&D Program of China(2018YFB1201500), National Natural Science Founds of China (61602376, 61773313, 61602374,61702411), National Natural Science Founds of Shaanxi (2017JQ6020, 2016JQ6041), Key Research and Development Program of Shaanxi Province (2020GY-039, 2017ZDXMGY-098, 2019TD-014), Science Technology Project of Shaanxi Education Department(16JK1573, 16JK1552).

## References

[1] Krizhevsky A, Sutskever I, Hinton G. ImageNet Classification with Deep Convolutional Neural Networks[C]// NIPS. Curran Associates Inc. 2012.

[2] MD Zeiler, Fergus R. Visualizing and Understanding Convolutional Neural Networks[J]. Springer International Publishing, 2013.

[3] Privitera C M, Plamondon R. A self-organizing neural network for learning and generating sequences of target-directed movements in the context of a delta-lognormal synergy. IEEE, 1995.

[4] Stevenson, G A. Analysis of Pre-Trained Deep Neural Networks for Large-Vocabulary Automatic Speech Recognition. 2016.

[5] Forouzan B A. Cryptography and Network Security[M]. Tsinghua University Press, 2009.

[6] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative Adversarial Nets. MIT Press, 2014.

[7] Hitaj B, Gasti P, Ateniese G, et al. PassGAN: A Deep Learning Approach for Password Guessing[J]. arXiv, 2017.

[8] Gomez A N, Huang S, Zhang I, et al. Unsupervised Cipher Cracking Using Discrete GANs[J]. 2018.

[9] Mathur N, Bansode R. AES Based Text Encryption Using 12 Rounds with Dynamic Key Selection[J]. Procedia Computer Science, 2016, 79:1036-1043.

[10] Kundi D S, Aziz A, Ikram N. A high performance ST-Box based unified AES encryption/decryption architecture on FPGA[J]. Microprocessors & Microsystems, 2016, 41(Mar.):37-46.

[11] Indrayani R, Nugroho H A, Hidayat R, et al. Increasing the security of mp3 steganography using AES Encryption and MD5 hash function[C]// 2016 2nd International Conference on Science and Technology-Computer (ICST). IEEE, 2016.

[12] Naman, Sumiran, Sayari Bhattacharyya, and Tufan Saha. "Remote Sensing and Advanced Encryption Standard Using 256-Bit Key." Emerging Technology in Modelling and Graphics. Springer, Singapore, 2020. 181-190.

[13] Hinton G E. Learning multiple layers of representation[J]. Trends in Cognitive Sciences, 2007, 11(10):428-434.

[14] Pasquini D, Cianfriglia M, Ateniese G, et al. Reducing Bias in Modeling Real-world Password Strength via Deep Learning and Dynamic Dictionaries[J]. 2020.

[15] Das N, Park H, Wang Z J, et al. Bluff: Interactively Deciphering Adversarial Attacks on Deep Neural Networks[J]. IEEE, 2020.

[16] Biesner D, Cvejoski K, Georgiev B, et al. Generative Deep Learning Techniques for Password Generation. 2020.

[17] Hagiwara, Masafumi, Sato, et al. Analysis of Momentum Term in Back-Propagation[J]. IEICE transactions on information and systems, 1995, 78(8):1080-1086.

[18] Magoulas G D. Improving the Convergence of the Backpropagation Algorithm Using Learning Rate Adaptation Methods[J]. Neural Computation, 1999, 11(7):1769.

[19] Krizhevsky A, Sutskever I, Hinton G. ImageNet Classification with Deep Convolutional Neural Networks[C]// NIPS. Curran Associates Inc. 2012.

[20] Fan H, Wang J, Zhuang B, et al. A Hierarchical Attention Based Seq2seq Model for Chinese Lyrics Generation[J]. 2019.

[21] Wei T Z, Pham K, Dillon B, et al. Evaluating Syntactic Properties of Seq2seq Output with a Broad Coverage HPSG: A Case Study on Machine Translation. 2018.

[22] D Pasquini, Gangwal A, Ateniese G, et al. Improving Password Guessing via Representation Learning[C]// 2021 IEEE Symposium on Security and Privacy (SP). IEEE, 2021.

[23] Babu R G, Dhineshkumar K, Sharma R, et al. A Survey of Machine Learning Techniques using for Image Classification in Home Security[J]. IOP Conference Series: Materials Science and Engineering, 2021, 1055(1):012088 (9pp).

[24] Chudow J J, Jones D, Weinreich M, et al. A Head-to Head Comparison of Machine Learning Algorithms for Identification of Implanted Cardiac Devices[J]. The American Journal of Cardiology, 2020.

[25] Fan Z, Liu R. Investigation of machine learning based network traffic classification. 2017:1-6.

[26] Fast, Lean, and Accurate: Modeling Password Guessability Using Neural Networks[C]// Usenix Security Symposium. 2016.

[27] Association U. Proceedings of the 7th USENIX conference on Networked systems design and implementation. 2010.

[28] Morris R, Thompson K. Password Security: A Case Study. comm of the acm, 1979.

[29] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space[J]. Computer Science, 2013.

[30] Sutskever I, Vinyals O, Le Q V. Sequence to Sequence Learning with Neural Networks[J]. Advances in neural information processing systems, 2014.

[31] Chairatluri G, Vijay, Chairmeadows P, et al. Proceedings of the 12th ACM conference on Computer and communications security. 2005.

[32] Alani M M. Neuro-cryptanalysis of DES[C]// World Congress on Internet Security (WorldCIS-2012). IEEE, 2012.

[33] Alani M M. Neuro-Cryptanalysis of DES and Triple-DES[C]// International Conference on Neural Information Processing. Springer, Berlin, Heidelberg, 2012.

[34] Alsaffar Q S, Mohaisen H N, Almashhdini F N. An encryption based on DNA and AES algorithms for hiding a compressed text in colored Image[J]. IOP Conference Series: Materials Science and Engineering, 2021, 1058(1):012048 (12pp).

[35] Blasch E P, Gao J, Tung W W. Chaos-based image assessment for THz imagery[C]// Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on. IEEE, 2012.

[36] Zhao F. A Hyper-Chaotic Color Image Encryption Algorithm and Security Analysis[J]. Journal of Computers, 2019, 14(7):496-506.

[37] Zhang Y, Bo Z. Algorithm of image encrypting based on Logistic chaotic system. Application Research of Computers, 2015.

[38] Zhang Z J, Xiao B X, Zheng X D, et al. An Image Encryption Algorithm Based on Chaos System and DNA Strand Displacement Model[J].

[39] Arab A, Rostami M J, Ghavami B. An image encryption method based on chaos system and AES algorithm[J]. Journal of Supercomputing, 2019(3).

[40] Sbiaa F, Kotel S, Zeghid M, et al. High-Level Implementation of a Chaotic and AES Based Crypto-System[J]. Journal of Circuits, Systems and Computers, 2017, 26(7).

[41] Abbas A M. Image encryption based on Independent Component Analysis and Arnold's Cat Map[J]. Egyptian Informatics Journal, 2016, 17(1).

[42] Hao X, Zhang G, Ma S. Deep Learning[J]. International Journal of Semantic Computing, 2016, 10(03):417-439.

[43] Graves A. Long Short-Term Memory[J]. Springer Berlin Heidelberg, 2012.