Cryptography Using Artificial Neural Networks

Ololade Aworinde
Texas Tech University,
Lubbock, Texas, USA
Oaworind@ttu.edu

Abstract- Cryptography science includes many algorithms and protocols that vary in complexity and security aims. In general, there are two branches of encryption in cryptography: symmetric encryption and asymmetric encryption. In symmetric encryption, we use one shared secret key for encryption and decryption process, while for asymmetric encryption, we have two different keys, one for encryption which is a public key and one for decryption it uses a private key. However, there are many other operations used between encryption and decryption process, which is XOR operation which depends on a logic gate if we have two inputs equivalent to 0 or 1 output 0, otherwise it output 1. This logic operation is widely used in most cryptographic algorithms, so we build a model based on an artificial neural network to be trained on performing and predicting the XOR operation output.

Keywords: Cryptography, Neural Network, Symmetric Key Encryption

I. INTRODUCTION

Currently, information security is a crucial factor for every firm or organization. Information should be end-to-encrypted to assure people of the message's authenticity between sender and receiver. The sender wants assurance that his/her message is read only by the receiver and that no other intruder is spying on them during this communication. To achieve this security and assurance, cryptography comes in an important role. Here we will see the implementation of cryptography using artificial neural networks.

Artificial neural networks work on the principle of automatic decision-making. These decisions are based on the calculation of appropriate weights (parameters). It makes the system compatible. Here, one most important thing is that keys used in stream cipher cryptography should be available for these networks so that the system completely avoids vulnerability.

Our brain computes in a completely different manner from the conventional digital computer. The concept of the Artificial neural network is motivated by the human brain's functionality. There are millions of neurons in the brain with billions of interconnections. The same concept is applied to artificial neural networks. In neural networks, many parallel and distributed processors are present and build an artificial neural network with a massive number of interconnections. These artificial neural networks are nonlinear dynamic machines that can expand the input data expression into a linear form or combine inputs to synapses. After that, they compute the accurate output using nonlinear transformation. [1]

Nowadays, the security of information is one of the essential things in today's business and corporate in terms of both the authenticity of transactions and uninterrupted communications between two ends. Security from malicious attacks is also a feature that must be achieved during communication. Cryptography is a concept in which we study the security of information and the feasibility of communication over a communication channel to achieve the privacy of the transmitting information. [2] [3].

There are different cryptographic techniques, and stream cipher is the most used technique. Confidentiality, authentication, and integrity are three essential features that should be achieved when we use any cryptographic technique. Confidentiality is the most crucial feature. Third parties can see the encrypted message but cannot decipher it. Authenticity is the concept by which the receiver gets the verification of the genuine sender. Without proper authentication, there is always a risk of malicious senders, which can harm the whole network. Integrity ensures that the transmitted message is complete and reaches the receiver's end without any defect. Here the receiver can check the integration of the transmitted data. Messages or data should not be altered or modified during communication. There is constant pressure during the formation of cryptographic artificial neural networks to avoid cryptanalytic attacks and ensure the cost is low. [4] [5]

This research aims to implement an uncomplicated and inexpensive secure system in digital communication and internet applications or any system in which data is transferred between ends. The role of this system is to avoid malicious attacks and ensure the system's security.

II. ANN (ARTIFICIAL NEURAL NETWORK)

As the name indicates, it is an artificial network that mimics the ability to understand and learn, just like human neural networks. The Artificial Neural Network is an information processing and modeling system based on biological neural networks [5]. These systems are designed to achieve a specific objective. That is why they are used in applications like pattern recognition or data classification with the help of a specific learning process. As we know learning process in a human neural network depends on the synaptic connections among the neurons. The same thing is applicable here on ANNs. [7]

The generalization of mathematical models based on human neural biology is the basis for developing an ANN. The basic building blocks behind the development of ANNs includes:

1. **Input** - It is the set of data that are fed into the model for the training and learning process. For instance, an array of image-related pixel values can be used as the input data for an object detection model.
2. **Weight** - Its primary purpose is to prioritize the characteristics that contribute most to the learning. It accomplishes this by applying scalar multiplication on the weight matrix and input value. They self-adjust depending on the difference between predicted outputs vs training inputs.

3. **Transfer function** - The role of transfer function is to aggregate many inputs into a single output value so that the activation function can be applied. It is carried out by adding up all of the transfer function's inputs.
4. **Activation Function** – The activation function introduces non-linearity in the working of perceptrons to consider varying linearity with the inputs. In the absence of this, the output would simply be a linear combination of the input values and would not be able to add nonlinearity into the network.
5. **Bias** - The bias shifts the value that the activation function provides. Its function is comparable to that of a constant in a linear function.

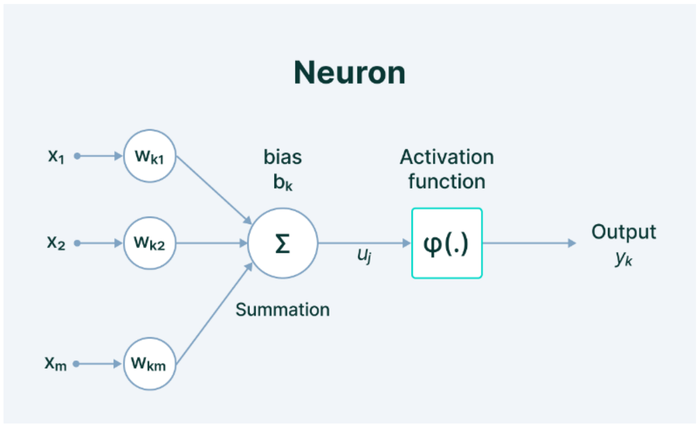


Fig 1: Basic components of an artificial neural network.

Many simple units(processors) form an Artificial Neural Network. Each processor has a small amount of local memory. All these units relate to a communication link, which is unidirectional. Every link carries some numeric data. All these units work on their local data, and the data is received as input through a communication link between the two units. There are different types of neural networks, and all of these differ in strength and functionality based on the different applications.

III. ANN ARCHITECTURE

The architecture of ANN depends on the number of neurons involved, the way of interconnection between neurons, and the processing methodology throughout the network. There are three types of ANN architecture based on the neural network's linking structure with the learning algorithms that are used to train the network [4]. These three categories are-

1. Single layer feedforward ANN,

2. Multilayer feedforward ANN,

3. Recurrent ANN.

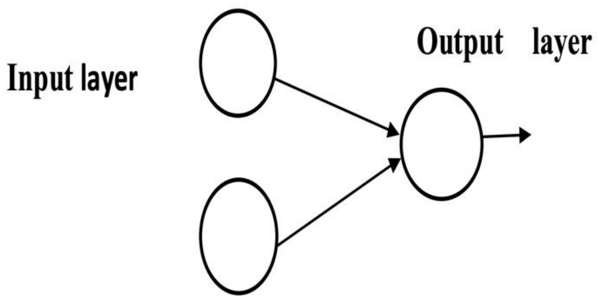


Fig 2a: Single layer feed forward network.

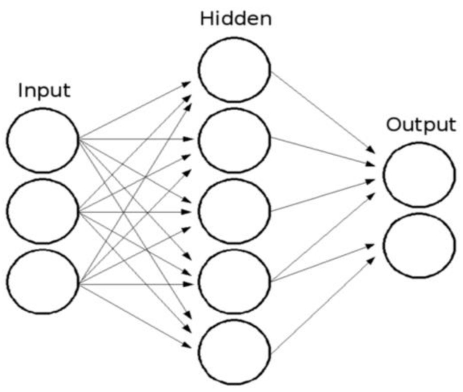


Fig 2b: Multilayer feedforward network

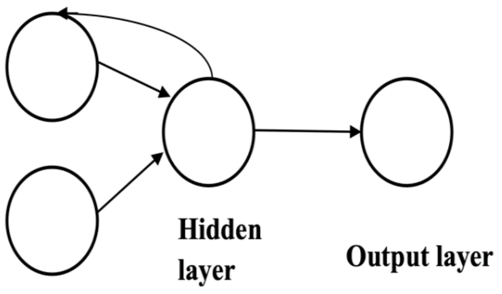


Fig 2c: Recurrent Network

Supervised learning involves an external supervisor to ensure each output unit gives the desired output according to the given input. In this learning process, global information is required. This scenario of supervised learning involves error correction learning too.

The most crucial aspect of this supervised learning is error manipulation and correction between desired and computed output. The main objective is to obtain a set of weights that minimizes the error.

IV. **Cryptography**

Cryptography is defined as the exchange of data into a mingle code that can be deciphered and sent across a public or private network. It is the practice and study of hiding information. Before the 1970s, cryptography was used only in the official work of the military and government. However, nowadays, after the immense use of computers and the internet, cryptography is an essential part of our life. We cannot imagine the security of different vulnerable applications like online banking, online shopping, ticket booking systems, Wireless LANs, WANs, and MANs. Here we get familiar with some cryptographic primitives essential to developing any secure application.

The four primary use of cryptography in any secure application are – authentication, data integrity, sender and receiver privacy, and non-repudiation. Here, privacy is a must to ensure secret communication between sender and receiver. That means nobody can inspect the contents of the transmitted message, and communication should be secret. In modern communication, encryption gives this privacy. A Password is a typical example of one-way authentication in which a user gets access after authenticating himself/herself by inputting the password. In this method, a secret key is shared between the sender and receiver. That means both sender and receiver have the same key for the encryption and decryption of the message. This method is known as symmetric key encryption or secret key encryption.

There are 3 major types of cryptography which includes

1. Secret Key Cryptography (Symmetric)
2. Public Key Cryptography (Asymmetric)
3. Hash Functions

SECRET KEY CRYPTOGRAPHY:

In secret Key Cryptography, or symmetric cryptography, a single key to encrypt and decrypt data. It is the simplest form of cryptography where cryptographic algorithm uses the secret key in a cipher to encrypt the data and decrypts the data with the use of the same secret key. Secret Key Cryptography can be used for many applications but is commonly only used on data stored on servers and databases.

PUBLIC KEY CRYPTOGRAPHY:

In public Key Cryptography, or asymmetric cryptography, two keys are used to encrypt and decrypt data. One key is used to encrypt the data, while the other key is used to decrypts the data. One key is the private key, while the other is the public key because it can be used by anyone. However, Public keys cannot be created from private keys, but the private keys can be created from public keys.

HASH FUNCTIONS:

Hash functions are irreversible functions that make it impossible to recover the original data. A given string can be changed into a string of a specific length by hashing. For every input data, a good hashing algorithm will provide a different output. The only method to decipher a hash is to try every input until you obtain the same hash.

V. PROPOSED METHOD

The goal of machine learning entails optimizing the selected loss function. The network gains knowledge by modifying the biases and weights to reduce model error with each iteration. Reverse-mode automated differentiation is particularly crucial for neural networks. To minimize the cost function, one must choose the appropriate weights and biases. Computing the gradient with respect to the parameters can help achieve that, as by definition the gradient is a vector of partial derivatives of C ($w1$, $w2$, …, wm, $b1$, $b2$…, bn). As we know, derivatives measure the change of a function's output with respect to its input. The gradient of the cost function tells us in which direction C ($w1$, $w2$, …, wm, $b1$, $b2$…, bn) decreases faster. It is frequently referred to as Gradient Descent. The network converges towards the minimum with each iteration. The gradient can be extracted from a potentially huge, complex model using automatic differentiation, which combines the chain rule with enormous computational capacity. This algorithm is better known as Backpropagation. Backpropagation is recursively completed through every single layer of the neural network.

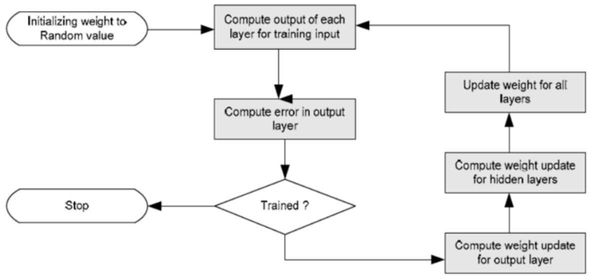

Initializing weight to Random value
Compute output of each layer for training input
Update weight for all layers
Compute error in output layer
Compute weight update for hidden layers
Stop
Trained ?
Compute weight update for output layer

Fig 3: Backpropagation training algorithm

The backpropagation algorithm is a supervised learning algorithm [10]. It is used to train a multilayer neural network that maps the relation between the target output and output of the network. During the training period, the input pattern is fed through the network with network connection weights and biases of the activation functions. The values of weights and biases are initially assigned by a small random number. In this rule, the network repeatedly provides input-output pairs and then adjusts weights after each epoch; modification of the weight minimizes the network error.

The training of the network is repeated iteratively. In each epoch, the following occurs:

FEED FORWARD PROPAGATION

When a training dataset is fed to the input layer, the weighted sum of the input to the jth node in the hidden layer is given by

$$\mathrm{Net}_j = \sum w_{i,j}\, x_j + \theta_j$$

The equation above is used to calculate the aggregate input to the neuron. The θ_j term is the weighted value from a bias node that always has an output value of 1. The bias node is considered a "pseudo input" to each neuron in the hidden layer and the output layer and is used to overcome the problems associated with situations where the values of an input pattern are zero. If any input pattern has zero values, the neural network could not be trained without a bias node.

To decide whether a neuron should fire, the "Net" term, also known as the action potential, is passed onto an appropriate activation function. The resulting value from the activation function determines the neuron's output and becomes the input value for the neurons in the next layer connected to it.

The Sigmoid equation is frequently used as the activation function in the backpropagation algorithm to introduce non-linearity.

$$O_j = x_k = \frac{1}{1 + e^{-Net_j}}$$

Similarly, they are used to determine the output value for node k in the output layer.

BACKPROPAGATION

Output Layer

Given that the actual activation value of the output node, k, is O_k, and the expected target output for node k is t_k, then the difference between the expected output and the actual output is given by:

$$\Delta_k = t_k - O_k$$

The error signal for node k in the output layer can be calculated as

$$\delta_k = \Delta_k O_k (1 - O_k)$$

where the $O_k(1-O_k)$ term is the derivative of the Sigmoid function.

According to the delta rule, the error at node k multiplied by the activation of node j determines how much the weight between input node j and output node k changes.

To modify the weights, $w_{j,k}$, between the output node, k, and the node j is given by:

$$\Delta w_{j,k} = l_r \delta_k x_k$$
$$w_{j,k} = w_{j,k} + \Delta w_{j,k}$$

where $\Delta w_{j,k}$ is the change in the weight between nodes j and k, l_r is the learning rate The relative change in weights is indicated by the learning rate, which is a usually a small constant. If the learning rate is too high, the network may fluctuate around the minimum point, surpassing the lowest point with each weight change but never truly reaching it. If the learning rate is too low, the network may learn very slowly. The learning rate is typically quite low, with ≤0.1 being a frequent value. The learning rate can decrease from a high number during the learning phase thanks to several modifications made to the Backpropagation algorithm. This offers a lot of benefits. Training will begin quickly since it is believed that the network commences at a state that is far from the ideal set of weights. The learning rate reduces as it gets closer to the ideal point in the minima as learning advances. The network is encouraged to converge to a solution while the chance of overshooting is decreased by slowing the learning process close to the optimal point. But, if the learning process begins near to the ideal point, the system may at first oscillate; however, when the learning rate slows over time, this effect is lessened. It should also be noted that, in equation above, the X_k term is the input value to the node k, and is the same value as the output from node j.

A change is made to the equation above to improve how the weights are updated:

$$\Delta w_{j,k}^{n} = l_r \delta_k x_k + \Delta w_{j,k}^{(n-1)} \mu$$

Here the weight update during the nth iteration is determined by multiplying a momentum term (μ) to the (n-1)th iteration of the $\Delta w_{j,k}$ term.

Hidden Layer

The error signal for node j in the hidden layer can be calculated as

$$\delta_k = (t_k - O_k) O_k \sum (w_{j,k} \delta_k)$$

where the summing term adds the weighted error signal for each of the k output layer nodes.

To the adjust the weight, $w_{i,j}$, between the input node, i, and the node, j in the hidden layer is given by:

$$\Delta w_{i,j}^{n} = l_r \delta_j x_j + \Delta w_{i,j}^{(n-1)} \mu$$
$$w_{i,j} = w_{i,j} + \Delta w_{i,j}$$

The error function, E, is given by:

$$E = ½ \sum (\sum (t_k - O_k)^2)$$

When the neural network has been properly trained, the error function should ideally be zero. However, this is numerically unrealistic.

VI. IMPLEMENTATION:

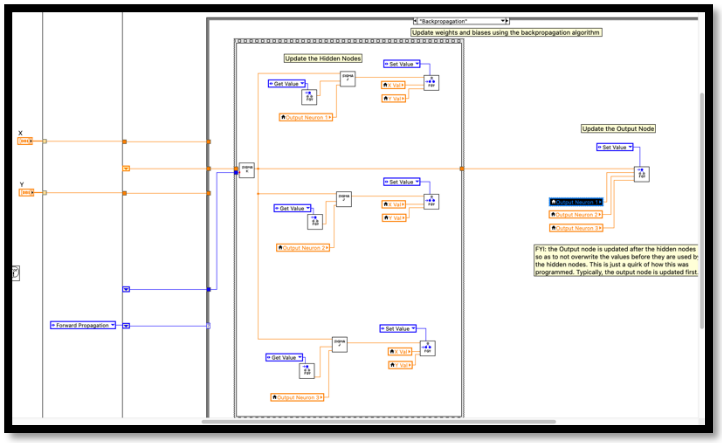
My project aims to build professional neural network deep learning as it is considered an advanced branch of machine learning closer to Artificial Intelligence. It reaches us to the modeling of complex relationships and concepts, as the current research methodology solutions worldwide follow these days as building deep learning neural networks using the best algorithms such as the emerging feed-forward neural and backpropagation algorithms in one application to build a brilliant model. This enables us to reduce error rate intelligently and apply best practices of in decent gradient principle to be a more efficient and practical artificial intelligent model that competes with counterpart models applied in recent research papers from the point of view that the efficiency of applying cryptography model prediction increased more than 99.9%

So, learning the dilemma of building professional deep learning neural network / artificial intelligence is the new trend we all compete in to let machines think and behave like human beings, and as long as we can build efficient and reliable algorithms and techniques as we can go deep into protecting our cipher networks and guarantee high-security measures.

I implemented a neural network model program using Labview as I used the backpropagation algorithm to train itself to produce an XOR output. The code uses functional global variables (FCVs), which is an efficient coding technique in view heavily to avoid the wires being scattered.

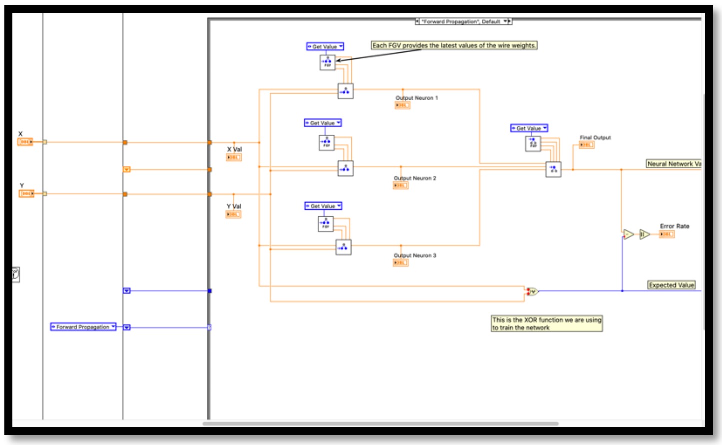
In this model, the neural network is based on a sigmoid neural network with minimal log function output, which is very close to 0. In contrast, when the output of the log function is very large, the output of the neuron reaches 1.

The neural network is built based on three hidden neurons, two input neurons, and one output neuron.



FYI: the Output node is updated after the hidden nodes so as to not overwrite the values before they are used by the hidden nodes. This is just a quirk of how this was programmed. Typically, the output node is updated first.

Fig 4: The back propagation Block Diagram



Each FGV provides the latest values of the wire weights.

This is the XOR function we are using to train the network

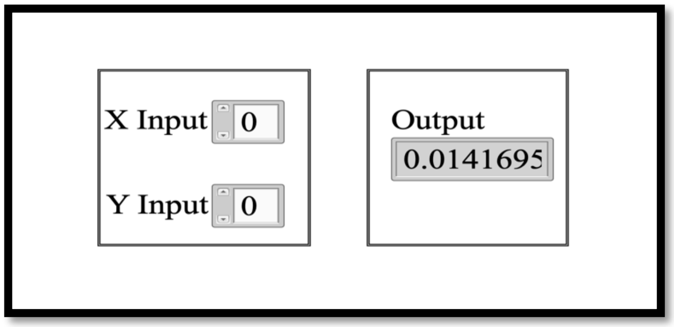
Fig 5: feed-forward algorithm block diagram

The neural network was initially trained based on the XOR operation using a feed-forward model neural network, then processed using the backpropagation algorithm to justify the results and reduce the error.

I considered the delta rule related to the learning rate, as a high learning rate will make my model reach the bottom quickly. The probability of diverse real surface increase and a low learning rate will cause the slow rate of my training model, so these aspects will take into my consideration to reach the best optimum algorithm in my model.

VII. RESULTS AND DISCUSSION

By applying the backpropagation algorithm and testing the algorithm, we get expected good results when X input and Y input = 0 the output approaches 0 as shown in fig. When input X & Y = 1, the output also approaches 0. When testing with X & Y= 0, 1 respectively or vice versa, the output approaches 1 as shown below.

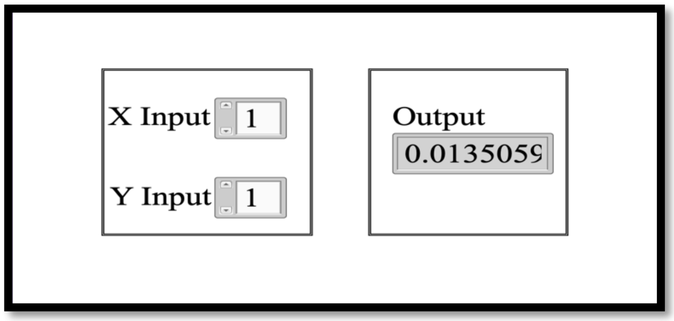


X Input 0

Y Input 0

Output 0.0141695

Fig 6a:

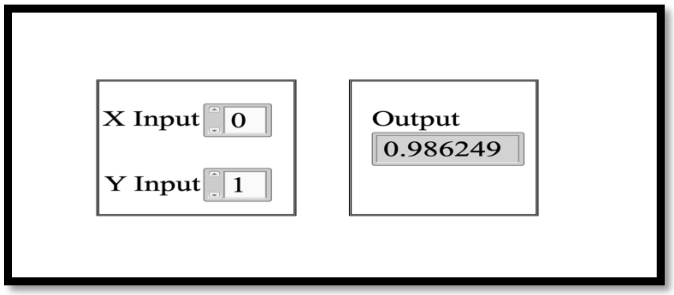


X Input 1

Y Input 1

Output 0.0135059

Fig 6b:

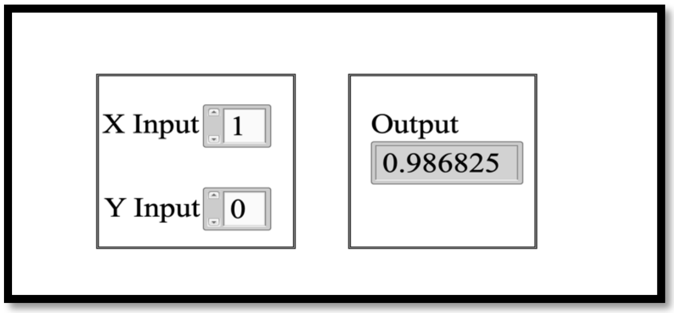


X Input 0

Y Input 1

Output 0.986249

Fig 6c:



X Input 1

Y Input 0

Output 0.986825

Fig 6d: Test Result Block Diagram when different inputs

ERROR RATE:

As in the proposed neural network I performed 80,000 iterations of training based on random initial weights generated randomly from LabVIEW function random number which generates random fraction value between 0 and 1, so to train the model based on that as output of the neural network will be either higher probability which can reach 0.999 for right prediction of XOR operation when inputs are different and reach to 0.0111 when inputs are equal. In neural networks, their error surfaces are guaranteed to have a large and, in some cases, an infinite number of local minima. So local minima are only problematic when they are spurious which means it corresponds to a configuration of weights in a neural network that incurs a higher error than the configuration at the global minimum.

The error rate after each iteration is computed based on the difference between the output of neuron and actual output of a XOR operation. At the end of the iterations, we can notice in the graphs in figure 7a and 7b below that error is close to zero.

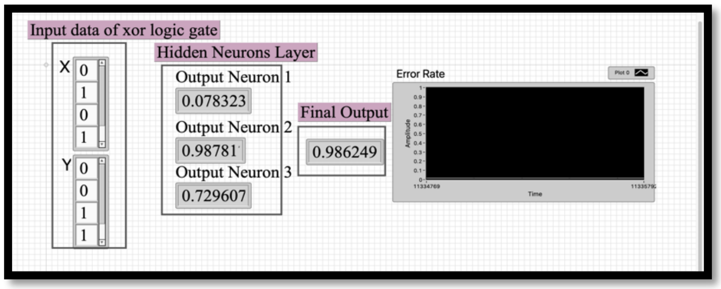

Input data of xor logic gate
X: 0, 1, 0, 1
Y: 0, 0, 1, 1
Hidden Neurons Layer
Output Neuron 1: 0.078323
Output Neuron 2: 0.98781
Output Neuron 3: 0.729607
Final Output 0.986249

Fig 7a:

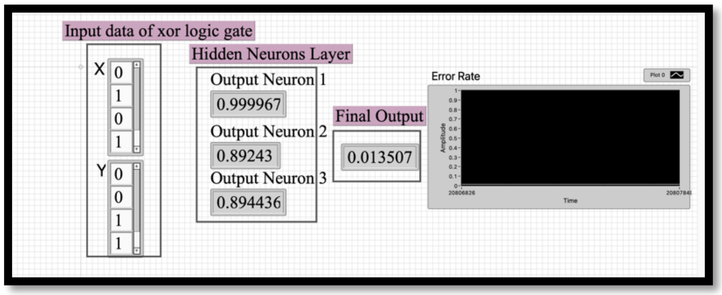

Input data of xor logic gate
X: 0, 1, 0, 1
Y: 0, 0, 1, 1
Hidden Neurons Layer
Output Neuron 1: 0.999967
Output Neuron 2: 0.89243
Output Neuron 3: 0.894436
Final Output 0.013507

Fig 7b

After 80,000 iterations, we can see from the table 1 below that the model is trained such that the errors are minimal or negligible.

X input	Y input	Neuron Output	Target Output	Error
0	0	0.0141695	0	0.0141695
0	1	0.986249	1	0.013751
1	0	0.986825	1	0.013175
1	1	0.0135059	0	0.0135059

Table 1: Error rate of network output.

XOR CRYPTOGRAPHY

With the application of the trained model, we achieved a simple XOR cryptographic operation as shown in the table 2 below. This table shows the encryption and decryption of data using a single key where the inputs are sets of binary combinations.

Plaintext	Key	Ciphertext	Key	Plaintext
0	0	0	0	0
1	0	1	0	1
0	1	1	1	0
1	1	0	1	1
0	1	1	1	0
0	1	1	1	0
1	0	1	0	1
1	0	1	0	1

Table 2: XOR Cryptographic Operation

As the XOR operation is used in many encryption algorithms like a one-time pad, block cipher, and stream cipher, the XOR function is essential in modern cryptography operations, so we applied a neural network to train our model to apply this operation. So now we can easily use our neural network model to be applied in the middle of complicated cryptography schemes used at the software or hardware level.

VIII. CONCLUSION

With the help of Artificial Neural Networks, we can emulate highly complex computational machines. It is a simple and very powerful technique used in this project.

ANNs are not only used in complex combinational circuits but in sequential circuits also.

The most crucial thing is data security in data communication systems. Data security is made sure with the use of ANNs along with cryptography. ANNs are applied with cryptography using two methods. First is the sequential machine-based method by

which data is encrypted. Another is the neural network by which digital signal cryptography is analyzed. Training algorithms should be incorporated to get better results. Therefore, using Artificial Neural Networks is a new and more secure technology for the encryption and decryption of traveling data between two nodes.

REFERENCES

[1] S. Haykin. *Neural Networks: A Comprehensive Foundation*. 2nd edition, Prentice Hall, 1998.

[2] R. Stinson. *Cryptography, Theory, and Practice*, 2nd edition, CRC Press, 2002.

[3] A.J. Menezes, P.C. Van Oorschot, and S.A. Vanstone. Handbook of Applied Cryptography, CRC Press, 1997.

[4] W. Stallings. *Cryptography and Network Security: Principles and Practices*. Prentice Hall, 2003.

[5] B. Schneier. *Applied Cryptography*. John Wiley & Sons Inc., New York, 1996.

[6] A. Sadeghian, M. Zamani, and V. Akbarzadeh, “Applications of Artificial Neural Networks in Cryptography—A Survey,” in progress.

[7] R.L. Rivest, “Cryptography and Machine Learning,” Advances in Cryptology, Lecture Notes in Computer Science, vol. 739, pp. 427–439, 1991.

[8] D. Pointcheval, "Neural Networks and their Cryptographic Applications," in *Proc. Of Eurocode*, pp. 183–193, 1994.

[9] W. Kinzel, and I. Kanter, “Neural Cryptography,” *Proc. of the 9th Int’l Conf. on Neural Information Processing (ICONIP’02)*, vol. 3, pp. 1351–1354, 2002.

[10] A. Ruttor, W. Kinzel, L. Shacham and I. Kanter, “Neural cryptography with feedback,” *Phys. Rev. E.*, (69): 046110—1-7, 2004.

[11] R. Mislovaty, E. Klein, I. Kanter, and W. Kinzel, “Public Channel Cryptography by Synchronization of Neural Networks and Chaotic Maps,” *Phys. Rev. Lett., (91), 118701-1-4*, 2003.

[12] L.N. Shechem, E. Klein, R. Mislovaty, I. Kanter, and W. Kinzel, “Cooperating attackers in neural cryptography,” *Phys. Rev. E.*, (69), 066137-1-4, 2004.

[13] R. Mislovaty, Y. Perchenok, I. Kanter, and W. Kinzel, “Secure key-exchange protocol with an absence of injective functions,” *Phys. Rev. E.*, (66): 066102-1-5, 2002.

[14] W. Kinzel and I. Kanter, “Interacting Neural Networks and Cryptography,” *Advances in Solid State Physics*. B. Kramer (ed.), vol. 42, pp. 383–391, Berlin: Springer, 2002.

[15] M. Rosen-Zvi, I. Kanter and W. Kinzel, “Cryptography based on neural networks—analytical results,” *J. Phys. A: Math. Gen.*, vol. 35, no. 47, pp. 707–713, 2002.