The Scientific Journal of King Faisal University

Basic and Applied Sciences

Asymmetric Encryption Method Proposed for Arabic Letters Using Artificial Neural Networks

Mohammad Taha Kafarnawi

Department of Control and Automation Engineering ,College of Electrical and Electronic Engineering , Aleppo University, Aleppo, Syria

LINK	RECEIVED	ACCEPTED	PUBLISHED ONLINE	ASSIGNED TO AN ISSUE
https://doi.org/10.37575/b/eng/210044	16/08/2021	26/11/2021	26/11/2021	01/12/2021
NO. OF WORDS	**NO. OF PAGES**	**YEAR**	**VOLUME**	**ISSUE**
6048	7	2021	22	2

ABSTRACT

Asymmetric encryption algorithms suffer the problem of high execution time. This paper presents a proposed methodology to perform encryption and decryption of messages written in Arabic, using artificial intelligence represented by artificial neural networks based on the RSA algorithm. The method is based on dividing the message into partial messages. The arrays of weights and biases in the neural network layers are considered the encryption or decryption key, according to their function. All steps of the proposed method and how it works were shown and used in the MATLAB environment to design a system for encrypting and decrypting messages written in Arabic. The results proved the effectiveness of the proposed methodology and its superiority over the RSA algorithm in terms of execution time for both encryption and decryption. The encryption time in the proposed algorithm is close to the time of decryption, unlike what it was in the RSA algorithm, where the encryption time was relatively much greater than the decryption time. The proposed method was tested on four text files of 50KB, 100KB, 150KB, and 200KB, over two hundred iterations. The average improvement in execution time was 1,330ms and 4,497.5ms for encoding and decoding, respectively.

KEYWORDS

Data protection, decryption, execution time, security, RSA algorithm, performance

CITATION

Kafarnawi, M.T. (2021). Asymmetric encryption method proposed for Arabic letters using artificial neural networks. *The Scientific Journal of King Faisal University: Basic and Applied Sciences*, **22**(2), 106–12. DOI: 10.37575/b/eng/210044

1. Introduction

Cryptography is one of the most important techniques used to protect data against illegal access and tampering. Many algorithms are used to perform encryption. Encryption algorithms are classified into two types: symmetric encryption algorithms, such as the AES encryption algorithm, and asymmetric encryption algorithms, such as the RSA algorithm. RSA is considered one of the most important asymmetric encryption algorithms, developed by the three scientists Ron Rivest, Adi Shamir and Leonard Adleman, and is named by their initials. The RSA algorithm consists of three main stages: the stage of key generation, the stage of encryption, and the stage of decryption. The public key (PU) is generated to be used in encrypting the message before sending it and the private key (PR) is used to decrypt the message at the receiver, according to the next steps (Intila *et al.*, 2019; Moghaddam *et al.*, 2013; Amalarethinam and Leena, 2016). The Advanced Encryption Standard (AES), also known by its original name Rijndael, is a specification for the encryption of digital data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. The AES is a subset of the Rijndael block cipher developed by two Belgian cryptographers, Vincent Rijmen and Joan Daemen, who submitted a proposal to the NIST during the AES selection process. In the United States, the AES was announced by the NIST on November 26, 2001. The AES uses the same key in both encryption and decryption (Lu *et al.*, 2021). The basic difference between the two types is the key used in encryption and decryption. In symmetric encryption, the same key is used for both encryption and decryption, while asymmetric encryption algorithms use two different keys. One of them is a public key for data encryption and the other is a private key for decryption. Private and public keys are different from each other, but they are related to each other. The public key is available to all users, while the private key is hidden, secure, and only owned by its owner. One of the disadvantages of asymmetric encryption is that it takes much longer than symmetric encryption (Taleb and Obaid, 2020). Each type of encryption has advantages and disadvantages. Symmetric encryption algorithms are much faster because they require low computational cost and they are simpler, but their main weakness is that hackers may discover the key because of only using one key for both data encryption and decryption, so the key must remain secure (Taleb and Obaid, 2020). Asymmetric encryption solves the problem of publishing key by using public keys for encryption and private keys for decryption, but this type of encryption is very slow compared to asymmetric encryption because it requires much more computational resources due to the length of the keys. The bottom line is that symmetric encryption is faster than asymmetric encryption, but it is less secure. Asymmetric encryption is more confidential and provides a higher degree of data protection. Moreover, asymmetric encryption provides the features of verification, authentication, and non-repudiation. However, it suffers from relative slowness compared to symmetric encryption. A Slovak study in late 2019 presented an attempt to employ convolutional neural networks in symmetric encryption to perform the standard AES encryption algorithm. The study concluded with one recommendation and the result, which is the possibility of using convolutional neural networks in the field of encryption in addition to the other fields, but only to encrypt files of small sizes (Forgá and Okay, 2019).

In 2020, a group of researchers in Singapore used a Graphics Processing Unit (GPU) to accelerate the work of convolutional neural networks in order to employ them in the symmetric encryption process. The model provides cloud computing to protect user data. Deep Learning as a Service (DLaaS) technology is employed. Two image databases were able to encrypt with different times and with two different performances (Al Badawi *et al.*, 2020). The researchers proposed an encryption model based on integrating a chaotic diagram with multi-layered perceptron neural networks to encode grayscale images by segmentation of message to encrypt and then collect the sub-messages (Kengnou *et al.*,

Corresponding Author: Mohammad Taha Kafarnawi 00963947478706, mtktaha@gmail.com

2014). Some studies use neural networks in the process of some functions of the RSA algorithm as functions of key generation (Haghipour and Sokouti, 2009). The assistant professors, Navita and Barchi, presented a research paper for a theoretical study of the possibility of using neural networks in the field of encryption using neural networks trained by the back propagation algorithm. They did not apply it but indicated the possibility of applying it in the Matlab environment (Agarwal *et al.*, 2013). An encoding and decryption model based on the chaotic scheme and the artificial neural network with an encryption and decryption algorithm to raise the level of confidentiality was also presented, but this study considered neural networks as a supportive part rather than an essential part in the encryption and decryption process (Al-Abaidy, 2020). Another study tried to employ neural networks to perform the encryption task, but it was weak, and the use of neural networks was limited to the generation of encryption keys and the field of generating the keys; it did not present a clear method of using neural networks to perform encryption and decryption in their total form (Al Azawee, *et al.*, 2015). Chaotic technology was also used to improve neural networks to perform encryption and decryption and proposed what is called chaotic neural networks (CNN), but it was of great complexity and included three stages: the stage of generation of the chaotic key and education, the stage of generation of the chaotic chain, and the encryption process. However, this method needs a new training process every time to generate the weights required for the encryption stage (Bevi *et al.*, 2018). Therefore, this research aims to propose a method to perform asymmetric encryption using artificial neural networks for the entire process in order to speed up the encryption process, to improve asymmetric encryption, and to encrypt data in less time and at higher speed. The importance of research in the field of protection against tampering or viewing of data applies to all the stages: transferring through computer networks and storing, securing, protecting to a high degree, achieving the characteristics of authentication and verification, and to get the feature of non-deny in a relatively short time. The artificial neural networks are employed in the field of asymmetric encryption and data protection. The method proposed by this research to employ the artificial neural network to achieve asymmetric encryption of Arabic characters is meant to protect data written in the Arabic language and can be used in the field of e-government to protect data from tampering, forgery, or access in a short time, encoding a larger amount of data in less time. This research is characterized by proposing a method to make the Artificial Neural Network (ANN) completely perform the encryption and decryption functions.

2. Materials and Working Methods

This research was achieved in four steps:

- Suggesting a general method to perform asymmetric coding using Artificial Neural Networks (ANN) and defining the stages of design.
- Explanation of the mechanism of performing asymmetric encryption of Arabic letters using the proposed neural networks and determining the public key that is called the encryption key.
- Explanation of the mechanism of performing asymmetric decryption of Arabic letters using the proposed neural networks and determining the private key that is called the decryption key.
- Practical application using the Matlab environment, getting conclusions, comparing with the RSA algorithm, and analyzing the results.

2.1. The Proposed Method to Achieve Asymmetric Encoding of Arabic Characters:

Figure 1 shows the scheme of the proposed method for performing asymmetric encryption and decoding Arabic letters, which works as follows:

- **The Coding Section**: As shown in Figure 1-a, it consists of several stages:
  - **The Input**: This is the text of the message written in Arabic M. The computer representation of Arabic characters according to the Unicode code starts with the number 1,536 and ends with the number 1,791, and the field of representation of the Arabic characters is [06FF-0600] in the hexadecimal counter-system (2021, Unicode organization).
  - **The Segmentation Stage**: The message M is distributed to partial messages mi. Each message consists of two Arabic letters, which means the length of the partial message is 32 bits because the Unicode code represents each letter in two bytes.
  - **Encryption Stage**: The partial messages are distributed by artificial neural networks designed to perform encryption (the design of ANN will be explained in section 2.2), where the input of each network is a partial message, inputs = mi. The public encryption key Puk is the weights and biases of each layer in a neural network and is presented by the following matrix:

    $$Puk = [W_h\ b_h\ W_o\ b_o] \quad (1)$$

    Where:

    W_h: weights matrix in Hidden Layer

    b_h: biases matrix in Hidden Layer

    W_o: weights matrix in Output Layer

    b_o: biases matrix in Output Layer

    Determining the encryption key is done by specifying W_h, b_h, W_o, and b_o, which will be clarified in section 2.2. The output of this stage is partial codes, outputs = ci, each code corresponding to the partial message of the input of the neural network.
  - **Collection Stage**: At this stage, the sub-ciphers are compiled to get the output of the coding stage, which represents the total code of the message C.

- **The decoding section**: As shown in Figure 1-b, it consists of several stages:
  - **Input**: It is code message *C*, resulting from the encryption stage.
  - The segmentation stage: the code message C is distributed to the sub-codes *Ci*; each code consists of two characters, meaning the length of the partial code is 32 bits because the Unicode code represents each character with two bytes.
  - **Decryption Stage**: The partial codes are distributed by artificial neural networks designed for decryption. (The design of ANN will be explained in section 3.2). The input of each network is a partial code, outputs = c_i, and the private decryption key $\Pr k$ is the weights and offsets of each layer in the neural network and is presented by the following matrix:

    $$\Pr k = [W_{hd}\ b_{hd}\ W_{od}\ b_{od}] \quad (2)$$

    Where:

    W_{hd}: weights array in hidden layer in ANN for decryption.

    b_{hd}: biases array in hidden layer in ANN for decryption.

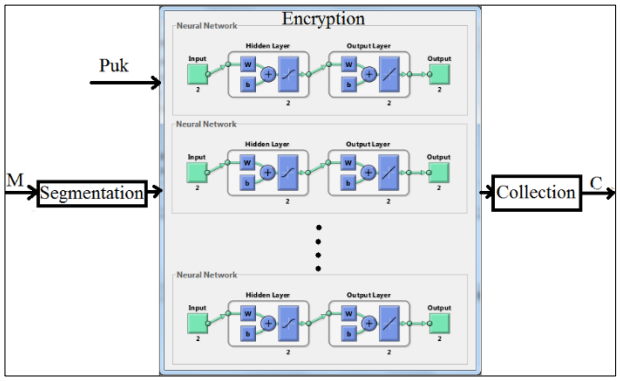
    W_{od}: weights array in output layer in ANN for decryption.

    b_{od}: biases array in output layer in ANN for decryption.

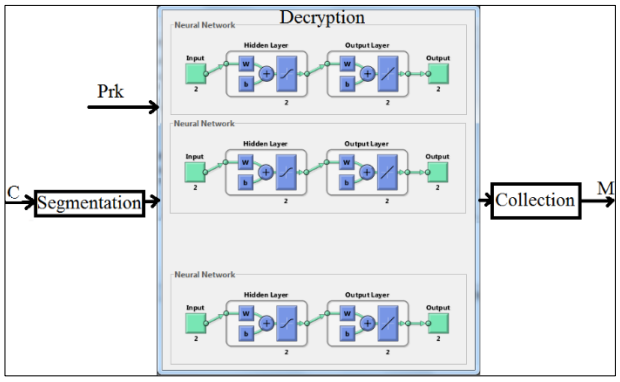
    Determining the decryption key is achieved by specifying W_{hd}, b_{hd}, W_{od}, and b_{od}, which will be clarified in section 3.2. The output of this stage is partial messages. Outputs = m_i with each code corresponding to the partial message of the output of the neural network.
  - **Collection Stage**: At this stage, the partial messages mi are collected to get the output of the decoding stage, which represents the total message M.

Figure 1: The proposed method for performing asymmetric encoding of Arabic characters

A – Encryption section



B - Decryption section



2.2. Design of Neural Networks to Perform Encryption and Generate the Public Encryption Key:

The process of designing the ANN to perform encryption was based on the RSA algorithm, and the following is a summary of RSA steps (Shambhavi and Sharma, 2018):

- Randomly generate two large prime numbers, p and q.
- Calculate the shared part between the public and private key, according to the equation:

$$n = p.q \quad (3)$$

- Calculate the factor $\varnothing$ according to the equation:

$$\varnothing(n) = (p-1)(q-1) \quad (4)$$

- Choose an integer at random, e, through the equation:

$$1 < e < \varnothing(n) \quad (5)$$

Where gcd is the greatest common divisor:

$$\gcd(e, \varnothing(n)) = 1 \quad (6)$$

- Using the extended Euclid algorithm, calculate the integer number d:

$$d = e^{-1} \pmod{\varnothing(n)} \quad (7)$$

Where mod is the remainder operation.

- The public key $Pu = (e, n)$ is used to encrypt the text via the equation:

$$C = M^e \pmod{n} \quad (8)$$

- The private key $PR = (d, n)$ is used for decryption using the equation:

$$M = C^d \pmod{n} \quad (9)$$

Where e is the Encryption Exponent, d is the Decryption Exponent, while n is called the modulus.

The process of encryption and the generation of the encryption key using ANN was carried out in three stages:

- Forming a database for a single ANN: the number of Arabic characters represented by the computer according to the Unicode code is 16 x 16 = 256, so the number of possibilities needed to form the database is 256^2 = 6,5536, because one partial message (m_i) consists of two characters. All the possibilities of the partial message (m_i) are taken and encoded by using the RSA algorithm using equation (8) after generating the cipher key using equations (3) to (7). All the codes (c_i) corresponding to all the possibilities of the partial message (m_i) are found. Pairs of the database from the input of a neural network and its corresponding desired output $\{m_{i,\ ci}\}$ are formed. The formed database is divided into three groups: the training set, the test set, and the validation set, where each set has 65,536 desired input and output pairs. This is to ensure that the network is trained, tested, and verified on all possibilities of receiving the partial message.
- Training the ANN and finding the encryption key *Puk*: a two-layer ANN is proposed; a hidden layer and an output layer, each layer containing two neurons because the input and output are composed of two Arabic letters. The network was trained according to the Levenberg-Marquardt algorithm with the following relationship (Du and Stephanus, 2018):

$$\omega_{K+1} = \omega_K - \left[J^T J + \mu I\right]^{-1} J^T \mathrm{e} \quad (10)$$

Where ω_{K+1} : the new weight (adjusted) and ω_K : the old weight (before modification).

J : Jacobian matrix, which is the value of the first derivative of the network errors related to weights and biases in backpropagation.

μ : Learning rate speeds up or slows down the network training process.

I : Identification matrix.

e : The error or difference between the current network output and the desired output.

Figure 2 shows the training, validation, and test curves of the two-part cryptographic ANN. The performance curve shows the extent of training and learning of the network, where it is found that the value of the mean squared error $MSE = 3.69e-5$. The training, validation, and test curves are identical, as shown in Figure 2, and match the fit and performance curves for all training, validation, and testing data. The reason for this is that the training, validation, and test sets match. The training is repeated until MSE equals zero or is stopped at maximum epochs equal to 1,000. The fit curves gave a fit ratio of R = 100% = 1 for all training, validation, testing, and all data. As a result, the weights and offsets (W_h) , (b_h), (W_o), and (b_o) are found, which constitute the general encryption key given by the relation (1), which is required to perform the encryption process using the ANN according to the following equation:

$$c_i = f_1\,(W_o\, f_2\,(W_h\, m_i + b_h) + b_o) \quad (11)$$

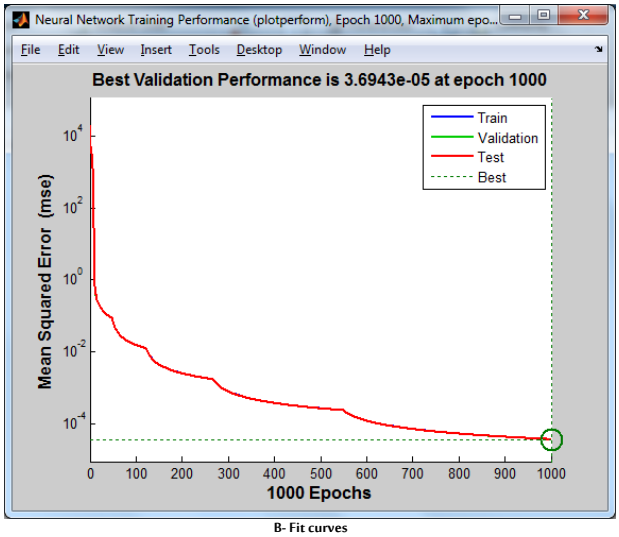
Where f_1 is the function of the activation of the output layer neurons and is a linear function given by the relation:

$$y = x \quad (12)$$

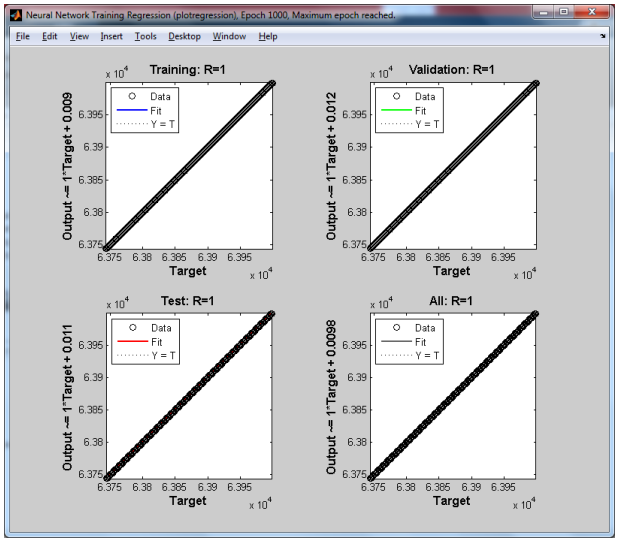
and where f_2 is the function of activation of the hidden layer neurons and is the tansig function that is given by the relation:

$$y = \tan sig(x) = \frac{2}{1+e^{-2x}} - 1 \quad (13)$$

Figure 2: Curves for an artificial coding neural network
A - Performance curves



B- Fit curves



- Determining the number of ANN needed to encode the total Arabic message M: This is done by determining the degree of division for the length of M, estimated by the number of bits, and the length of the partial message mi, estimated in bits, which is 32 bits. The partial message is made up of two Arabic letters, each character with a length of 16 bits according to the computer representation in Unicode. If the degree of division is 128 bits, the number of neural networks is 128 / 32 = 4. In addition, if the degree of division is higher, for example 256, the number of neural networks is 8. This step is important for making the encryption process faster, especially when using multi-core computers with a parallel structure. Therefore, it is more efficient than the process of designing a single neural network to encrypt the entire degree of division in terms of speed, in addition to the almost impossibility of building an ANN with relatively high degrees of division. If we want to design a single ANN to encrypt the message with a degree of division of 128 bits at once, the number of Arabic characters will be 8 in one partial message. Thus, the number of possibilities is e + 19256^8=1.8447. This needs supercomputers in terms of memory size and execution speed to conduct the training process and determine the encryption key.

2.3. Design of Neural Networks to Perform Decryption and Generate the Decryption Private Key:

The process of decrypting and generating the decryption key Prk is carried out using ANN, according to three stages:

- **Configuring a Database for Decryption Artificial Neural Network**: The same database formed to train the neural network is used to perform encoding but with opposite pairs, so the input is (c_i) and the desired output is (m_i), that is, the database is pairs to train the neural network for decoding {c_i m_i}. This database is divided into three groups: the training set, the test set, and the validation set, where each set has 65,536 desired input-output pairs. This is to ensure that the decryption network is trained, tested, and verified on all the possibilities of the partial code.
- **Training the Artificial Neural Network to Decode and Generate the PRK Decryption Key**: A two-layer ANN is proposed, having a hidden layer and an output layer, each layer containing two neurons because the input and output are composed of two characters. The network has been trained according to the Levenberg-Marquardt algorithm that is presented by the relation (10). Figure 3 shows the training, validation, and testing curves of the decoding ANN, consisting of two parts. The performance curve, which shows how well the network is trained and learned, is where we find that the value of the mean squared error MSE = 3.84e-5, as shown in the figure. The fit and performance curves are the same for all training, validation, and test data due to the matching sets for the training, validation, and test data. The fit curves gave a fit ratio R = 100% = 1 for training data, validation data, testing data, and total data. As a result, the weights and biases of the decoding network W_{hd}, b_{hd}, W_{od}, and b_{od} are found, and they constitute the decoding key given by the relation (2), which is required to perform the decoding process using the ANN, according to the following equation:

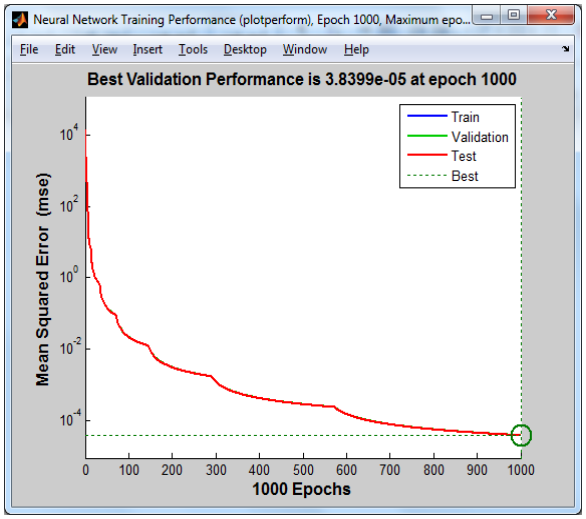
$$m_i = f_1\left(W_{od}\, f_2\left(W_{hd}\, c_i + b_{hd}\right) + b_{od}\right) \qquad (14)$$

Where:

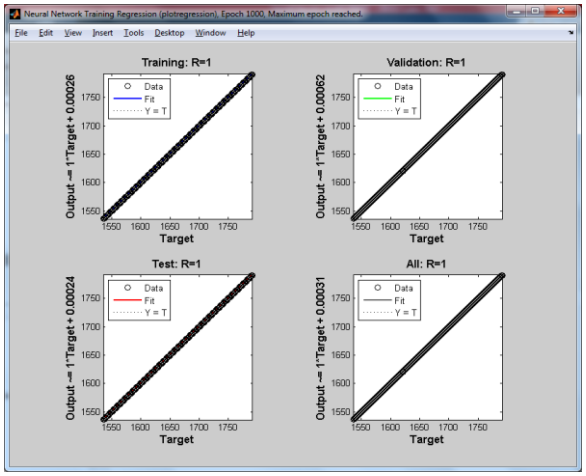
f_1 is the function of the activation of the output layer neurons, and it is a linear function given by the relationship (12)

f_2 is the function of activation of the hidden layer neurons, and it is the tansig function that is given by the relation (13)

Figure 3: Curves of an artificial neural network for decoding
A - Performance curves



B - Fit curves



- Determining the number of ANN needed to decrypt the total cipher

C: This is done in the same way that was used in the encryption process, and the number of neural networks needed for decoding is identical to the number of neural networks needed to perform the encryption.

3. Results and Discussion

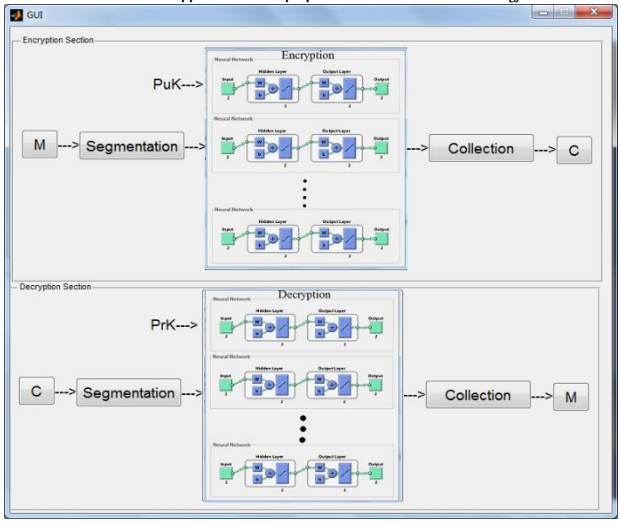
The proposed methodology for encoding and decoding using ANN has been applied and programmed in the MatLab environment using the Graphical User Interface (GUI) technology as shown in Figure 4A, where the Arabic text message is entered by downloading a text file with an extension (.txt) through the software button "M" in the encryption section, then performing the encryption using neural networks and the general encryption key by programming the equation (11) in the steps shown in section 2.1 after obtaining the encryption key as explained in section 2.2.

Figure 4B shows the encoding output of the message written in Arabic that appears by pressing the "C" software button. It is clear from the figure that the encoding output is Chinese characters, which are represented by the computer according to the Unicode code within a specified minimum limit estimated in hexadecimal and decimal: (F900)16 = (63,744)10, and the maximum limit estimated in hexadecimal and decimal: (F9FF)16 = (63,999)10. This explains the desired output range in the fitting curves for the coding neural network shown in Figure 2B. It should be noted that the two programmed functions shown in Table (1) were used to obtain the digital encoding of the computer representation of Arabic characters according to the Unicode code when starting the encryption procedure and upon completion of the decryption process to move between characters and numeric representation according to the Unicode code. As for the decryption process, it was done by the programming equation (14) in the steps shown in section 2.1 of the decryption section. This is after obtaining the decryption key as shown in section 2.3. Figure 3C shows the results of the decryption, which is the entered Arabic message. This output appears by pressing the "M" button in the decryption section shown in Figure 4A, where the range of Arabic letters is limited by the two limits: minimum (0600)16 = (1,536)10 and maximum (ff06)16 = (1,791)10. This is confirmed by the fit curves shown in Figure 3B for training the artificial neural decoding network.

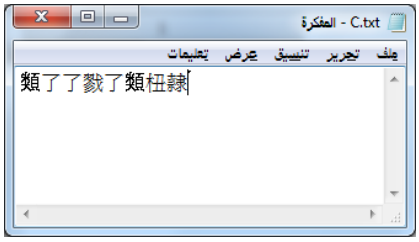
Table (1): Programming functions used to switch between Arabic characters and Unicode

programming form	programming function
M_values = unicode2native(M, ISO-8859-1')	Converts the letter M to Unicode
C = native2unicode(C_values)	Converts the Unicode of the message encoding output C_values to C

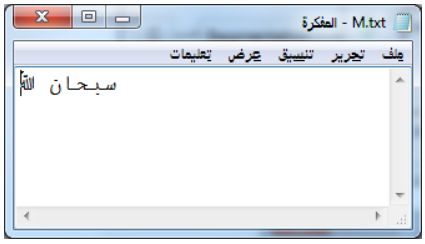
Figure 4: The proposed method for performing asymmetric encoding of Arabic characters using artificial intelligence.
A - Practical application of the proposed method with GUI technology



B - Encryption result C



C - Decryption result M

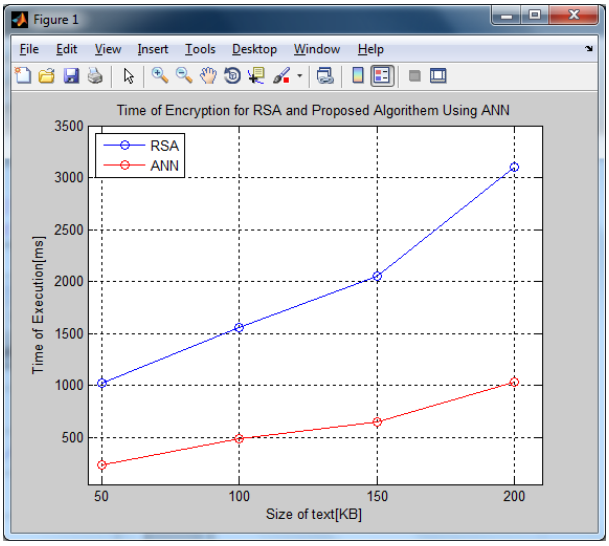


The results were obtained by testing the proposed encryption mechanism by encrypting and decrypting four sizes of Arabic messages, calculating the average execution time for two hundred iterations, and comparing those times with the execution times of the RSA algorithm to encrypt and decrypt the same message sizes with the same average number of iterations. The message sizes encrypted were 50KB, 100KB, 150KB, and 200KB. Figure 5 shows the encryption and decryption times that were obtained, where the proposed methodology for performing encryption and decryption using ANN proved superior to the RSA algorithm in terms of execution time for both encryption and decryption. This can be explained by the fact that the arithmetic operations needed by the proposed methodology to perform encryption are less complex than those required for the RSA algorithm, since the RSA algorithm contains an exponentiation with a relatively large number. Thus, the number of multiplication operations is many. In addition, the remainder of the division requires repeated subtraction operations, the number of which increases with the increase in the value of the number resulting from the exponent operation, and this is not needed for the proposed methodology to perform the encryption. The proposed method requires only a specific number of multiplications and additions because the structure of the neural network is simple and consists of only two neurons, distributed in two layers. This is evident in the decoding time of both the proposed methodology and the RSA algorithm (Figure 5B), because the resulting number of encryptions is greater than the number to be encrypted. As is clear from the computer representation of the Arabic characters to be encrypted and the computer representation of the Chinese characters resulting from the encryption process that is required to be decrypted to return to the Arabic characters, the private key in the RSA algorithm is greater than the public key. This leads to an increase in the RSA decryption time, making it longer than the RSA encryption time. This is due to the increase in the repeated subtraction operations with an increase in the value of the decryption key and the value of the number when decrypting compared to when encrypting. In the proposed methodology, the number of computer operations does not change, which makes the decoding time not very different from the encoding time. Table (2) shows the improvement resulting from the proposed methodology, which is the difference in time required to perform both encryption and decryption for both the RSA algorithm and the proposed methodology when encrypting four files of different sizes (50KB, 100KB, 150KB, and 200KB). The average improvement in time for the four files is 1,330ms and 4,497.5ms for encryption and decryption, respectively. A comparison was made between the proposed methodology and two reference studies that employ neural networks in the field of encryption using the RSA algorithm. The comparison was made in terms of execution time by calculating the average encryption and decoding time for one byte

by depending on the execution time of the largest file to discover the feasibility of the proposed methodology. The comparison showed the effectiveness of the proposed method and its superiority over the two proposed methods in the reference studies (Yousif *et al.*, 2017; Chakraborty *et al.*, 2018), as shown in Table (3). The reason for the superiority of the proposed methodology is due to two main factors. The first is the presence of a partial degree of 32 bits, which leads to a simple ANN structure consisting of only two neurons per layer. The second factor is that the proposed mechanism relies completely on the ANN for the encryption and decryption processes and there are no auxiliary algorithms, as their presence leads to additional execution times.

Figure 5: Comparison of the execution time of the proposed methodology for performing asymmetric encryption of Arabic characters using artificial intelligence with the RSA algorithm.

A - Encryption time



B - Decryption time

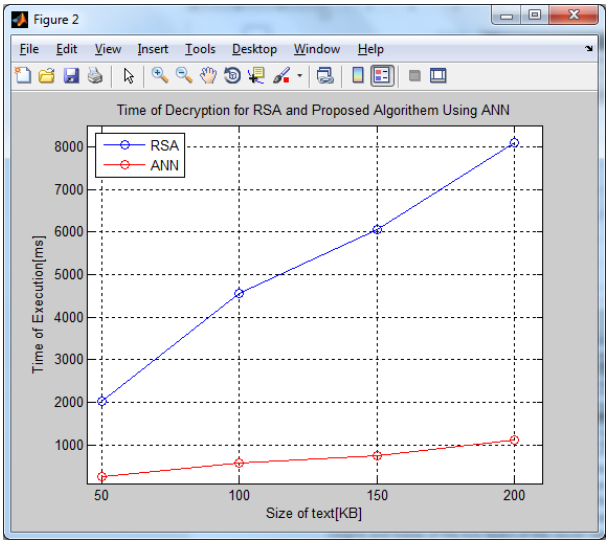


Table (2): The amount of improvement in execution time of the proposed methodology compared to the RSA algorithm

File size	50KB	100KB	150KB	200KB
Encryption time [ms]	780	1,070	1,400	2,070
Decryption time [ms]	1,760	3,970	5,290	6,970

Table (3): Comparison between the algorithm proposed in this research and two algorithms proposed in two reference studies

The method	Average encryption time of one byte	Average decryption time of one byte
Proposed method	5ns	5.37ns
(Yousif *et al.*, 2017)	84ns	270ns
(Chakraborty *et al.*, 2018)	253ns	253ns

The protection of data by encrypting according to the proposed mechanism based on ANN gives complete effectiveness and the ability to encrypt and decrypt, so neural networks can be used to perform asymmetric encryption and decryption to protect data from tampering, illegal access and securely allows for digital signatures, verification, authentication, and non-denial. The method used is highly effective against hacking because it depends on a key consisting of four unknowns in the form of matrixes, which are the matrixes of weights and biases of the two layers of the neural network. The process of network training needs to define the code, which can only be obtained by knowing the key of the original RSA algorithm. Thus, the degree of protection of the proposed mechanism depends on the degree of the original algorithm. It is equal when trying to break it by breaking the original algorithm and superior when trying to hack it by means of probability, because the number of unknowns in the key of the proposed method is greater than in the RSA algorithm. The degree of confidentiality of the proposed method can be increased by using dynamic keys and changing the weights through periodic network training, which increases the confidentiality and degree of protection used and the improbability of breaking it. The code resulting from the proposed mechanism is not necessarily Arabic letters, which increases the number of possibilities in front of the hacker, and the experimentation process does not work for penetration. With the presence of the order factor, a hacker may use experimental letters that give a correct arrangement in terms of language but are not necessarily the correct word, and this method needs scrutiny, which forces the hacker to analyze every possibility. Hackers are faced with an infinite number of possibilities, 'and they soon realize that the code is impossible to break. The encryption and decryption process used in the RSA algorithm depends on two equations with fixed functions; the method is clear to the hacker and the strength of the algorithm lies in the encryption key only. The proposed methodology uses two activation functions, and different activation functions can be used, not only the tansig and linear functions. This increases the confidentiality of encryption by adding a factor of insularity to the method and by the presence of more than one possibility to perform encryption and decryption by changing the activation functions according to their number and order of use in each layer. The process of generating keys in the RSA algorithm is dependent on a fixed methodology, and the strength of this methodology is a factor of randomness in choosing the cryptographic parameters represented by the two large primes, q and p. In the proposed methodology, the method of generating the keys is not fixed because an algorithm other than RSA can be used as the original algorithm to obtain training pairs. The proposed method depends on the original algorithm only for the encryption key to obtain the training pairs. After obtaining the code, it does not need the private key, which means that it needs only one key. Therefore, symmetrical algorithms, such as the AES algorithm, can also be used to design the encryption and decryption keys for the proposed method, which contributes to the reduction in the design time needed for encryption and decryption with neural networks. The proposed methodology is not affected by the original algorithm used (whether RSA or others) to design it in terms of encryption and decryption time, but it — the original algorithm — affects the values of weights and biases, and therefore the values of the encryption and decryption keys.

4. Conclusion

In this research, a methodology was proposed to encrypt and decrypt messages, written in Arabic, by using artificial intelligence represented by artificial neural networks, to determine how this network was designed to perform asymmetric encryption, and to discover how to obtain the private and public keys of the proposed mechanism. This mechanism proved highly effective and superior to the RSA algorithm in terms of execution time, both in encryption and decryption. In

addition, a software system was designed to perform encryption and decryption in the MatLab programming environment. It proved effective and gave an accuracy of 100% in retrieving 10,000 encrypted messages of different sizes. The average improvement in encryption and decryption time was 1,330ms and 4,497.5ms, respectively, for text message sizes of 50KB, 100KB, 150KB, and 200KB.

Biography

Mohammad Taha Kafarnawi

Computer and Automatic Control Engineering Department, University of Aleppo, Aleppo, Syria, mtktaha@gmail.com, 00963947478706

Dr Kafarnawi is a University of Aleppo graduate. He is a recipient of Al-Basel Award for Academic Excellence, ranked first. He obtained his PhD in Computer and Automatic Control Engineering, Tishreen University. He is a Member of the teaching staff at the University of Aleppo since 2010. He Published research papers in Aleppo University research, Tishreen University research, and the French journal Energy Procedia. He has a patent in field of artificial intelligence and control. He also teaches at Tishreen University, Ittihad University and Cordoba Private University.

References

Al-Abaidy, S.A. (2020). Artificial neural network based image encryption technique. *Int. J. Services Operations and Informatics*, **10**(3), 181–9.

Agarwal, N. and Agarwal, P. (2013). Use of artificial neural network in the field of security. *MIT International Journal of Computer Science & Information Technology*, **3**(1), 42–44.

Al Azawee, H., Husien, S. and Mohd Yunus, M.A. (2015). Encryption function on artificial neural network. *Springer, Neural Comput & Applic*. **2015**(3), a/b. DOI 10.1007/s00521-015-2028-3

Al Badawi, A., Chao, J., Lin, J. Mun, C.F., Jie Sim, J., Meng Tan, B.H., Nan, X.M., Aung, K.M. and Chandrasekhar, V. (2020). Towards the AlexNet moment for homomorphic encryption: HCNN, the first homomorphic CNN on encrypted data with GPUs. *IEEE Transactions on Emerging Topics in Computing*, **9**(n/a), 1330–43. DOI: 10.1109/TETC.2020.3014636.

Bevi, A.R., Tumu, S. and Prasad, N.V. (2018). Design and investigation of a chaotic neural network architecture for cryptographic applications. *Computers and Electrical Engineering/Elsevier*. **72**(2018) 179–90.

Chakraborty, M., Jana, B., Mandal, T. and Kule, M. (2018). A Performance Analysis of RSA Scheme using artificial neural network. In: *2018 9th International Conference On Computing, Communication and Networking Technologies (ICCCNT)*, Bengaluru, India, 10-12/07/2018.

Du, Y. and Stephanus, A. (2018). Levenberg-Marquardt neural network algorithm for degree of arteriovenous fistula stenosis classification using a dual optical photoplethysmography sensor. *Sensors (Basel)*, **18**(7), 1–18. DOI: 10.3390/s18072322.

Faraz, F.M., Maen, T.A. and Omidreza, K. (2013). A hybrid encryption algorithm based on RSA small-e and efficient-RSA for cloud computing environments. *Journal of Advances in Computer Network*, **1**(3), 238–41.

Forgá, R. and Okay, M. (2019). Contribution to symmetric cryptography by convolutional neural networks. *Slovak Academy of Sciences*, **2**(16), n/a.

George Amalarethinam, D.I. and Leena, H.M. (2017). Enhanced RSA algorithm with varying key sizes for data security in cloud. *IEEE,* **9**(17), 172–5.

Haghipour, S. and Sokouti, B. (2009). Approaches in RSA cryptosystem using artificial neural network. *Oriental Journal of Computer Science & Technology*, **2**(1), 11–17

Intila, C., Gerardo B. and Medina R. (2019). A study of public key 'e' in RSA algorithm. In: *International Conference on Information Technology and Digital Applications (ICITDA 2018),* Manila, Philippines, 08-09/11/2018. *IOP Conference Series: Materials Science and Engineering*, **482**(012016), 1–9. DOI:10.1088/1757-899X/482/1/012016

Kengnou Telem, A.N., Segning, C., Kenne, G. and Fotsin, H.B (2014). A simple and robust gray Image encryption scheme using chaotic logistic map and artificial neural network. *Hindawi Publishing Corporation*, **2014**(n/a), 1–13.

Lu, Z.M. and Mohamed, H. (2021). A complex encryption system design implemented by AES. *Journal of Information Security*, **12**(2), 177–87. DOI:10.4236/jis.2021.122009

Omar, G.A. and Shawkat, K.G. (2018). A survey on cryptography algorithms. *International Journal of Scientific and Research Publications (IJSRP)*, **8**(7), 495–516.

Shambhavi, D. and Sonal, S. (2018). Enhanced RSA algorithm for data security in cloud. *IRE Journals*, **1**(9), 2456–8880.

Taleb Obaid, A.S. (2020). Study a public key in RSA algorithm. *EJERS, European Journal of Engineering Research and Science*, **5**(4), 395–8.

Unicode Organization. Available at: https://home.unicode.org/ (accessed on 3/3/2021)

Yousif, E.Y. and ANabi, M.A.B. (2017). Performance enhancement of RSA algorithm using artificial neural networks. *International Journal of Computer Science and Mobile Computing*, **6**(9), 21–7.