Analysis of the applicability of artificial neural networks for the post-quantum cryptography algorithms development

S S Tarasenko¹, N A Andriyanov², A A Gladkikh³

¹ Russia FSS Academy, Oryol, Russia
² Financial University under the Government of the Russian Federation, Moscow, Russia
³ Ulyanovsk State Technical University, Ulyanovsk, Russia

E-mail: naandriyanov@fa.ru

Abstract. This article discusses the mathematical apparatus of artificial neural networks (ANNs) and analyzes the possibility of its application in the implementation of post-quantum asymmetric cryptosystems. The authors show that ANNs can be used for development such systems and analysis basic terms and processes in ANNs.

1. Introduction

Since the publication of algorithms that solve the problems of discrete logarithm and factorization of integers on a quantum computer in polynomial time by Peter Shor [1], the scientific community has been actively looking for mathematical problems for the implementation of asymmetric cryptography algorithms that are not based on the above problems of number theory.

In 2016, National Institute of Standards and Technology (NIST) announced the start of a competition to develop new standards for asymmetric encryption and digital signatures resistant to cryptanalysis using quantum computing [2]. On July 22, 2020, NIST announced the start of stage 3 of the competition for the choice of a post-quantum cryptography standard, in which asymmetric encryption algorithms on lattice-based cryptography [3] and error-correcting codes passed [4]. Candidates to the post-quantum standard, digital signatures are also based on lattices and multivariate transformations.

However, we should not exclude the possibility that one of the presented algorithms can be hacked, or it turns out that its real security is much lower than the theoretically calculated one, as was the case, for example, with asymmetric cryptography backpack schemes [5]. In this regard, the actual question is on the search for new mathematical algorithms, on the basis of which post-quantum asymmetric cryptosystems can be implemented. One of such solutions can be the mathematical apparatus of artificial neural networks (ANNs) and its application for solving problems of post-quantum cryptography.

The text deals with apparatus of ANNs after analyzing of which it is possible to conclude that ANNs may become a usefool tool for post-quantum cryptography.

2. Materials and methods

The using of ANNs also looks much prettier for mentioned task than using some stochastic dynamic equations, such as autoregressive [6] or doubly stochastic [7] models. At the moment, there are a large
number of different variations of artificial neural networks [8], but they all consist of certain primitives. Figure 1 shows the known structure of ANN part.

Thus, each neuron has (Figure 1):

- input values - graph nodes, which are values obtained from the previous (previous) layers of an artificial neural network, or values received at the input of an artificial neural network;
- synapses - edges of a graph with a certain weight, where the vertices of the graph are the input values of the neuron and the neuron kernel. Weight determines the degree of influence of the i-th input neuron on the resulting state of the neuron (layer output). The number of synapses of a neuron is equal to the number of input values of the neuron;
- axon - the edge of the graph connecting the neuron kernel with the neuron output. The number of axons of a neuron is equal to the number of output values of the neuron. In neural networks, the number of neuron axons is equal to the number of neuron synapses in the next layer of the ANN;
- the kernel of a neuron is a graph node that processes input values and synapses, and transfers the processing result to the output of a neuron through an axon.

Processing using the neuron kernel is determined by the following formula (1):

$$C = \sum_{i=1}^{n} x_i \cdot w_i,$$

where $x_i$ is the i-th input value of the neuron, $w_i$ is the weight of the i-th synapse, $n$ is the number of input values of the neuron, $C$ is the state of the neuron.

The value characterizing the axon of a neuron is calculated according to the formula (2):

$$U = f(C),$$

where $f(...)$ is the neuron activation function.

In artificial neural networks, the following functions are most often used as an activation function:

- logistic curve (sigmoid) function, defined by expression (3):

$$f(x) = \frac{1}{1 + e^{-x}};$$

- rectified linear unit (ReLU) function, defined by expression (4):
\[
    f(x) = \begin{cases} 
    0, & x < 0 \\
    x, & x \geq 0 
    \end{cases}; \quad (4)
\]

- leaky rectified linear unit (Leaky ReLU) function, defined by expression (5):

\[
    f(x) = \begin{cases} 
    0.01 \cdot x, & x < 0 \\
    x, & x \geq 0 
    \end{cases}; \quad (5)
\]

- hyperbolic tangent function, defined by expression (6):

\[
    f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}. \quad (6)
\]

There are a number of alternative neuron activation functions such as exponential linear unit (ELU), identity function, Heaviside function (step function), inverse square root unit (ISRU), parametric linear rectification function (PreLU), softPlus, bent identity function.

The function of activating neurons in a network should preferably have the following properties:

- Non-linearity of the function. If the activation function has this property, there is evidence that a two-level ANN can approximate an arbitrary function [9].

- Differentiability of the function over the entire range of admissible values. Possession of the function of this property will allow training an ANN using optimization methods based on gradient descent.

Training ANNs using the backpropagation method, based on the gradient descent method, means that the network weights (synapse weights) change in such a way that the average error generated at the output of the neural network when the sequence arrives at the input training data decreased.

The integral element in training artificial neural networks using the backpropagation method is the network estimation function (loss function), which characterizes the deviation of the real result of the network from the required one. The most common function for evaluating the performance of an artificial neural network is a quadratic loss function \(L\) that calculates the sum of the squares of the differences between the output values of the neural network and the required values according to the formula (7):

\[
    L = \frac{1}{2} \sum_{i=1}^{r} (y^*_i - y_i)^2, \quad (7)
\]

where \(r\) is the number of neurons in the last layer of the neural network, \(y_i\) is the real value of the \(i\)-th output of the neural network, \(y^*_i\) is the required value of the \(i\)-th output of the neural network.

The method of assessing the performance of an ANN presented by (7) is called the least squares method.

There are several implementations of the error backpropagation method:

- Stochastic gradient descent. This implementation, immediately after calculating the output values of the neural network, calculates and applies corrections to the weights of the neural network synapses.

- Batch Gradient Descent is an implementation in which a network performance score is computed for all training inputs, after the end of the training epoch.

- Mini-batch gradient descent is an implementation in which a network performance score is computed for all training inputs contained in a single batch after the end of the training epoch. For better network training, the data contained in the batches is randomly selected from the entire training data set at each training epoch of the neural network.

The essence of the error backpropagation algorithm is reduced to the following steps:
• **step 1:** using the network estimation function to find the amount of losses during the operation of the neural network according to formula (7);

• **step 2:** finding a correction \( \delta_i \) for the weights of the synapses of the ANN last layer nodes using formula (8):

\[
\delta_i = -2\alpha y_i(1 - y_i)(y_i^* - y_i),
\]

where \( \alpha \) is an arbitrary constant;

• **step 3:** finding a correction \( \delta^* \) for the weights of the synapses of the nodes of a lower level through the corrections for the weights of the synapses of the nodes of a higher level using formula (9):

\[
\delta^*_i = 2\alpha y_i(1 - y_i) \sum_{k=1}^{N_i} \delta_i w_{i,k},
\]

where \( w_{i,k} \) is the weight of the synapse connecting the \( i \)-th neuron of the current layer of the neural network with the \( k \)-th neuron of the previous layer of the neural network, \( N_i \) is the number of neurons in the previous neural network layer;

• **step 4:** calculation of the value by which it is necessary to change the weight of each synapse in the neural network according to the formula (10):

\[
\Delta w_{i,j} = -\eta \delta_j \frac{\partial C_j}{\partial w_{i,j}},
\]

where \( i \) is the serial number of the synapse of the neuron, \( j \) is the serial number of the neuron in the neural network, \( \eta \) \((0 < \eta < 1)\) is the coefficient that determines the rate of change in the weights of the synapses when training the ANN;

• **step 5:** application of the calculated changes \( \Delta w_{i,j} \) for the weights of the neural network corresponding synapses.

It is necessary to note that in order to concretize the example the overview of the error backpropagation method based on stochastic gradient descent is given above.

### 3. Conclusion

The analysis of the mathematical apparatus of artificial neural networks shows that it is based on the calculation of loss functions, derivatives and the calculation of nonlinear activation functions of neurons in the network. These calculations do not include the problems of discrete logarithm or factorization of integers, which allows concluding that cryptographic algorithms based on the mathematical basis of such neural networks can not be cryptanalyzed using Shor's quantum algorithms [1].

In addition, since computations in neural networks lend themselves to parallelization, it is possible to gain in the performance of cryptoalgorithms when approximating tasks that can not be parallelized using neural networks. A stable asymmetric cryptographic system, according to [12], is a system whose cryptographic strength is based on the computational complexity of solving an NP-complete problem. At the moment, not a single algorithm has been found that allows using calculations on a quantum computer to obtain a solution to an NP-complete problem in polynomial time. Since it is possible to approximate any function using artificial neural networks, the implementation of an asymmetric cryptosystem on the ANNs mathematical apparatus resistant to quantum cryptanalysis is possible. A practical confirmation of the above conclusion is the protocol for the exchange of key
information between two subscribers, based on the synchronization of two tree-like parity machines (TPM) [11].

Thus, summarizing the above mentioned aspects, it can be argued that there is a potential for the use of the mathematical apparatus of artificial neural networks in the development of post-quantum cryptography algorithms and further practical use of these algorithms in modern network security protocols.

References

[1] Shor P 1994 *Foundations of Computer Science: Proceedings, 35th Annual Symposium on — IEEE* 124-134 doi:10.1109/SFCS.1994.365700
[2] Official page of the NIST Post-Quantum Cryptography Standardization, *url: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization*, last accessed 05/25/2021
[3] Goldreich O, Goldwasser S, Halevi S 1997 *CRYPTO ’97: Proceedings of the 17th Annual International Cryptography Conference on Advances in Cryptology* 112–131
[4] McEliece R J 1978 *DSN Progress Report* 44 114–116
[5] Adi S 1982 *CRYPTO ’97: Proceedings of the 2nd Annual International Cryptography Conference on Advances in Cryptology* 303-308
[6] Andriyanov N A, Sluzhivyi M N 2019 *CEUR Workshop Proceedings* 2391 79–85 doi: 10.18287/1613-0073-2019-2391-79-85
[7] Vasiliev K K, Dementyev V E, Andriyanov N A 2019 *Journal of Physics: Conference Series* 1368 032019 doi: 10.1088/1742-6596/1368/3/032019
[8] Leijnen S, Veen F 2020 *Proceedings* 47(9) 1–6
[9] Leshno M, Lin V Ya, Pinkus A, Schocken S 1993 *Neural Networks* 6(6) 861-867 doi:10.1016/S0893-6080(05)80131-5. S2CID 206089312
[10] Schneier B 1994 *Applied Cryptography, John Wiley & Sons* ISBN 0-471-59756-2.
[11] Kinzel W, Kanter I 2002 *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP ’02* 3 1351–1354 doi: 10.1109/ICONIP.2002.1202841.