The BlockChain Neural Network: Neuron as a Service

Will Serrano
Intelligent Systems and Networks
Alumni Imperial College London
London, United Kingdom
g.serrano11@alumni.imperial.ac.uk

***Abstract*—The BlockChain Neural Network increments gradually neurons as user information or data increases, offering a biologically inspired "Neuron as a Service" (NaaS) solution. The additional neurons codify both the new information to be added to the "neural block" and the previous neurons potential to form the "neural chain". This configuration provides the proposed algorithm with the same properties found in the BlockChain: security and decentralization with the same validation process: mining the input neurons until the neural network solution is found. The mathematical model of the Neural Networks in BlockChain configuration is presented in this paper with evidences of its stable learning convergence during its learning process. The main advantage of this research proposal is the biological simplicity of the solution, however it suffers high computational cost when the number of neurons increase. Experimental results that validate the proposed method are presented with optimistic conclusions; this paper provides the biologically inspired mathematical model as a digital step forward to avoid physical currencies, documentation and contracts.**

***Index Terms*—Neural Network, BlockChain, Neuron as a Service**

I. Introduction

The concept of "as a Service" outsources the management and ownership responsibilities to a third party service provider; it also increases efficiency and reduces cost as the service provider optimizes the usage of the assets that are required to deliver the service. The inconvenient truth is that business performance relies on the service provider in key aspects such as Cybersecurity; to address this dependency, Service Level Agreements (SLA) and free market competition forces both parties to perform optimally. Key business applications of the "as a Service" concept are data centers, that optimize CPU, memory, software, power and networks based on user data demand. Similarly, Mobility as a Service (MaaS) detaches the ownership between the user and the vehicle including the related liabilities such as taxes, parking, and maintenance by paying for service based on usage such as time or distance. The key decision to be made is the balance between the additional costs associated with the services a business outsources and the assets it owns. This paper presents the concept: "Neuron as a Service", a biologically inspired model where neurons are gradually incremented on demand as user information expands.

BlockChain enables the digitalization of contracts as (1) authentication between the parties within the agreement, (2) encryption of information that gradually increases in size as more data is generated between the parties, and finally (3) validation of the encrypted information that is stored in a decentralized network externally from the relevant parties. Due to these key properties, BlockChain functionality has several applications such as Cryptocurrency [1], Smart Contracts [2] [3] [4], Smart Cities [5], 5G in Intelligent Buildings [6] and Internet of Things [7] [8]. BlockChain applications mostly cover the digitalization of physical agreements based on paper eliminating the need for an external supervisor or authorizer that generates the third party in the agreement.

This paper presents a BlockChain system where each successive "neural blocks" contains additional neurons and synapses so that neurons in successive blocks are interconnected and chained. Additional neurons codify (1) the new data to be inserted within the "neural block" and (2) the potential of the previous neurons generates the "neural chain". The BlockChain Neural Network has analogue biological properties as the BlockChain:

- user authentication based on a mining process that randomly assigns potential to the input neurons until the neural network solution is found
- data encryption as information is contained in neural network weights rather than the neurons
- user information is gradually incremented and learned with additional neurons and stored in a decentralised network

The key benefit of the presented neural network in a new BlockChain configuration is the biological simplicity of the solution however it suffers high computational cost when neurons increase. Section 2 of this paper describes the Brain and BlockChain related work, Section 3 presents the Random Neural Network whereas Section 4 describes Long Short Term Memory Networks.Section 5 proposes the BlockChain Neural Network model. Section 6 evaluates the proposed model whereas finally Section 7 shares the conclusions.

II. Brain and BlockChain related work

This paper proposes "Neuron as a Service" (NaaS) where neurons are added as information increases. This method emulates the biological brain, in particular the Short Term Memory, with an approach very similar to how a datacentre handles and stores information. The number of neurons in the brain is approximately 10^{11}, this figure is similar to the number of stars of our galaxy or the storage of a standard datacentre, at a very reduced weight (approximate 1.5 kg) [9] which shows the brain neural density. Although every neuron has the same genes and components (a cell body, a dendrite and an axon), each neuron is different in its form and connectivity, therefore remains unique. Spikes generated by a neuron travel through its axon as an electrical impulse, however the signal is transmitted between the emitting axon and the receiver dendrite via specialised molecules in the synapse. Normally neurons are connected to and from hundreds or thousands of other neurons in very specific structures highly closed to each other creating specialised clusters of neurons that perform different functions [10]. Single neurons in the brain perform key specific functions in the short term memory with precise firing patterns. Short term memory is active only during very short periods of time, normally in the scale of seconds, therefore it avoids the conscious stage; this feature makes short term memory fundamental during the human learning process due its fast processing, although the drawback is that the learned knowledge is not included in the long term memory [11]. Long term memory in the brain is distributed in several locations rather than centralised into specific neurons where the representation of any single event is enabled by chains of neurons [12]; this process emulates the redundancy of datacentres where the same data is stored in dispersed sites. The activation and conversion between short term memory into long term memory in the brain is fairly independent of the specific details of neuron functions. Short term memory is activated by external stimulus which can be artificially maintained by persistent neuron firing for periods of seconds after the removal of the stimuli in spiking neural network models, although neurons cannot maintain accurate graded levels of activity for longer time periods as they eventually relax to one of the constant activities or equilibrium where random noise generates different equilibriums [13].

In addition, the neural synapses of the "Neuron as a Service" model are chained to form a neural chain with the same functionality as the BlockChain. This functionality also emulates the process of thinking when thoughts move from topic to topic based on chains of concepts or ideas. The thinking process of the brain and the intelligent human behaviour is based on some general principles that manage the structure and operation of the neural circuits [14]. At cellular level, there is standardization in the way information is transmitted, however, the thinking process involves a selection of methods and a following adaption in order to calculate the right concept based on several optimizations. Goals and the required optimizations to achieve them are the foundations of thinking and decision making which can be modelled as a descriptive and prescriptive search inference process, where thinking is described as conclusions generated from possibilities, evidence and goals that are discovered through searching [15]. Thinking is conceived as a method of choosing among potential options where possibilities might consist of actions, beliefs or personal goals. These choices are based on a search for relevant information and the implications which are made from the information obtained; information is broadly defined to include goals, choices, and evidence where choices are evaluated on the basis of evidence in light of these goals. The cognitive mechanisms of thinking have been described through the three major components [16] of problem solving that includes planning and design. While 1) reasoning focuses on drawing deductive conclusions and the process to make them, 2) judgment and 3) decision-making involve the rate of wrong arguments that make use of the formal structure of probability theory. BlockChain thinking defines thinking as a BlockChain process where all input elements are discrete units that can be encoded and stored; processing is generated in a massively distributed computing architecture that includes the nonlinearity of human thought [17]. The BlockChain thinking solution proposes the integration of Artificial Intelligence with potentially new consensus models for self-mining and intelligence in which neural networks are similar to currencies that are distributed as knowledge and ideas.

Neural Networks have also been already applied to Cryptography for smart card applications [18]. Two multilayer neural networks on their mutual output bits are trained with discrete weights to achieve a synchronization that can be applied to exchange secret keys over a public channel [19]. Three crypt-analytic attacks (genetic, geometric and probabilistic) are launched to the above neural network [20] to evaluate its performance and functionality. Feed forward neural networks have also been applied as an encryption and decryption algorithm with a permanently changing key [21]. A two stage cryptography multilayered neural network consists in a first stage to generate neural network-based pseudo random numbers and a second stage that encrypts information based on the non-linearity of the neural network [22].

III. The Random Neural Network

The Random Neural Network was introduced in [23] and many of its properties have been developed in [23] [24] [25] [26]. It is a spiking stochastic model which can be used as either a feedforward or feedback (recurrent) network (Fig. 1). The Random Neural Network consists on n-neurons. The state

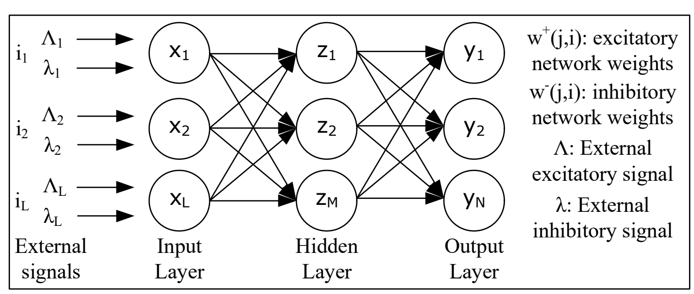

Fig. 1. The Random Neural Network structure.

of the n neuron network at time t is represented by the vector of non-negative integers $K(t) = [K_1(t), \ldots K_i(t)]$ where $K_i(t)$ is the potential of neuron i at time t. Neurons interact with each other by interchanging signals in the form of spikes of unit amplitude:

- A positive spike is interpreted as an excitation signal because it increases by one unit the potential of the receiving neuron m, $K_m(t^+) = K_m(t) + 1$;
- A negative spike is interpreted as inhibition signal decreasing by one unit the potential of the receiving neuron m, $K_m(t^+) = K_m(t) - 1$, or has no effect if the potential is already zero, $K_m(t) = 0$.

Each neuron accumulates signals and it will fire if its potential is positive. Firing will occur at random and spikes will be sent out at rate r(i) with independent, identically and exponentially distributed inter-spike intervals:

- Positive spikes will go out to neuron m with probability $p^+(i, m)$ as excitatory signals;
- Negative spikes will go to neuron m with probability $p^-(i, m)$ as inhibitory signals.

Neuron i can also send send spikes out of the network with probability d(i), so that:

$$d(i) + \sum_{j=1}^{n} [p^+(i, m) + p^-(i, m)] = 1 \quad for \quad 1 \leq i \leq n \tag{1}$$

The potential of any neuron i decreases by one unit when the neuron fires either an excitatory or an inhibitory spike. External (or exogenous) excitatory or inhibitory signals to neuron i can also arrive at rates $\Lambda(i), \lambda(i)$ respectively according to stationary Poisson processes.

A. Mathematical Model

The Random Neural Network excitatory and inhibitory weight parameters $w^+(j, i)$ and $w^-(j, i)$, respectively are directly related to the r(i) and the $p^+(i, j)$, $p^-(i, j)$, and are expressed as:

$$w^+(j, i) = r(i)p^+(i, j) \geq 0 \tag{2}$$

$$w^-(j, i) = r(i)p^-(i, j) \geq 0 \tag{3}$$

Thus information in this model is transmitted by the rate or frequency at which spikes travel. Each neuron i, if it is excited, behaves as a frequency modulator emitting spikes at rate $w(i, j) = w^+(i, j) + w^-(i, j)$ to neuron j.

Thus each neuron j acts as a non-linear frequency demodulator transforming the incoming excitatory and inhibitory spikes into the "excitation level" or potential potential $K_j(t)$.

This network model has a product form solution; i.e. the network's stationary probability distribution is the product of the marginal probabilities of the state of each neuron as shown in [23], where the further result is shown. Let p(k,t) = Prob[K(t) = k)] and the marginal probability a neuron i is excited at time t as $q_i(t) = Prob[k_i(t) > 0]$. The stationary probability distribution $p(k) = lim_{t\to\infty} p(k, t)$ and $q_i = lim_{t\to\infty} q_i(t)$ where k(t) is a continuous time Markov chain that satisfies Chapman-Kolmogorov equations.

Let's define:

$$q_i = \frac{\lambda^+(i)}{r(i) + \lambda^-(i)} \tag{4}$$

$$r(i) + \sum_{j=1}^{n} \left[w^+(i, j) + w^-(i, j) \right] \quad for \quad 1 \leq i \leq n \tag{5}$$

where the $\lambda^+(i)$, $\lambda^-(i)$ for i=1,...,n satisfy the system of nonlinear simultaneous equations:

$$\lambda^+(i) = \sum_{j=1}^{n} \left[q_j r(j) p^+(j, i) \right] + \Lambda(i) \tag{6}$$

$$\lambda^-(i) = \sum_{j=1}^{n} \left[q_j r(j) p^-(j, i) \right] + \lambda(i) \tag{7}$$

If a nonnegative solution $\lambda^+(i), \lambda^-(i)$ exists to the equations (4), (6) and (7) that meets $q_i < 1$ then:

$$p(k) = \prod_{i=1}^{n} [1 - q_i] \, q_i^{ki} \tag{8}$$

The network will be stable if a value $q_i < 1$ can be found. The average potential at a neuron i is $q_i/[1 - q_i]$ and the rate of emission of spikes from neuron i in steady state is $q_i r(i)$. If we have $\lambda^+(i) > [r(i) + \lambda^-(i)]$ for any neuron means that the neuron is unstable or saturated; this implies that it is constantly excited in steady state and its rate of excitatory and inhibitory spike emission r(i) to another neuron j will be $r(i)p^+(i, j)$ and $r(i)p^-(i, j)$ respectively (Fig. 2).

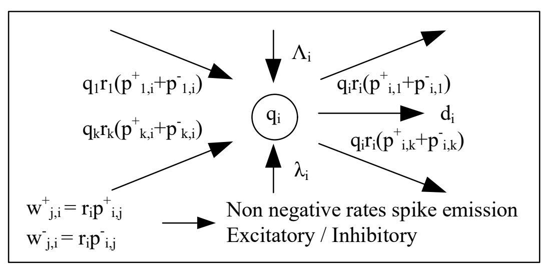

Fig. 2. The Random Neural Network model.

B. Learning Algorithm

Neural networks are capable to learn iteratively through examples or training sets. The two main methods are supervised learning based on an input with the desired output and Reinforcement Learning based on the environment reactions from user actions. The Random Neural Network learning algorithm proposed by Erol Gelenbe [26] is based on gradient descent of a quadratic error function. The backpropagation model requires the solution of n linear and n nonlinear equations each time the n neuron network learns a new input and output pair.

IV. Long Short-Term Memory Network

Long Short Term Memory (LSTM) networks are widely used in time series data as their learning algorithm does not present exploding and vanishing gradient descent issues as traditional recurrent Neural Networks with back propagation Learning Algorithms (Fig. 3). LSTM networks are a type of artificial recurrent neural network composed of the following elements:

- the cell c_t that provides memory to the neural structure
- the input gate i_t controls the relevance of new sensorial activity to the cell
- the forget gate f_t manages the relevance of existing sensorial information stored in the cell
- the output gate o_t modulates the stimuli the current cells transmits to the next cell in the neural chain

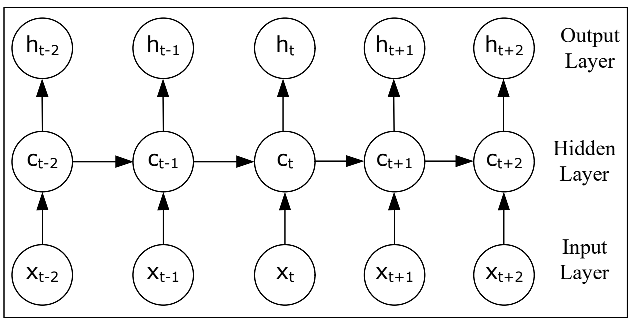

Fig. 3. Long Short Term Memory Network.

A. Mathematical Model

LSTM network enables a constant error flow through self-connected memory cells (Fig. 4), where input and output gates manage its transmission while protecting it from perturbations [27], although error signals contained within the memory cell can not be altered. Specifically, a LSTM network with P input cells and Q output cells is defined as:

- The forget vector $f_t \epsilon R^Q$:

$$f_t = \sigma_g \left(\sum_{p=1}^{P} w_{fp} x_t + \sum_{q=1}^{Q} u_{qf} h_{t-1} + b_f \right) \tag{9}$$

- The input activation vector $i_t \epsilon R^Q$:

$$i_t = \sigma_g \left(\sum_{p=1}^{P} w_{ip} x_t + \sum_{q=1}^{Q} u_{iq} h_{t-1} + b_i \right) \tag{10}$$

- The output activation vector $o_t \epsilon R^Q$:

$$o_t = \sigma_g \left(\sum_{p=1}^{P} w_{op} x_t + \sum_{q=1}^{Q} u_{oq} h_{t-1} + b_o \right) \tag{11}$$

- The cell state vector $c_t \epsilon R^Q$:

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c \left(\sum_{p=1}^{P} w_{cp} x_t + \sum_{q=1}^{Q} u_{cq} h_{t-1} + b_c \right) \tag{12}$$

- The hidden state vector $h_t \epsilon R^Q$:

$$h_t = o_t \circ \sigma_c(c_t) \tag{13}$$

where:

- $x_t \epsilon R^P$ is the input vector to the LSTM network
- $w \epsilon R^{QxP}$ is the weight matrix for the input vector
- $u \epsilon R^{QxQ}$ is the weight matrix for the hidden state vector
- $b \epsilon R^Q$ is the bias vector
- σ_g represents the sigmoid function
- σ_c represents the hyperbolic tangent function

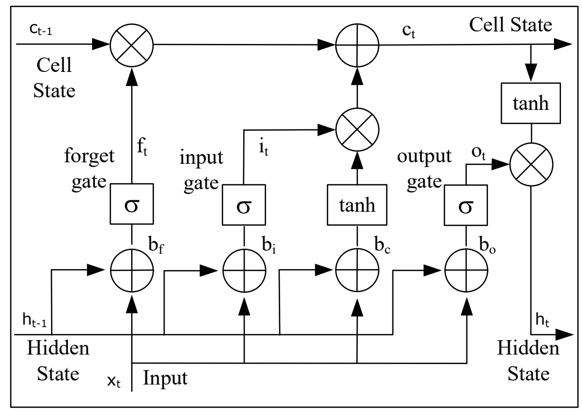

Fig. 4. Long Short Term Memory Cell.

B. Learning Algorithm

The Long Short Term Memory network learning algorithm proposed by Sepp Hochreiter and Jürgen Schmidhuber [27] is based on the Back Propagation Through Time (BPTT) learning algorithm that truncates the error flow making it constant when it departs memory cell. Errors arriving to memory cells do not get propagated back further in time although they affect the memory cell it arrives. Only within the memory cell, errors are backpropagated through previous internal states. The cell gates ensure that there are no feedback in the network as they disable the re-entrance of an error that has already departed a memory cell. The LSTM learning algorithm has a complexity of O(w) or O(u) where w or u is the number of neural weights.

V. The BlockChain Neural Network

The BlockChain Neural Network model consists of L Input Neurons, M hidden neurons and N output neurons Network. Information in this model is contained within the neural network weights W rather than neurons x_L, z_M, y_N:

- $X = (x_1, x_2, \ldots, x_L)$, a variable L-dimensional vector $X\epsilon[0,1]^L$ represents the input state q_L for the neuron L; where scalar L values range $1 < L < \infty$;
- $Z = (z_1, z_2, \ldots, z_M)$, a M-dimensional vector $Z\epsilon[0,1]^M$ that represents the hidden neuron state q_M for the neuron M; where scalar M values range $1 < M < \infty$;
- $Y = (y_1, y_2, \ldots, y_N)$, a N-dimensional vector $Y\epsilon[0,1]^N$ that represents the neuron output state q_N for the neuron N; where scalar N values range $1 < N < \infty$;

- W(j,i) is the (L+M+N) x (L+M+N) matrix of network weights that represents the connection from neuron j to neuron i; where $i\epsilon[x_L, z_M, y_N]$ and $j\epsilon[x_L, z_M, y_N]$;

The key concept of the Neural Network BlockChain is that the neuron vector sizes, L, M and N are not fixed; they are variable. The input layer X represents the user's incremental data; the hidden layer Z represents the values of the chain and the output layer Y represents the user Private Key (Fig. 5). Neurons or blocks are iteratively added where the value of the additional neurons consists on both the value of the additional information and the value of previous neurons therefore forming a neural chain. Information in this model is transmitted in the neural network weigh matrix, W rather than in the neurons rather than neurons x_L, z_M, y_N.

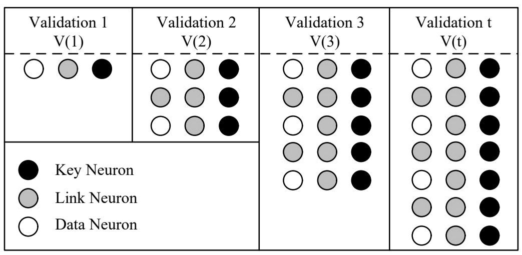

Fig. 5. The BlockChain Neural Network model.

A. The BlockChain Neural Network model

The Random Neural Network BlockChain model is based on the main concepts shown on Figure 6":

- Private key, y_N;
- Validation, V(t) and Data D;
- Neural Chain Network and Mining;
- Decentralized information, W(j,i).

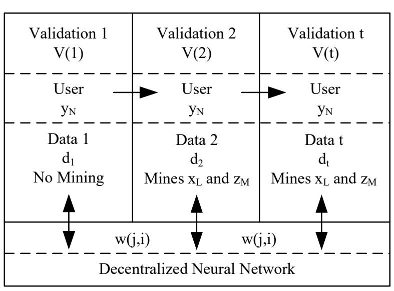

Fig. 6. The BlockChain Neural Network model.

B. Private key

The private key $Y = (y_1, y_2, \ldots, y_N)$ consists on the user or application digital credentials specifically assigned to them such as biometrics or any other cryptographic algorithm such as Advanced Encryption Standard (AES) 256-bit cipher. The private key is presented by the user, or the application, every time new data is to be inserted therefore credentials require validation.

C. Validation and Data

The neural network block chain model defines Validation and Data as:

- Validation, $V(t) = V(1), V(2), \ldots V(t)$ as a variable accumulative vector where t is the validation stage;
- Data, $D = d_1, d_2, \ldots d_t$ as a set of t I-vectors where $d_o = (e_{o1}, e_{o2}, \ldots e_{oI})$ and e_o are the I different dimensions for o=1,2, ... t.

The first Validation V(1) has associated an input state $X = x_I$ that corresponds d_1 representing the user data. The output state $Y = y_N$ corresponds to the user Private Key and the hidden layer $Z = z_M$ corresponds to the value of the neural chain that will be inserted in the input layer for the next transaction. The calculated network weights W(j,i) are stored in the decentralized network and are retrieved in the mining process. The second Validation V(2) has associated an input state $X = x_I$ that corresponds to the user data d_1 for the first Transaction T(1), the chain, or the value of the hidden layer z_M and the additional user data d_2. The output state $Y = y_N$ still corresponds the user Private Key and the hidden layer $Z = z_M$ corresponds to the value of the neural chain for the next transaction. This process iterates as more user data is inserted. The neural chain can be formed of the values of the entire hidden layer neurons, a selection of neurons, or any other combination to avoid the reverse engineering of the private key from the stored neural weights.

D. Neural Chain Network and Mining

Data is validated or mined by calculating the outputs of the Random Neural Network using the transmitted network weighs, W(j,i) at variable random inputs $X = x_I$, or following any other method. The Random Neural Network with BlockChain configuration is mined when an Input I is found that delivers an output Y with an error E_k lesser than a Mining Threshold T_M for the retrieved user network weights W(j,i):

$$E_k = \frac{1}{2}\sum_{n=1}^{N}\left(y_i^{'} - y_n\right)^2 < T_M \tag{14}$$

where E_k is the error of the BlockChain Neural Network, $y_n^{'}$ is the output of the Random Neural Network with mining or random input $X = x_I$ and y_n is the user or application Private Key. The mining complexity can be tuned by adjusting E_k. Once the solution is found or mined the user or application data can be processed, the values of the hidden layer are used in the input of the next transaction, along with the new data. the potential value of the neural hidden layer $Z = z_M$ is added to form the Neural Chain as the input of the next transaction where more user data is added. Finally, the proposed method calculates the Random Neural Network with gradient descent learning algorithm for the new pair (I, Y) when the E_k) is lesser than a Learning Threshold T_L, following the described neural netwokrs learning algorithms. The new generated network weights W(j,i)are stored in the decentralized network. The more user data and validations; the mining process increases on complexity.

E. Decentralized Information

The user network weights W(j,i) are stored in the decentralized network rather than its data I directly where I is calculated with the mining process. The network weights expand as more verification data is inserted creating an adaptable method. In addition; only the user Data can be extracted when the user presents its biometric key therefore making secure to store information in a decentralized system.

VI. Validation and Experimental Results

The BlockChain Neural Network is validated with the Random Neural Network (RNN) and a Long Short Term Memory Network (LSTM). The algorithm is run 100 times to obtain statistically significant experimental results. The LSTM network requires more learning iterations than the RNN model and both networks learn faster as the Validation V(t) stages gradually progress (Fig. 7). The size of the neural chain has an impact during the learning stage as the increment of neurons in the hidden layer increases the number of learning iterations.

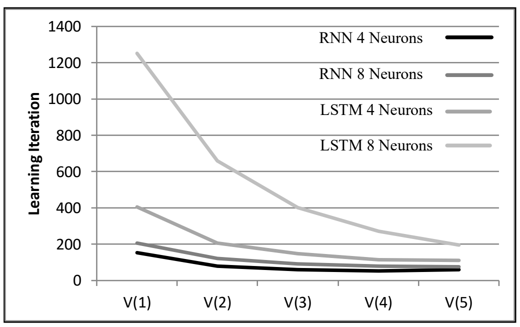

Fig. 7. The BlockChain Neural Network model.

A. Four neuron Neural Chain

This section presents the validation and experimental results for the private key $Y = y_N$, data $D = d_t$ and the neural chain $Z = z_M$ of 4 neurons. The potential value of the neurons for each data input d_t is shown on Table I, the chosen values are dispersed between the possible values [0-1].

TABLE I
FOUR NEURON NEURAL CHAIN CONFIGURATION

d_1	d_2	d_3	d_4	d_5	y_N
0.12	0.32	0.52	0.72	0.92	0.2
0.14	0.34	0.54	0.74	0.94	0.4
0.16	0.36	0.56	0.76	0.96	0.6
0.18	0.38	0.58	0.78	0.98	0.8

Table II shows the results for a 4 Neuron RNN with a Learning Threshold T_L=1.0E-30 and a Mining Threshold T_M=1.0E-5. This includes the number of learning iterations the BlockChain Neuron, the number of iterations to mine the BlochChain and finally the number of neurons for each layer; input x_L, hidden z_M and output y_N.

TABLE II
FOUR NEURON RNN VALIDATION

Stage V(t)	Learning Iteration	Learning Error	Mining Iteration	Mining Error	Neurons x_L,z_M,y_N
V(1)	153	7.53E-31	6.45E+03	3.41E-06	04-04-04
V(2)	79.87	6.09E-31	2.95E+05	3.75E-06	12-04-04
V(3)	60.12	6.64E-31	1.71E+05	3.44E-06	20-04-04
V(4)	54.49	6.08E-31	9.17E+03	3.58E-06	28-04-04
V(5)	58.97	4.16E-31	3.19E+02	3.64E-06	36-04-04

The number of Mining Iterations is not linear in contrast to the linear increment of user data (Fig. 8).

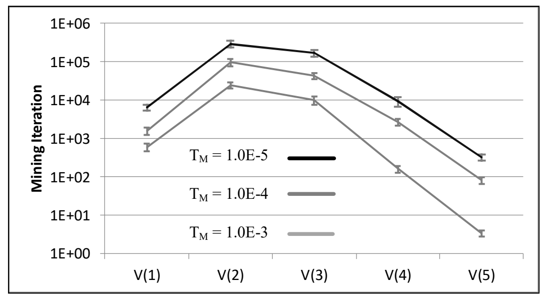

Fig. 8. The Four Neuron RNN validation.

Table III shows the results for a 4 Neuron LSTM with a Learning Threshold T_L=1.0E-30 and a Mining Threshold T_M=0.5E-2.

TABLE III
FOUR NEURON LSTM VALIDATION

Stage V(t)	Learning Iteration	Learning Error	Mining Iteration	Mining Error	Neurons x_L,z_M,y_N
V(1)	406	8.02E-31	3.43E+05	3.25E-06	04-04-04
V(2)	208.32	8.62E-31	2.13E+05	4.04E-03	12-04-04
V(3)	147	7.56E-31	5.23E+04	3.96E-03	20-04-04
V(4)	115.12	8.34E-31	3.98E+05	4.12E-03	28-04-04
V(5)	108	4.22E-31	1.36E+02	3.63E-03	36-04-04

Similar to the previous validation, the effort to mine the BlockChain LSTM Network does not increase linearly with the user data (Fig. 9).

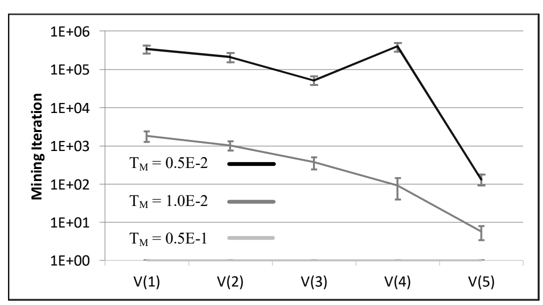

Fig. 9. The Four Neuron LSTM validation.

B. Eight neuron Neural Chain

This section presents the validation and experimental results for the private key $Y = y_N$, data $D = d_t$, and the neural chain $Z = z_M$ of 8 neurons. The potential value of the neurons for each data input d is shown on Table IV, similar to the previous validation, the chosen values are dispersed between the possible values [0-1].

TABLE IV
EIGHT NEURON NEURAL CHAIN CONFIGURATION

d_1	d_2	d_3	d_4	d_5	y_N
0.12	0.32	0.52	0.72	0.92	0.2
0.13	0.33	0.53	0.73	0.93	0.3
0.14	0.34	0.54	0.74	0.94	0.4
0.15	0.35	0.55	0.75	0.94	0.5
0.16	0.36	0.56	0.76	0.96	0.6
0.17	0.37	0.57	0.77	0.96	0.7
0.18	0.38	0.58	0.78	0.98	0.8
0.19	0.39	0.59	0.79	0.99	0.9

Table V shows the results for a 8 Neuron RNN with a Learning Threshold T_L=1.0E-30 and a Mining Threshold T_M=1.0E-5.

TABLE V
EIGHT NEURON RNN VALIDATION

Stage V(t)	Learning Iteration	Learning Error	Mining Iteration	Mining Error	Neurons x_L,z_M,y_N
V(1)	206	9.12E-31	1.80E+05	3.25E-06	08-04-04
V(2)	120.88	7.46E-31	2.25E+06	3.15E-06	20-04-04
V(3)	91.13	7.40E-31	1.04E+05	4.04E-06	32-04-04
V(4)	80.71	6.27E-31	7.12E+02	3.40E-06	44-04-04
V(5)	76.98	5.12E-31	2.26E+01	3.81E-06	56-04-04

The mining process for a ”neural chain” of neurons does not follow the linear pattern of the increment of user data or number of input neurons (Fig. 10), although the number of mining iterations has increased from a neural chain from four neurons to eight neurons. The process of mining is easier when at final validation stages.

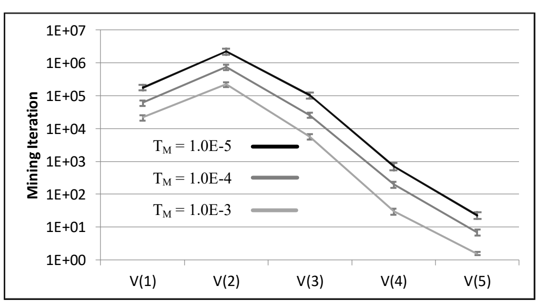

Fig. 10. The Eight Neuron RNN validation.

Table VI shows the results for a 8 Neuron LSTM with a Learning Threshold T_L=1.0E-30 and a Mining Threshold T_M=1.5E-2.

TABLE VI
EIGHT NEURON LSTM VALIDATION

Stage V(t)	Learning Iteration	Learning Error	Mining Iteration	Mining Error	Neurons x_L,z_M,y_N
V(1)	1251	6.81E-31	9.56E+05	1.31E-02	08-04-04
V(2)	660.4	9.78E-31	2.06E+06	1.38E-02	20-04-04
V(3)	401.3	9.39E-31	1.53E+06	1.31E-02	32-04-04
V(4)	271.9	8.37E-31	8.79E+07	1.24E-02	44-04-04
V(5)	193.5	8.41E-31	4.88E+03	1.21E-02	56-04-04

Similar to the previous validations, the increment of the ”neural chain” also increments the number of learning iterations, although the mining effort decreases with the validation stage. This effect follows a linear pattern alongside with the increasing user information. The Mining Threshold tunes the mining effort of the proposed BlockChain Neural network (Fig. 11).

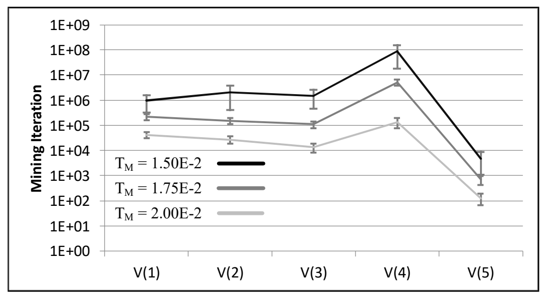

Fig. 11. The Eight Neuron LSTM validation.

C. Tampering validation

In order to be effective, the BlockChain Random Neural Network algorithm shall also detect tampering in the learned data. Table VII and Table VIII shows the tampering error where Δ represents the number of tampered neurons.

TABLE VII
RNN AND LSTM TAMPERING VALIDATION 4 NEURON CHAIN

Stage V(t)	RNN Δ=0.0	RNN Δ=1.0	LSTM Δ=0.0	LSTM Δ=1.0	Neurons x_L,z_M,y_N
V(1)	7.53E-31	2.24E-02	8.02E-31	8.00E-05	04-04-04
V(2)	7.08E-31	4.37E-04	9.99E-31	7.63E-05	12-04-04
V(3)	5.56E-31	3.62E-05	7.80E-31	7.08E-05	20-04-04
V(4)	5.40E-31	5.70E-06	8.17E-31	6.40E-05	28-04-04
V(5)	7.39E-31	1.30E-06	3.20E-31	5.66E-05	36-04-04

There is a clear difference within the Blockchain error between the real and the tampered values. Both configurations, RNN and LSTM, have similar error values although LSTM is more constant through the different validation stages than the RNN.

TABLE VIII
RNN AND LSTM TAMPERING VALIDATION 8 NEURON CHAIN

Stage V(t)	RNN Δ=0.0	RNN Δ=1.0	LSTM Δ=0.0	LSTM Δ=1.0	Neurons x_L,z_M,y_N
V(1)	9.19E-31	1.19E-02	6.81E-31	3.96E-05	08-04-04
V(2)	7.58E-31	2.68E-04	9.99E-31	3.88E-05	20-04-04
V(3)	6.90E-31	2.28E-05	8.02E-31	3.79E-05	32-04-04
V(4)	6.34E-31	3.62E-06	7.95E-31	3.63E-05	44-04-04
V(5)	6.09E-31	9.25E-07	9.98E-31	3.34E-05	56-04-04

The effects of tampering the Neural Block Chain (Fig. A) is detected by the learning algorithm even when the tampered values only differ in one neuron,Δ=1.0. The LSTM network, with a Back Propagation Through Time learning algorithm, keeps the error constant as the cell gates prevent its circulation through the network.

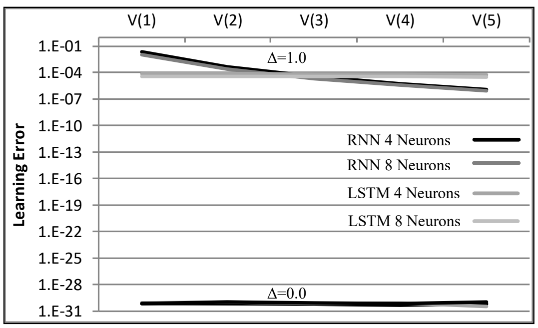

Fig. 12. RNN and LSTM Tampering Validation.

Both Neural Network models, RNN and LSTM, show similar properties: they manage almost insignificant learning errors with very reduced Learning Thresholds T_L = 1.00E-30 due to their iterative gradient descent learning algorithms. Their learning iterations decrease with the number of input neurons, or user data, although there is convergence as the validation stage increases. Due the mining process is based on random values, the results are not as linear as originally expected.

VII. CONCLUSIONS

This paper has proposed "Neuron as a Service" (NaaS) based on the BlockChain Neural Network that gradually increments neurons as user information or data increases. The additional neurons codify both the new information to be added to the "neural block" and previous neurons potential to form the "neural chain". This configuration provides the Neural Network with the same properties found in the BlockChain: security and decentralization with the same validation process: mining the input neurons until the neural network solution is found.

The mathematical model of the Neural Networks in BlockChain configuration has a stable learning convergence during its learning process although mining is not as linear as originally expected. Mining the Blockchain Neural network is easier when more user data is stored in the model. The main advantage of this research proposal is the biological simplicity of the solution, however it suffers high computational cost when the number of chain neurons increase. In addition, the BlockChain neural network successfully detects tampering without difficult or effort.

Both BlockChain Neural Networks, RNN and LSTM, perform similarly. The Random Neural Network requires less learning interactions dues its main analytical properties based on the "product form" and the existence of the unique network steady state solution. The LSTM Network with a Back Propagation Through Time learning algorithm keeps the error constant as the cell gates prevent its circulation through the network. Further research work will increase the validation stages, therefore expanding the number of input neurons or user data. In addition, the contribution of the number of hidden neurons, or "neural chain", to the mining process will be further assessed.

REFERENCES

[1] S Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System - Bitcoin.org. (2008), 1-9.
[2] A. Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou. Hawk: The BlockChain Model of Cryptography and Privacy-Preserving Smart Contracts. IEEE Symposium on Security and Privacy. (2016), 839-858.
[3] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu and J. Kishigami. BlockChain contract: Securing a blockchain applied to smart contracts. International Conference on Consumer Electronics. (2016), 467-468.
[4] W. Serrano. The Random Neural Network with a BlockChain Configuration in Digital Documentation. In T. Czachórski, E. Gelenbe, K. Grochla, R. Lent, 32nd International Symposium, ISCIS 2018, Poznan, Poland, September 20-21, 2018, Proceedings, CCIS 935, Springer, 2018.
[5] K. Biswas and V. Muthukkumarasamy. Securing Smart Cities Using BlockChain Technology. International Conference High Performance Computing and Communications / Smart City / Data Science and Systems. (2016), 1392-1393.
[6] W. Serrano. 5G Cybersecurity Based on the BlockChain Random Neural Network in Intelligent Buildings. UK Workshop on Computational Intelligence. (2019), 409-422.
[7] S. Huh, S. Cho and S. Kim. Managing IoT devices using BlockChain platform. International Conference on Advanced Communication Technology. (2017), 464-467.
[8] W. Serrano. The BlockChain Random Neural Network in Cybersecurity and the Internet of Things. IFIP International Conference on Artificial Intelligence Applications and Innovations. (2019), 50-63.
[9] D. Hubel, C. Stevens, E. Kandel. The brain. Scientific American.Vol. 241, No. 3, (1979), 1-146.
[10] D. Smith, E. Bullmore. Small-World Brain Networks. The Neuroscientist : a review journal bringing neurobiology, neurology and psychiatry. 12, (2007), 512-23
[11] P. Goldman-Rakic. Cellular Basis of Working Memory. Neuron. Vol. 14, (1995), 477-485
[12] B. Milner, L. Squire, E. Kande. Cognitive Neuroscience and the Study of Memory. Neuron, Vol. 20, (1998), 445–468.
[13] D. Zipser, B. Kehoe, G. Littlewort, J. Fuster. A Spiking Network Model of Short-Term Active Memory. The Journal of Neuroscience. Vol. 13, Issue 8, (1993), 3406-3420.
[14] F. H. C. Crick. Thinking about the brain. Scientific American, Vol. 241, No. 3 (1979), 219-233.
[15] J. Baron. Thinking and Deciding Cambridge University Press. (1988), 1-563.
[16] V. Goel. Cognitive Neuroscience of Thinking. In Handbook of Neuroscience for the Behavioral Sciences, Eds. G. G. Berntson and J. T. Cacioppo. New York: John Wiley and Sons. (2007), 1-46.
[17] M. Swan. BlockChain Thinking : The Brain as a Decentralized Autonomous Corporation. IEEE Technology and Society Magazine. Volume: 34 , Issue: 4, (2015), 41-52.

[18] D. Pointcheval. Neural Networks and their Cryptographic Applications. Livre des resumes Eurocode Institute for Research in Computer Science and Automation. (1994), 1-7.

[19] W. Kinzel and I. Kanter. Interacting Neural Networks and Cryptography Secure exchange of information by synchronization of neural networks. Advances in Solid State Physics. 42, (2002), 383-391.

[20] A. Klimov, A. Mityagin and A. Shamir. Analysis of Neural Cryptography. International Conference on the Theory and Application of Cryptology and Information Security. 2501, (2002), 288-298.

[21] E. Volna, M. Kotyrba, V. Kocian and M. Janosek. Cryptography based on the Neural Network. European Conference on Modelling and Simulation. (2012), 1-6.

[22] A. Yayık and Y. Kutlu. Neural Network based cryptography. International Journal on Neural and Mass - Parallel Computing and Information Systems. 24, 2, (2014), 177-192.

[23] E. Gelenbe. Random Neural Networks with Negative and Positive Signals and Product Form Solution. Neural Computation, 1. (1989), 502-510.

[24] E. Gelenbe. G-Networks with Triggered Customer Movement. Journal of Applied Probability, 30. (1993), 742-748.

[25] E. Gelenbe: Stability of the Random Neural Network Model. Neural Computation. 2,2, 239-247 (1990)

[26] E. Gelenbe. Learning in the Recurrent Random Neural Network. Neural Computation, 5. (1993), 154-164.

[27] S Hochreiter, J Schmidhuber - Long short-term memory. Neural computation, 9 (1997), 1735-1780.

APPENDIX