Coal City University Journal of Science

VOLUME 2 ISSUE 1 JULY 2022

ISSN: 2734-3758(Print), 2734-3766 (Online)

A publication of faculty of Natural and Applied Sciences, Coal City University, Enugu Nigeria.

CCU Journal of Science
Vol. 02, Issue 01, July, 2022

ISSN: 2734-3758(Print), 2734-3766 (Online)
https://ccujos.com

Neural Networks versus Conventional Computers

Arinze Steve Nwaeze

Dean, Faculty of Environmental Sciences; Head of Department, Computer Science
Caritas University, Amorji-Nike, Enugu, Nigeria

Corresponding Author: **Arinze Steve Nwaeze** (arinze.nwaeze@caritasuni.edu.ng)

Abstract

The first artificial neuron was produced in 1943 by the neurophysiologist *Warren McCulloch* and the logician *Walter Pits*. But the technology available at that time did not allow them to do much. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. Conventional computers use an algorithmic approach i.e. *the computer follows a set of instructions in order to solve a problem*. Unless the specific steps that the computer needs to follow are known the computer cannot solve the problem. Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks that are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (*normally a conventional computer is used to supervise the neural network*) in order to perform at maximum efficiency.

Keywords: Neural Network, Artificial Intelligence, Machine learning, Neurons, Perturbation, Supervised training, Synapse

Introduction

There are many emerging technologies that are now changing the word for better (Onyema et al, 2020) and assisting to improve different systems. One of such technologies is the neural network and machine learning algorithms (Brown et al, 2004, Onyema et al, 2022). A neural network is man's crude way of trying to simulate the brain electronically. So, to understand how

a neural net works we first must have a look at how the old grey matter (the brain) does its business. Much is still unknown about how the brain trains itself to process information, so theories abound. Our brains are made up of about 100 billion tiny units called *neurons*. In the human brain, a typical neuron collects signals from others through a host of fine structures called *dendrites (*Martini and Frederic, 2005).. The neuron sends out spikes of electrical activity through a long, thin strand known as *axon*, which splits into thousands of branches. At the end of each branch, a structure called a *synapse* converts the activity from the axon into *electrical effects* that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

In other words, what the neuron does (this is over simplified I must add) is sum up the inputs to itself in some way and then, if the end result is greater than some threshold value set inside, the neuron fires up. It generates a voltage and outputs a signal along an *axon* (Brunk andTerman, 2002; Gerber, 2003) Just have a good look at the illustrations and try to picture what is happening within this simple little cell as shown in the figure 1 below:

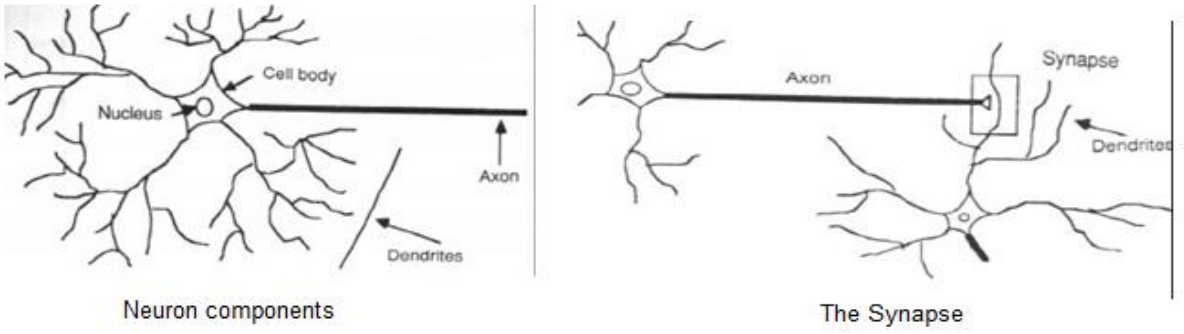

Figure 1: Neuron Component and Synapse Source: Culled from: *https://www.istockphoto.com/search/more-like-this/1266728575?assettype=image&mediatype=illustration&phrase=synapse%20drawing*

Artificial Neural Networks

Artificial Neural Networks simulate the above natural process shown in figure 1. Neural networks are made up of many artificial neurons (Chudler, 2021). An artificial neuron is simply an electronically modeled biological neuron (Giménez, 1998). How many neurons are used depends on the task on hand. It could be as few as three or as many as several thousand. There are many different ways of connecting artificial neurons together to create a neural network but I shall be concentrating on the most common which is called a *feed-forward network*. So, what does an artificial neuron look like? Well, here:

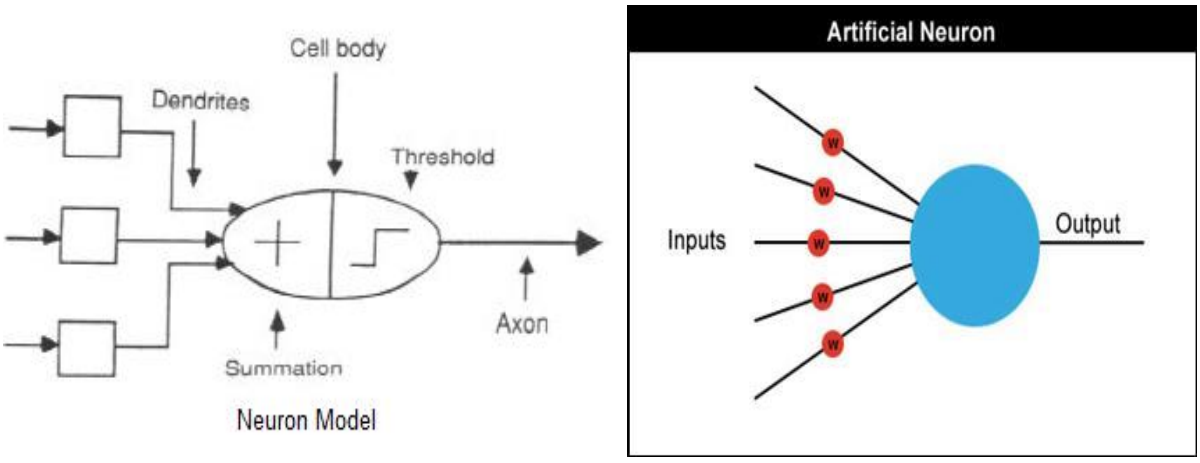

Figure 2: Neuron model and AI neuron

*Source: https://www.tutorialspoint.com/artificial_neural_network/artificial_neural*_network_basic_concepts.htm

Each input into the neuron has its own weight associated with it illustrated by the red circle. A weight is simply a floating point number and it is these we adjust when we eventually come to train the network. The weights in most neural nets can be both negative and positive, therefore providing excitatory or inhibitory influences to each input. As each input enters the nucleus (blue circle) it is multiplied by its weight (Davies, 2022). The nucleus then sums all these new input values which gives us the activation (again a floating point number which can be negative or positive). If the activation is greater than a threshold value - let's use the number 1 as an example - the neuron outputs a signal. If the activation is less than, 1 the neuron outputs zero (.

An artificial neuron is a device with many inputs and one output (Herrero et al, 1993; Herrup and Yang, 2007). The neuron has two modes of operation; the *training mode* and the *using mode*. In the training mode, the neuron can be trained to fire (or not), for particular input patterns as can be seen in figure 3. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

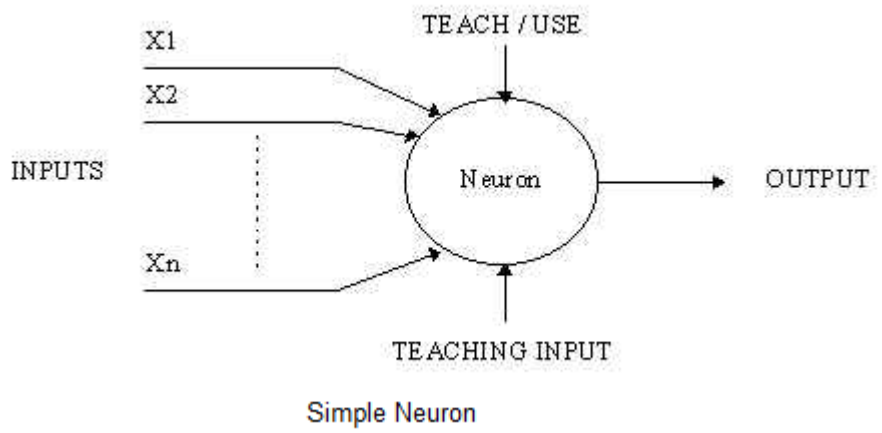

Figure 3: Simple neuron

Now let's do a little Math

Now we'll have to introduce some equations. We'll try to keep the math down to an absolute minimum but it will be useful for you to learn some notations. We'll feed in the math little by little and introduce new concepts when we get to the relevant sections. This way our mind can absorb all the ideas a little more comfortably and you'll be able to see how the math is put to work at each stage in the development of a neural net.

A neuron can have any number of inputs from *1* to *n*, where *n* is the total number of inputs. The inputs may be represented therefore as:

x1, x2, x3… x_n.

And the corresponding weights for the inputs as:

w1, w2, w3… w_n.

This is known as the *McCulloch and Pitts model* (MCP). Now, the summation of the weights multiplied by the inputs we talked about above can be written as: x1w1 + x2w2 + x3w3 …. + x_nw_n, which I hope you remember is the activation value. So;

a = x1w1+x2w2+x3w3... + x_nw_n

Fortunately there is a quick way of writing this down which uses the Greek capital letter sigma S, which is the symbol used by mathematicians to represent summation:

$$a = \sum_{i=0}^{i=n} w_i x_i$$

Now remember that if the activation > threshold we output a 1 and if activation < threshold we output a 0. Let's illustrate everything we've discussed so far with a diagram:

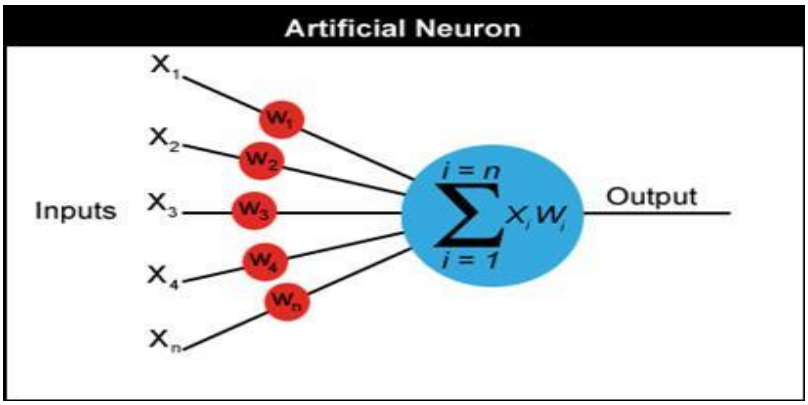

Figure 4: An MCP model

How do you actually use an artificial neuron?

Well, we have to link several of these neurons up in some way. One way of doing this is by organizing the neurons into a design called a *feed-forward* network. It gets its name from the way the neurons in each layer feed their output forward to the next layer until we get the final output from the neural network. This is what a very simple *feed-forward* network looks like:

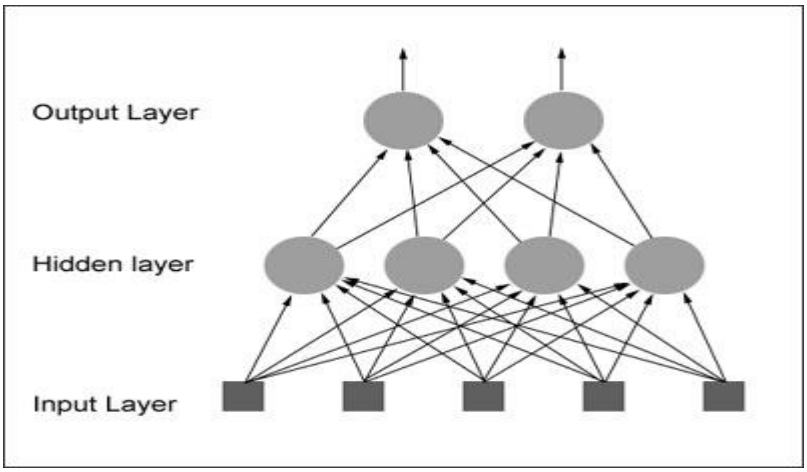

Figure 5: Operations done by neurons: Source: https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_basic_concepts.htm

Each input is sent to every neuron in the hidden layer and then each hidden layer's neuron's output is connected to every neuron in the next layer. There can be any number of hidden layers within a *feed-forward network* but one is usually enough for most problems you will tackle. The number of neurons chosen for the above diagram was completely arbitrary. There can be any number of neurons in each layer; it all depends on the problem. By now you may be feeling a little dazed by all this information. Well, let's see if we can get your very own brain's neurons firing; let's get a real world example of how a neural net can be used.

You probably know already that a popular use for neural nets is character recognition. So let's design a neural network that will detect the number '4'. Given a panel made up of a grid of lights which can be either *on* or *off*, we want our neural net to let us know whenever it thinks it sees the character '4'. The panel is eight cells square and looks like this:

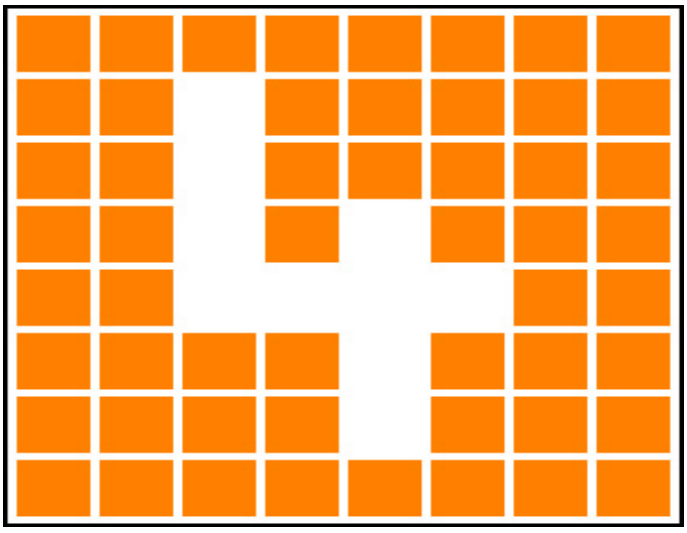

Figure 6: Neural net

We would like to design a neural net that will accept the state of the panel as an input and will output either a 1 or zero. A '*1*' will indicate that it thinks the character '4' is being displayed and *0* if it thinks it's not being displayed. Therefore the neural net will have 64 inputs, each one representing a particular cell in the panel and a hidden layer consisting of a number of neurons all feeding their output into just one neuron in the output layer. I hope you can picture this in your head because the thought of me drawing all those little circles and lines for you is not a happy one.

Once the neural network has been created it needs to be trained (Harris-Warrick, 2011). One way of doing this is to initialize the neural net with random weights and then feed it a series of inputs which represent, in this example, the different panel configurations. For each configuration we check to see what its output is and adjust the weights accordingly so that whenever it sees something looking like a number 4 it outputs a *1* and for everything else it outputs a zero. This type of training is called *supervised learning* and the data we feed it is

called a *training set*. In *BackProp* networks, learning occurs during a training phase in which each input pattern in a **training set** is applied to the input units and then propagated forward. The pattern of activation arriving at the output layer is then compared with the correct (associated) output pattern to calculate an error signal. The error signal for each such target output pattern is then backpropagated from the outputs to the inputs in order to appropriately adjust the weights in each layer of the network.

Training the Artificial Neural Network (ANN)

We can teach a three-layer artificial neural network to perform a particular task by using the following procedure:

1. We present the network with training examples, which consist of a pattern of activities for the input units together with the desired pattern of activities for the output units.
2. We determine how closely the actual output of the network matches the desired output.
3. We change the weight of each connection so that the network produces a better approximation of the desired output.

To illustrate the BackProp teaching procedure:

In order to train a neural network to perform some task, we must adjust the weights of each unit in such a way that the error between the desired output and the actual output is reduced (Ivannikovand Macleod, 2013). This process requires that the neural network computes the error derivative of the weights (**EW**). In other words, it must calculate how the error changes as each weight is increased or decreased slightly.

Assume that we want a network to recognize hand-written digits. We might use an array of, say, 256 sensors, each recording the presence or absence of ink in a small area of a single digit. The network would therefore need 256 input units (one for each sensor), 10 output units (one for each kind of digit) and a number of hidden units. For each kind of digit recorded by the sensors, the network should produce high activity in the appropriate output unit and low activity in the other output units.

To train the network, we present an image of a digit and compare the actual activity of the 10 output units with the desired activity. We then calculate the error, which is defined as the square of the difference between the actual and the desired activities. Next, we change the weight of each connection so as to reduce the error. We repeat this training process for many different images of each different images of each kind of digit until the network classifies every image correctly.

To implement this procedure we need to calculate the *error derivative* for the weight (EW) in order to change the weight by an amount that is proportional to the rate at which the error

changes as the weight is changed. One way to calculate the EW is to perturb (adjust) a weight slightly and observe how the error changes. But that method is inefficient because it requires a separate perturbation for each of the many weights.

After a *BackProp* network has learned the correct classification for a set of inputs, it can be tested on a second set of inputs to see how well it classifies untrained patterns. Thus, an important consideration in applying *BackProp* learning is how well the network generalizes.

In many real world situations, we are faced with incomplete or noisy data, and it is important to be able to make reasonable predictions about what is missing from the information available. This can be an especially difficult task when there isn't a good theory available to help reconstruct the missing data. It is in such situations that Backpropagation (BackProp) networks may provide some answers.

BackProp

A *BackProp* network consists of at least three layers of units: an **input** layer, at least one intermediate **hidden** layer, and an **output** layer. Connection weights in a *BackProp* network are one-way. Typically, units are connected in a *feed-forward* fashion with input units fully connected to units in the hidden layer and hidden units fully connected to units in the output layer. When a BackProp network is cycled, an input pattern is propagated forward to the output units through the intervening input-to-hidden and hidden-to-output weights.

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.
- The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

If you think about it, you could increase the outputs of this neural net to 10. This way the network can be trained to recognize all the digits 0 through to 9. Increase them further and it could be trained to recognize the alphabet too.

Applications of neural networks

Neural networks have broad applicability to real world business problems. In fact, they have already been successfully applied in many industries. Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

- sales forecasting
- industrial process control

- customer research
- data validation
- risk management
- target marketing

Neural networks in medicine

Artificial Neural Networks (ANN) is currently a 'hot' research area in medicine and it is believed that they are already receiving extensive application to biomedical systems (Zao et al, (2017; Macpherson, 2002). The research is mostly on modeling parts of the human body and recognizing diseases from various scans (e.g. cardiograms, CAT scans, ultrasonic scans, etc.).

Neural networks are ideal in recognizing diseases using scans since there is no need to provide a specific algorithm on how to identify the disease (Wilson et al, 2012). Neural networks learn by example so the details of how to recognize the disease are not needed. What is needed is a set of examples that are representative of all the variations of the disease. The quality of examples is not as important as the 'quantity'. The examples need to be selected very carefully if the system is to perform reliably and efficiently. This paper presented various potentials of neural networks which are in tandem with many other studies (Onyema et al, 2020; Wilson et al, 2012) that highlighted the growing influence of technology and computing in particular in all aspects of human endevours including health and education.

Conclusion

The computing world has a lot to gain from neural networks. Their ability to learn by example makes them very flexible and powerful. Furthermore there is no need to devise an algorithm in order to perform a specific task; i.e. there is no need to understand the internal mechanisms of that task. They are also very well suited for real time systems because of their fast response and computational times which are due to their parallel architecture.

Neural networks also contribute to other areas of research such as neurology and psychology. They are regularly used to model parts of living organisms and to investigate the internal mechanisms of the brain (Drachman, 2004; Lee et al, 2006). Finally, even though neural networks have a huge potential we will only get the best of them when they are integrated with computing, AI, fuzzy logic and related subjects.

REFERENCES

Al, Martini, Frederic Et (2005). Anatomy and Physiology' 2007 Ed.2007 Edition. Rex Bookstore, Inc. p. 288. ISBN 978-971-23-4807-5.

Brown EN, Kass RE, Mitra PP (May 2004). "Multiple neural spike train data analysis: state-of-the-art and future challenges". Nature Neuroscience. 7 (5): 456–61. doi:10.1038/nn1228. PMID 15114358. S2CID 562815.

Brunk UT, Terman A (September 2002). "Lipofuscin: mechanisms of age-related accumulation and influence on cell function". Free Radical Biology & Medicine. 33 (5): 611–9. doi:10.1016/s0891-5849(02)00959-0. PMID 12208347.

Chudler EH. "Brain Facts and Figures". Neuroscience for Kids. Retrieved 2009-06-20. "16.7: Nervous System". Biology LibreTexts. 2021-01-14. Retrieved 2022-02-28.

Chudler EH. "Milestones in Neuroscience Research". Neuroscience for Kids. Retrieved 2009-06-20.

Davies, Melissa (2002-04-09). "The Neuron: size comparison". Neuroscience: A journey through the brain. Retrieved 2009-06-20.

Drachman DA (June 2005). "Do we have brain to spare?". Neurology. 64 (12): 2004–5. doi:10.1212/01.WNL.0000166914.38327.BB. PMID 15985565. S2CID 38482114.

Gerber U (January 2003). "Metabotropic glutamate receptors in vertebrate retina". Documenta Ophthalmologica. Advances in Ophthalmology. 106 (1): 83–7. doi:10.1023/A:1022477203420. PMID 12675489. S2CID 22296630.

Giménez, C. (February 1998). "[Composition and structure of the neuronal membrane: molecular basis of its physiology and pathology]". Revista de Neurologia. 26 (150): 232–239. ISSN 0210-0010. PMID 9563093.

Herrero MT, Hirsch EC, Kastner A, Luquin MR, Javoy-Agid F, Gonzalo LM, Obeso JA, Agid Y (1993). "Neuromelanin accumulation with age in catecholaminergic neurons from Macaca fascicularis brainstem". Developmental Neuroscience. 15 (1): 37–48. doi:10.1159/000111315. PMID 7505739.

Herrup K, Yang Y (May 2007). "Cell cycle regulation in the postmitotic neuron: oxymoron or new biology?". Nature Reviews. Neuroscience. 8 (5): 368–78. doi:10.1038/nrn2124. PMID 17453017. S2CID 12908713.

Harris-Warrick, RM (October 2011). "Neuromodulation and flexibility in Central Pattern Generator networks". Current Opinion in Neurobiology. 21 (5): 685–92. doi:10.1016/j.conb.2011.05.011. PMC 3171584. PMID 21646013.

Ivannikov MV, Macleod GT (June 2013). "Mitochondrial free Ca^{2+} levels and their effects on energy metabolism in Drosophila motor nerve terminals". Biophysical Journal. 104 (11): 2353–61. Bibcode:2013BpJ...104.2353I. doi:10.1016/j.bpj.2013.03.064. PMC 3672877. PMID 23746507.

Lee WC, Huang H, Feng G, Sanes JR, Brown EN, So PT, Nedivi E (February 2006). "Dynamic remodeling of dendritic arbors in GABAergic interneurons of adult visual cortex". PLOS Biology. 4 (2): e29. doi:10.1371/journal.pbio.0040029. PMC 1318477. PMID 16366735.

Macpherson, Gordon (2002). Black's Medical Dictionary (40 ed.). Lanham, MD: Scarecrow Press. pp. 431–434. ISBN 0810849844

Onyema, EM; Almuzaini,KK; Onu,FU;Verma,D; Gregory,US; Puttaramaiah,M and Afriyie,RK. (2022). Prospects and Challenges of Using Machine Learning for Academic Forecasting. Hindawi Computational Intelligence and Neuroscience, Volume 2022, Article ID 5624475, 1-7. https://doi.org/10.1155/2022/5624475

Onyema, E.M; Eucheria, N.C; Ezeanya, C.U; Eziokwu, P.N; Ani, U.E. (2020). Impact of E-learning Platforms on Students' Interest and Academic Achievement in data Structure Course. *Coal City University Journal of Science*, 1(1), 1-16. ISSN: 2734-3758(Print), 2734-3766 (Online).

Onyema, E.M; A. Sharma., C.E, Nwafor., A.G. Fyneface., S, Sen., E.C. Edeh (2020b). Impact of Emerging Technologies on the Job Performance of Educators in Selected Tertiary Institutions in Nigeria. *The Journal of Computer Science and its Applications, 27 (1), 52-62.* https://dx.doi.org/10.4314/jcsia.v27i1.4

Purves D, Augustine GJ, Fitzpatrick D, et al., editors. Sunderland (MA): Sinauer Associates; 2001.

"Researchers discover new type of cellular communication in the brain". The Scripps Research Institute. Retrieved 12 February 2022.

Wilson NR, Runyan CA, Wang FL, Sur M (August 2012). "Division and subtraction by distinct cortical inhibitory networks in vivo". Nature. 488 (7411): 343–8. Bibcode:2012Natur.488..343W. doi:10.1038/nature11347. hdl:1721.1/92709. PMC 3653570. PMID 22878717.

Zhao B, Meka DP, Scharrenberg R, König T, Schwanke B, Kobler O, Windhorst S, Kreutz MR, Mikhaylova M, Calderon de Anda F (August 2017). "Microtubules Modulate F-actin Dynamics during Neuronal Polarization". Scientific Reports. 7 (1): 9583. Bibcode:2017NatSR...7.9583Z. doi:10.1038/s41598-017-09832-8. PMC 5575062. PMID 28851982.