ARTIFICIAL NEURAL NETWORK MODEL: A TOOL FOR INTELLIGENT CONTROL SYSTEM

by

Eze Ukamaka Josephine[1], Bakare Kazeem[2], Idigo-Ibenta Florence[3]
Madonna University Nigeria[1], Oska.Jo and Partners Ltd[2], Government of Anambra State, Nigeri[3].
ukamakajoe@gmail.com[1], bakarekazeem@gmail.com[2], oxide141@gmail.com [3],

ABSTRACT

This research focused on the application of Artificial Neural Networks (ANN) for controlling the speed of a DC motor. It delved into the intricacies of ANN, encompassing the fundamental aspects of learning, training, and model development within this neural network paradigm. To achieve the goals of this research, an Artificial Neural Network (ANN) model controller for a DC motor alongside a conventional PID controller were developed. Simulations of both models were conducted and their results were compared to assess potential improvements. The simulations clearly demonstrated that the ANN controller exhibits greater stability compared to the PID controller. In terms of performance, when comparing the percentage error of the speed control of the DC motor, the PID controller had a deviation of 1.46%, whereas the ANN model displayed a significantly improved deviation of only 0.24%. In conclusion, the percentage improvement established from comparing the results of the conventional PID and the Proposed ANN model demonstrated that the ANN model outperformed the existing PID model, offering superior results.

Keywords: Artificial Intelligent, Neural Network, Proportional Integral Derivative, Simulation, Network Topologies

I. INTRODUCTION

Neural network research has spread through every field of science especially in area of control systems, speech recognition, medicine, synthesis, image processing and robotics (Esuo, 2017). Neural networks are seen to be a set of algorithms and models that imitate the human brain. The goal of artificial intelligence is to make computers behave like people, in comparison to humans, artificial intelligence attempts to tackle complicated issues in a more humane manner. Wajeeha et al., (2022). The ability of artificial entities to solve complicated problems is known as artificial intelligence (AI) (Mano, 2014). Neural networks are designed and trained to effectively recognize patterns. The patterns used by neural networks are numerical in nature. This means that real world data like sound, text, images etc must be translated to numerical form. They interpret data through different kinds of machine perception, labeling or clustering raw input data. It uses training as a learning tool. Neural networks are used in large machine learning applications, they involve algorithms with regression, reinforcement learning and classification. According to Casidy (2019), neural network a system that can compute, which consist of a number of simple, highly multiple-connected processing elements or data. ANN can process information by their dynamic state response to external inputs source. The ANN controller inputs are the accelerations of the sprung and unsprung masses, and the output is the valve opening area. (Anis Hamza el ta 2023)

Topology of a neural network refers to the way the Neurons are connected, and it is an important factor in network functioning and learning. A common topology in unsupervised learning is a direct mapping of inputs to a collection of units that represents categories. Neural networks are modelled using biological processes, for information processing. The information for processing in neural network are the nervous system, the main units of the nervous system are the neuron, the signals are propagated from the cell which form the potential differences between the inside and outside of cells. (Lewis et al., 2019).

II. MATERIALS AND METHODS

2.1 Software Materials

Controllers which are Artificial Neural Network (ANN) Controller, Proportional-Integral-Derivative (PID) Controller, Simulink. (for modeling, training, simulation, and performance analysis).

2.2 Methods

Differential equations representing electrical and mechanical dynamics were used to model DC Motors. For PID Controller design, a conventional PID controller was tuned using the Ziegler-Nichols method. For ANN Controller design, a feedforward neural network with one hidden layer, Input: Speed error; Output: Control signal (voltage); trained using the Levenberg–Marquardt backpropagation algorithm. Data for simulated speed reference and load variation, and data for training and testing the controllers. Simulation: Both controllers were simulated in MATLAB/Simulink. Step response tests and load variation scenarios were applied. Performance Evaluation: Key metrics evaluated: speed error, stability, rise time, and steady-state error. Comparative analysis was conducted to assess performance improvement of the ANN over PID.

2.2.1 Mathematical mode of Neuron

Figure 1: Components of a neuronal cell

The basic component of brain circuitry is a specialized cell called the neuron, which consists of the cell body made up of dendrites and axon the biological neuronal cell system has its components, the components are shown in Figure1 (Eduardo, 2022). The components are made up of the dendrites which gets signals from other neurons into the body cell or soma (Lewis and Ildirek 2019). Most often the dendrites multiply each input signal by a transfer weighting coefficient. In the soma, cell capacitance separates the signals which are collected in the axon. When the composite signal exceeds a cell threshold signal, the action is transmitted through the axon. From research it was noticed that cell nonlinearities make the composite action potential a nonlinear function of the combination of arriving signals. Axon which is another component connects through synapses with the dendrites of subsequent neurons and synapses operate through the discharge of neurotransmitter chemicals across intercellular gaps, also it can be either excitatory (tending to fire the next neuron) or inhibitory (tending to prevent firing of the next neuron) (Lewis and Ildirek 2019).

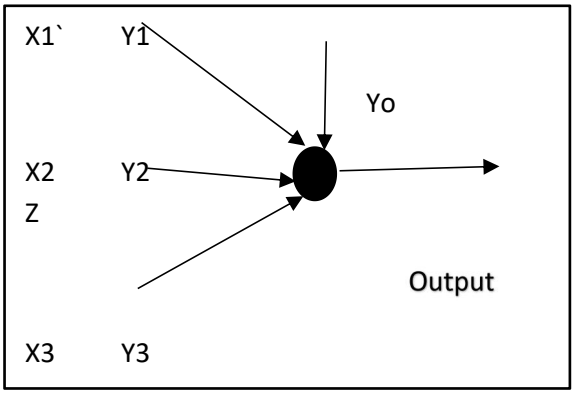


Figure 2: Mathematical Model Representing a Neuron

A Neuron can also be modelled mathematically and this can be shown in figure 2. The figure consists of the dendrite which weight is yj , the firing threshold yo which is also called the 'bias', the summing weight incoming signals, and the nonlinear function σ(·). The cell inputs are the n time signals x1(t), x2(t), . . . xn(t) and the output is the scalar y(t), which can expressed as:

$$Z(t) = \left(\sum_{i=1}^{n} yj\, xj\,(t) + Yo \right) \text{ ----- (1)}$$

Equation (1) shows the positive weights *yj* correspond to excitatory synapses and

negative weights to inhibitory synapses (Lewis. and. Ildirek 2019).

To gain a comprehensive grasp of the mathematical intricacies achievable through the interconnection of individual artificial neurons, it is advisable to steer clear of random interconnections. Random interconnections can result in an excessively complex system that becomes unmanageable. Early researchers have devised standardized topologies for artificial neural networks. These predefined topologies facilitate more straightforward, quicker, and efficient problem-solving. Various artificial neural network topologies are tailored for solving specific problem types. Once the problem type is identified, the choice of the artificial neural network's topology becomes crucial for the system.

The topology and its parameters need to be modified to solve the problem. Modifying topology of artificial neural network does not mean that we cannot use the artificial neural network, it is only a precondition. Before artificial neural network can be used, we need to teach it or train it on solving that type or particular problem. The artificial neural networks use the inputs they have just as biological neural networks to learn the behaviour/responses from their environment or surroundings.

Research has demonstrated that artificial neural networks utilize three primary learning methods: supervised learning, unsupervised learning, and reinforcement learning. The choice of learning method parallels the selection of the neural network's topology and depends on the nature of the problem at hand. Each learning method operates based on distinct principles, yet they share common elements, namely, learning data and learning rules. These two parameters play a critical role in determining the network's functions and associated costs. The fundamental objective of an artificial neural network is to generate appropriate output responses based on given input data. In summary, the process involves the selection of an optimal topology, potential modifications to it, and the subsequent choice of an appropriate learning method as prerequisites to effectively utilizing artificial neural networks for problem-solving Artificial neural networks has been in existence for many years and have been can find working in many areas such as process control, chemistry, gaming, radar systems, automotive industry, space industry, astronomy, genetics, banking, fraud detection, etc. and solving of problems like function approximation, regression analysis, time series prediction, classification, pattern recognition, decision making, data processing, filtering, clustering, etc (Kenji 2014).

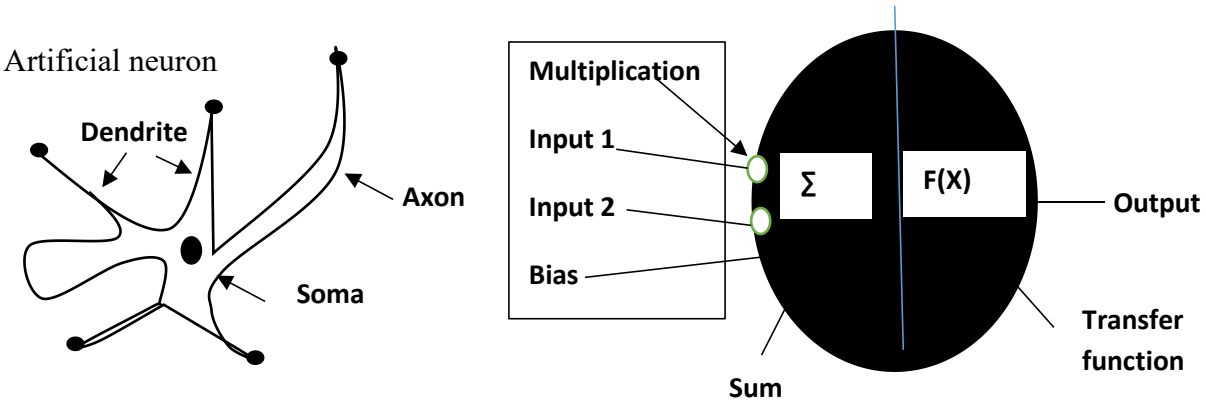


Figure 3: Artificial and Biological Neural

Figure 3 depicts the designs of a biological neuron and an artificial neuron. In the biological neuron, information enters through the dendrite, is processed within the soma, and

then transmitted through the axon. However, in the case of an artificial neuron, information enters the neuron's body through weighted inputs (each input is individually multiplied by a weight). The artificial neuron's body is responsible for summing the weighted inputs, incorporating a bias, and processing this sum using a transfer function. Finally, the artificial neuron conveys the processed information through its output(s). The important of artificial neuron model is the simplicity, which can be seen in its mathematical description (F.I. Lewis, S. J and A. Ildirek 2019).

Yi (k) F E wi(k) . xi (k) + b ------- (2)

Where: Xi (k) is input value in discrete time K and t goes from 0 to m; Wi is weight value in discrete time k where t goes from 0 to m, b is bias, F is a transfer function and Yi(k) is output value in discrete time k. the model of an artificial neuron and its equation has unknown variable, the major unknown variable of the model is its transfer function. The transfer function describes the properties of artificial neuron and it can be a mathematical function. We choose it on the basis of problem that artificial neuron (artificial neural network) needs to solve and in most cases we choose it from the following set of functions: Step function, Linear function and Non-linear (Sigmoid) function. Step function normally should be a binary function that has only two output values, which must be zero and one. That means if input value meets specific threshold the output value results in one value and if specific threshold is not meet that results in different output value. The threshold can be described using equation (3).

$$\begin{cases} 1 \text{ if } f \text{ wi } X_i > \text{threshold} \\ 0 \text{ if wi } X_t < \text{thresholds} \end{cases} \quad \text{-------- (3)}$$

Equation 3 illustrates the artificial neuron perceptron and the type of transfer function employed within artificial neurons. Perceptron are primarily applied in solving classification problems and are typically located in the output layer of artificial neural networks. Regarding the role of the linear transfer function within an artificial neuron, it serves to transform the sum of weighted inputs and bias. Such artificial neurons, in contrast to perceptions, are frequently situated in the input layer of artificial neural networks. The most prevalent nonlinear function used in this context is the sigmoid function.

An artificial neural network is produced when two or more artificial neuron are combined. One artificial neuron is not enough or effective in solving real-life problems, but combination of may artificial neuron which forms artificial neural networks have the capacity to solving real life time problems. The artificial neural networks have the ability of solving complicated problems. They solve problems making use of the building blocks which they use in processing information or data in a non-linear, distributed, parallel and local manner.

2.2.2 Neural Network Learning

The learning process in a neural network involves six key stages, each contributing to a researcher's understanding of how it operates. These stages can be summarized as: Initialization - In this stage, all neurons are assigned initial weights. Forward Propagation - Here training set inputs are passed through the neural network, and the output is computed. Error Calculation - Since we work with a known training set, the correct output is available. Therefore, an error function can be defined by measuring the difference between the model's output and the actual result. Back Propagation - The goal in this stage is to minimize the error function with respect to the neurons. Weight Update: The optimal weights are adjusted to affect the back propagation and improve the results, often using algorithms like Levenberg Marquardt. Iterate Until Convergence - The artificial neural network (ANN) is trained here, through repeated iterations until it accurately and adequately learns the underlying rule, enabling it to

replicate this knowledge when required. Back propagation networks acquire the ability to classify and generalize patterns. (Mano, 2014) The connections between the artificial neurons change as a pattern appears until they provide the right response. It is important to note that minimizing the error function is equivalent to optimizing convergence. These stages help illustrate the general process of training in a neural network.

Table 1: Stages and Learning Process in Neural Network

S/N	Stages	Description
1	Initialization	Neurons were identified with initial weights at this stage
2	Forward Propagation	Data from the input are processed here which will eventually computed and captured at the output.
3	Error Calculation	The error is calculated at this point by computing the difference between the output and the desired output model
4	Back Propagation	Minimization of error is done here by adjusting the weights
5	Weight Update	Weights update is done to improve convergence. Levenberg-Marquardt can be used to do weight update
6	Iterate Until Convergence	Learning is done here, ANN learns the rule accurately when the process is repeated iteratively

Table 1 summarizes the learning process. Table 2 demonstrates the process where the input is twice the desired output

Table 2: LM Propagation Applied in the Forward Direction and Initialization

S/N	**Input**	**Desired Output**
1	0	0
2	2	4
3	3	6
4	4	8
5	5	10

Let the weight value be w = 3, then the mode output becomes

Table 3: Forward Training Showing the Error and w

S/N	Input	Desired Output	Model Output (w = 3)
1	0	0	0
2	2	4	6
3	3	6	9
4	4	8	12
5	5	10	15

Table 4: Increased Error and Further Increase of the Value of w

S/N	Input	Desired output	Model output w =3	Absolute error	Square error
1	0	0	0	0	0
2	2	4	6	2	4
3	3	6	9	4	8
4	4	8	12	3	6
5	5	10	15	1	1

Table 5: A Backward Training in LM with Decrease Value of w

S/N	Input	Desired output	Model output w =3	Absolute error	Square error
1	0	0	0	0	0
2	2	4	6	1	1
3	3	6	9	3	6
4	4	8	12	2	4
5	5	10	15	0	0

Increasing the value of w increases the error, thus we stop increasing the value of w further. Suppose the value of w is now decreased

Table 6: Reduction in the Error Value

Input	Desired output	Model output w =3	Absolute error	Square error	Model output (w=2)	Absolute error	Absolute error
0	0	0	0	0	0	0	0
2	2	4	1	1	2	0	0
2	4	6	4	4	4	0	0
3	6	9	6	6	6	0	

Tables (1 to 6) provide illustrate that both the absolute error and squared error decrease as the weight (w) decreases. The following steps were carried out in the previous illustration:
Weight Initialization: The weight (w) was initialized, and LM (Levenberg-Marquardt) propagation was applied in the forward direction. Initial Error: Some errors were noticed during the forward training. Weight Adjustment: To address the errors, the weight (w) was increased, and forward training was continued. Increased Error: Unfortunately, this led to an increase in the error.
Backward Training: Backward training using the LM algorithm was applied with a decrease in the value of w. Reduced Error: This resulted in a reduction in the error value. The primary objective was to find the optimal weight value that minimizes the error. Once this state is reached and convergence of the neurons is achieved, the training process is halted. It's worth noting that the Levenberg-Marquardt training algorithm incorporates elements of the Newton algorithm. Levenberg–Marquardt, (LM), optimization training method is normally used to carry out the training in ANN.

There are some other training algorithms such as the steepest descent algorithm, and the Gauss Newton algorithm. Even though the

EBP algorithm is still widely in use, but the algorithm has low convergence.

2.2.3 Levenberg–Marquardt Training

The LM algorithm is adjudged to be much faster than other algorithms, this is because the size of the multi– layer perceptron (MLP) is not very large,

$H \approx J^T J + \mu I$ ---------- (4)

Is always positive, combination coefficient and I is identity matrix
With this approximation in equation 2.1, the assurance that matrix H is always invertible is guaranteed. Square error (SSE) is usually defined to evaluate the training process. To calculate this quantity, for all training patterns and network outputs, the following relation is used:

$$E(x,w) = \tfrac{1}{2} \sum_{p=1}^{p} \sum_{m=1}^{m} e_{p,m}^2 \quad \text{---------- (5)}$$

Where
x is the required input
w is the highest vector
$e_{p,m}$ is the training error at output m, when applying pattern p and defined as:

$$e_{p,m} = d_{p,m} - O_{p,m} \quad \text{-------- (6)}$$

Table 7: Summary of the Updates for Various Algorithms

S/N	Algorithm	Rules used	Convergence status	Computation of the algorithm
1	EBP algorithm	$w_{k+1} = w_k - \alpha g_{ki}$	Stable, Slow	Gradient
2	Newton algorithm	$w_{k+1} = w_k - \alpha g_k$	Unstable /fast	Gradient/Hessian
3	Gauss-Newton alg.	$w_{k+1} = w_k - (J_k^T J_k)^{-1} J_k e_k$	Unstable, fast	Jacobian
4	LM algorithm	$w_{k+1} = w_k - (J_k^T J_k)^{-1} J_k e_k$	Stable, fast	Jacobian

Table 7.0 summarizes the performance of different algorithms with respect to their stability
To implement the Levenberg–Marquardt algorithm for training neural network, two problems need to be solved namely: organizing the training process iteratively for weight updating, and calculating the Jacobian matrix. These two problems are addressed shortly.

In this study, since parameters for training were extracted from results obtained from the optimization results, the major work will be cantered on how to organize the training process iteratively for weight updating. This is because the calculation of the Jacobian matrix was part of the optimization calculation.
If we take that: W_x is the current weight, w_{k+1} is the next weight, E_{k+1} is the current total error, while E_k is the last total error. From the algorithm, using Levenberg-Marquardt algorithm, the training process was designed following the algorithm: Initial weights to be generated randomly and evaluation of the total error (SSE).
Applying the equation $w_{k+1} = w_k (J_k^T J_k + \mu I)^{-1} J_k e_k$ ..... (7)

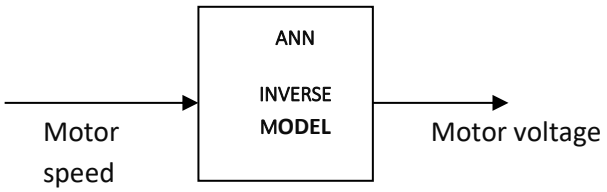
An update was carried out to update and adjust weights. Evaluation of the total error with addition of the new weights. Increasing the total current error due to the update requires thereafter, a repeat of step 2 is made for the update again. The Levenberg-Marquardt training algorithm gives a better result because it solves the problems existing in both the gradient descent (or EBP) method and Gauss – Newton method for neural network training.

2.2.4 Artificial Neural Network model controller

The focus here is using the application of artificial neural network (ANN) in controlling DC motor.

In the development of the control system, for instance using ANN inverse model controller of dc motor is shown in the block diagram of figure 4.

.



`

Figure 4: Block Diagram of ANN Inverse Model

The inverse model when given a particular motor speed generate at the output a corresponding voltage produces a speed. The inverse model as the name implies uses the inverse model of the dc motor. The dc motor takes voltage as input to give speed at the output but the inverse model takes speed as input and generates voltage at its output. In this case, speed at three consecutive instant of time for a particular trajectory are presented to the inputs while the inverse model produce at the output, the voltage that is required at the motor's inputs to generate the particular target reference speed.

2.2.5 The ANN inverse model structure

The ANN inverse model of the dc motor can be described as a three input single output structure with three speeds taken at three time instants of n, n+1, and n-1. These three speeds serve as the inputs while the motor terminal voltage serves as the output. The structure of the ANN inverse model of the dc motor is shown in figure 5.

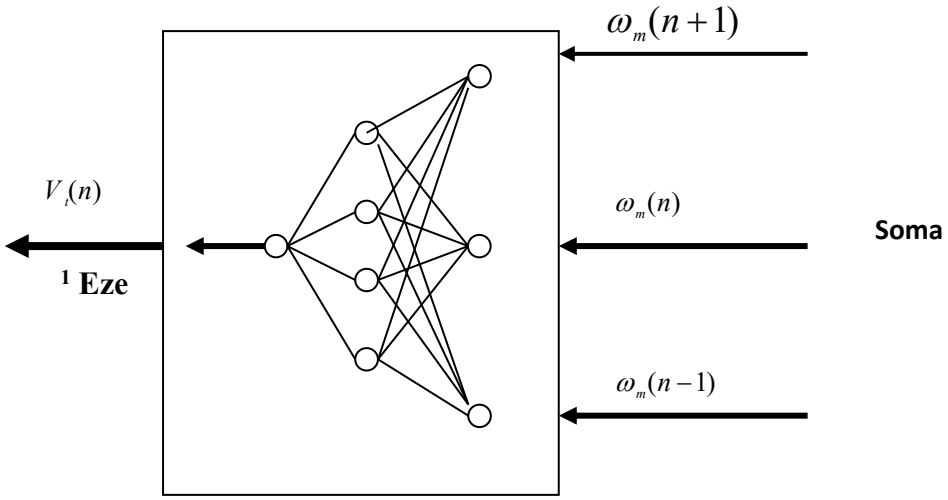


Figure 5: ANN Inverse Model Structure

The values of $\omega_m(n+1),\ \omega_m(n)\ and\ \omega_m(n-1),$

are seen and taken as the independent inputs of the ANN inverse model. It is pertinent to state here that the ANN inverse model can be trained to provide the terminal voltage for any DC motor with unknown parameters. The Artificial Neural Network Inverse Model (ANNIM) of the dc motor is achieved through simulation. Simulation is the imitation of the operation of real-world process or system over time. Simulation involves the generation of an artificial history of a system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented.

Simulation is an indispensable problem-solving methodology for the solution of many real-world problems. Simulation is used to describe and analyse the behaviour of a system, ask what if questions about the real system, and aid in the design of real systems. Both existing and conceptual systems can be modelled with simulation. The network data used in the training of this network is obtained from the physical reading taken from the developed dc servomotor system. The data is a two dimensional data. One is voltage and time while the other is speed and time. The network

parameters used in the neural network controller network coding are shown below:

2.2.6 Building Artificial Neural Network (ANN) Model

The following steps are used in the building of ANN model.

Step 1: At the input layer, the number of inputs=number of input neurons.

Step 2: At the output layer, the number of outputs=number of output neurons.

Step 3: At the hidden layer, the number of neurons and layers are not fixed and may take any number more than zero to map any complex functions.

Step 4: Assign weights

Step 5: Decide activation function. Here logistic function is taken for all neurons.

$$f(x) = \frac{e^x}{1+e^x} \quad \ldots\ldots (8)$$

Step 6: Select appropriate training pattern, that is, input-output pairs.

Inputs			Outputs
X_1	X_2	X_3	Y

Step 7: Training of ANN model; Here, the ANN output is calculated and compared with the desired output to determine the error E (desired output-actual output). Finally, minimize this error using some optimization technique. The sum-squared error may be written as

$$E = \sum_{i=1}^{n} (Y_i - V_i)^2 \quad \ldots\ldots\ldots\ldots\ldots (9)$$

The weights of the network have to be updated for the error to be minimized. This process is known as training of the neural network. In the training of the artificial neural network, the error is fed back to the network to update the weights. This will complete one cycle, the complete cycle is performed many times till the predicted error is reduced. The complete stage is called epoch.

Back propagation

Weight adjustment is done by the method of back propagation. The total prediction error E, is also a function of W.

$$\sum(\underline{W}) = \sum[Y_i - V_i(\underline{W})]^2 \quad \ldots\ldots\ldots (10)$$

2.2.7 Training Algorithm

The training algorithm has two process of information flow given, as back propagation and feed forward. Decide the network architecture (Hidden layers, neurons in each hidden layer)

Decide the learning parameter and momentum.

Initialize the network with random weights

Do till convergence criterion is met, for i=1 to # training data points. Feed forward the i-th observation through the net, Compute the prediction error on i-th observation, Back propagate the error and adjust weights, Next I, Check for convergence, End Do, When the global minima are reached, the network training has to stop. Practically, if the decrease in total prediction error since the last cycle is small or if the overall changes in the weights (since last cycle) are small. The training data is partitioned into training set and validation set. Training set to build the model, and validation set to test the performance of the model on unseen data.

2.2.8 Simulation Model

The Matlab Simulink was used in modelling and building of the model network that behaves exactly as the system. It was from this model that the input/output data pairs to train the model network are generated. Here, Artificial Neural Network inverse Model (ANN) of DC motor was used. For the speed control, Artificial Neural Network inverse model was used. Block diagram of figure 6 represents the system, and the designed model.

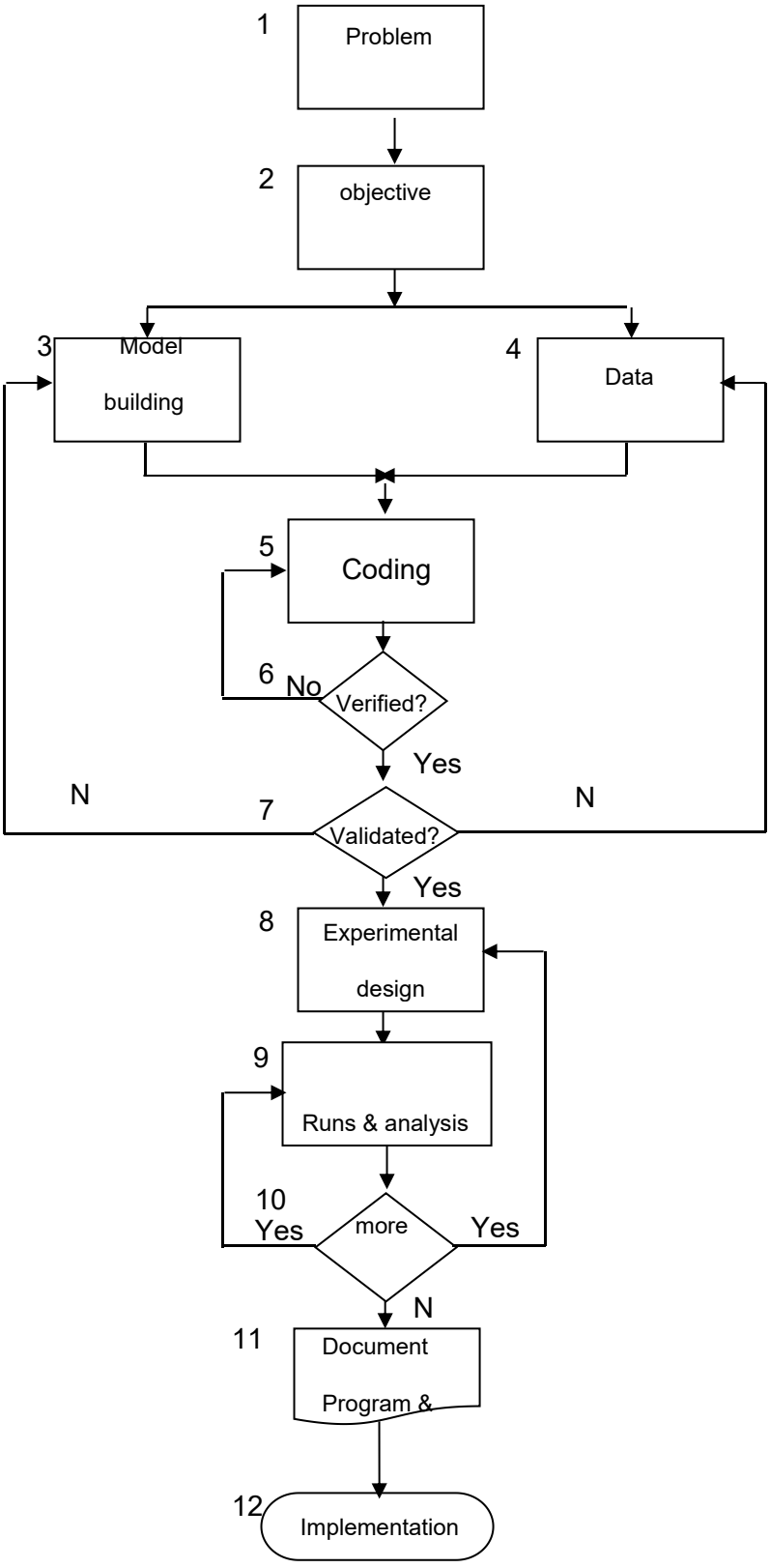


Figure 6: Steps in Simulation of ANNIM

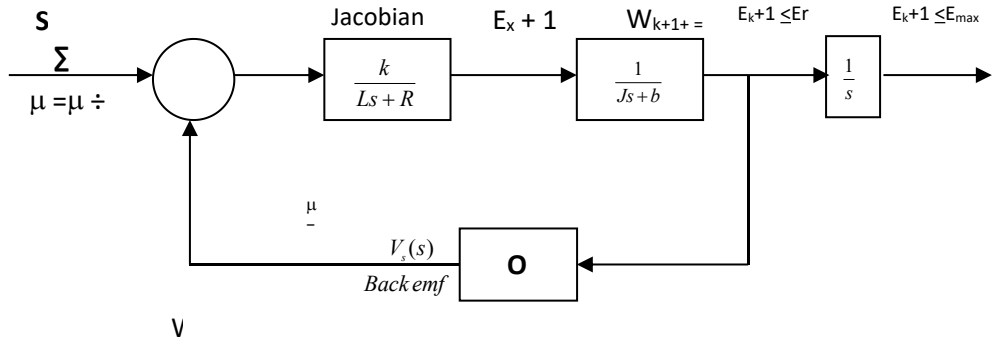


Figure 7: Block Diagram for Speed of DC Motor

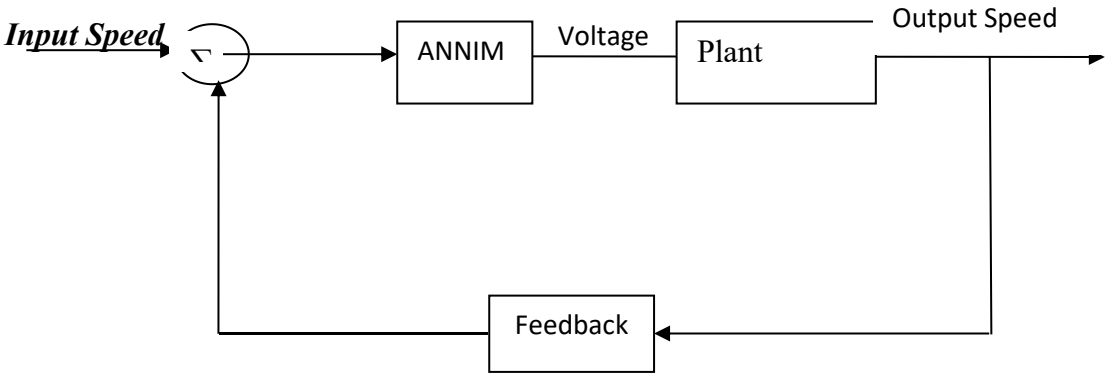


Figure 8: Block diagram for Simulink model

III. RESULTS

3.1 Speed Control Using ANN and PID Controllers in Simulink

The speed control using artificial neural network and proportional integral derivative controllers were carried out to see the performance of the two controllers, and know the best controller.

Different models were presented, namely DC motor model, Artificial Neural Network (ANN) model and Proportional Integral Derivative (PID) model. The simulation of the models is shown in Figure 9. Models for ANN and PID are shown in figure 10 and figure 11, respectively.

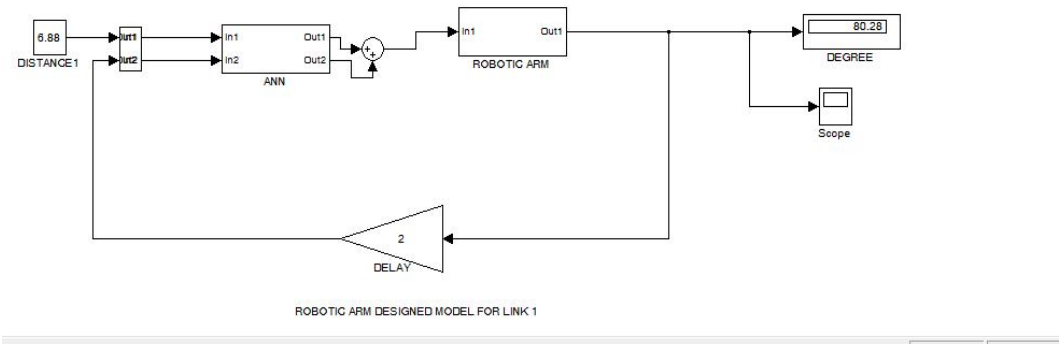


Figure 9: Artificial Neural Network Model for Speed Control

Proportional Integral Derivative PID controller was developed. PID controller is a conventional controller used when the system requires improvement under steady state transient condition, (Sukka et al, 2022). The conventional controller was used to compare its performance with the artificial neural network controller performance. The simulation graphs for the two controllers were shown in figures 12 and 14, respectively.

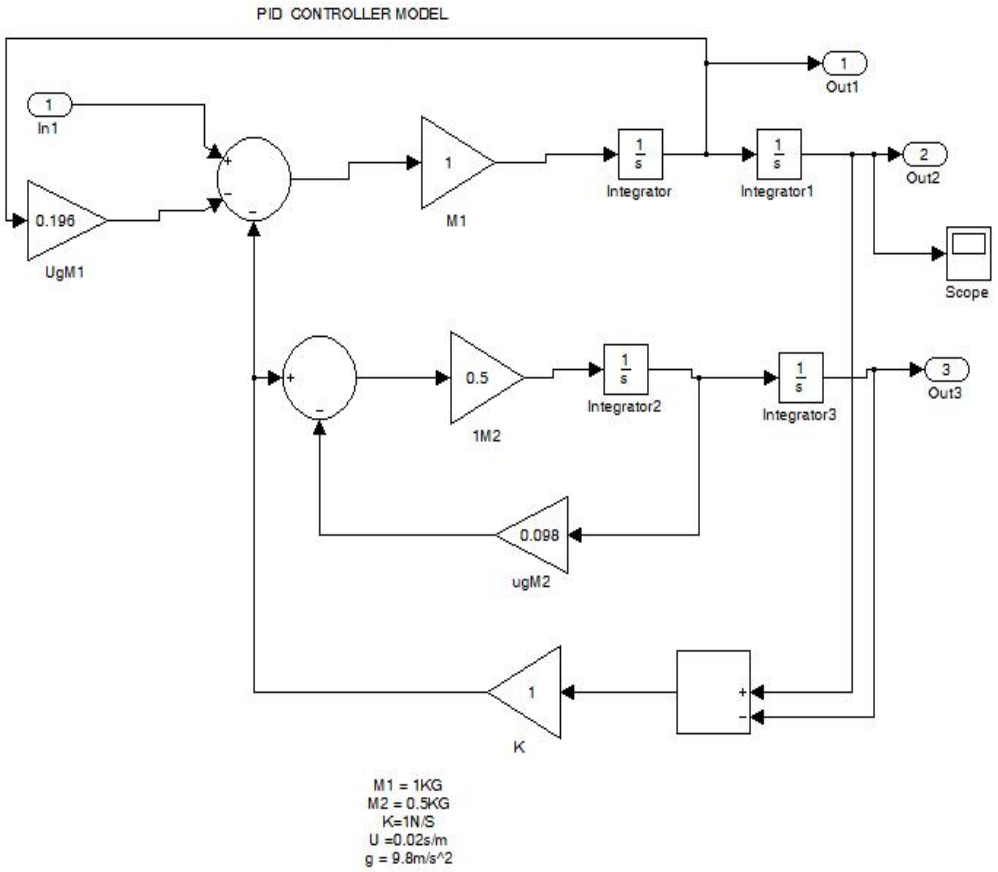


Figure 10: Designed Proportional Integral Derivative (PID) model

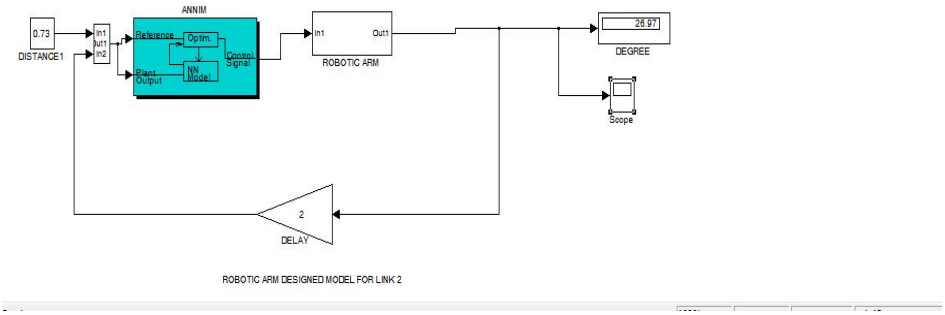


Figure 11: Simulated Artificial Neural Network (ANN) Model

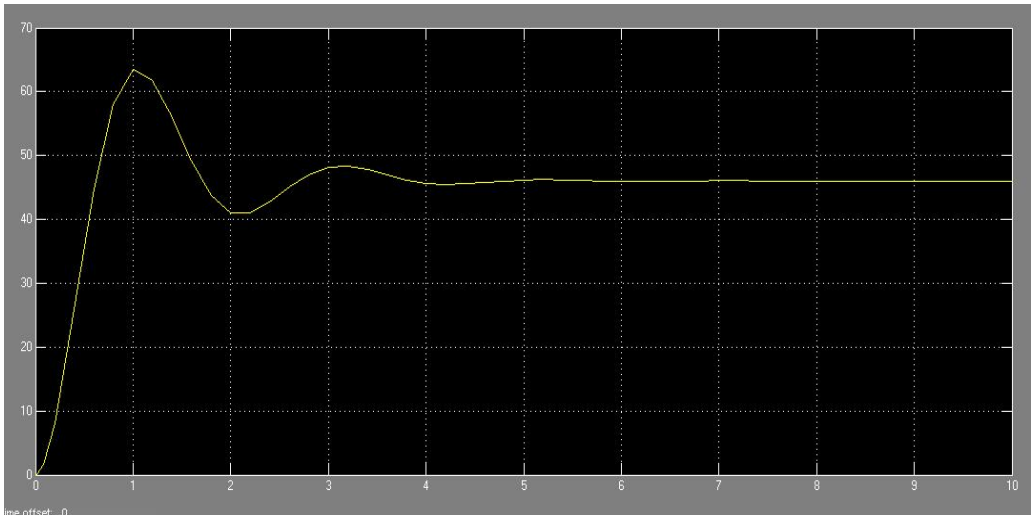


Figure 12: Simulated Graph of ANN Model

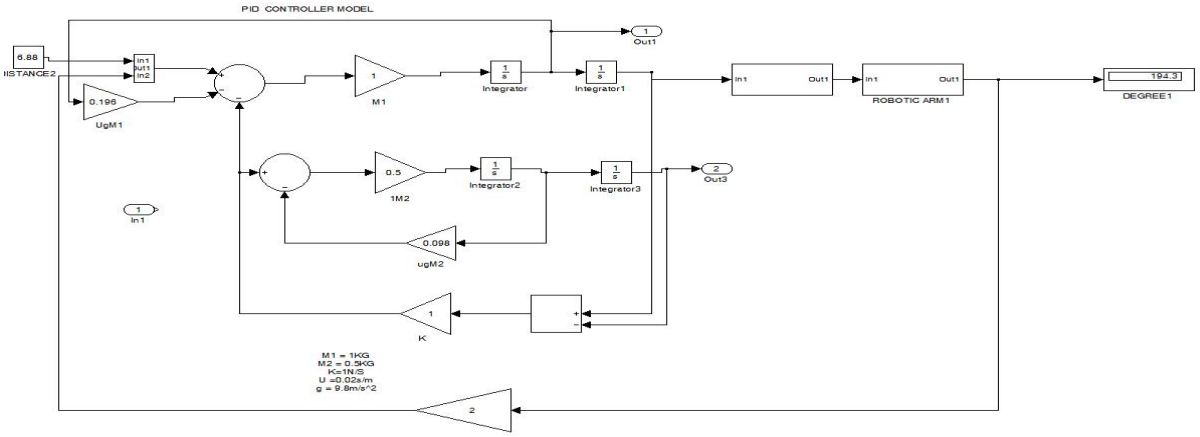


Figure 13: PID Controller Model

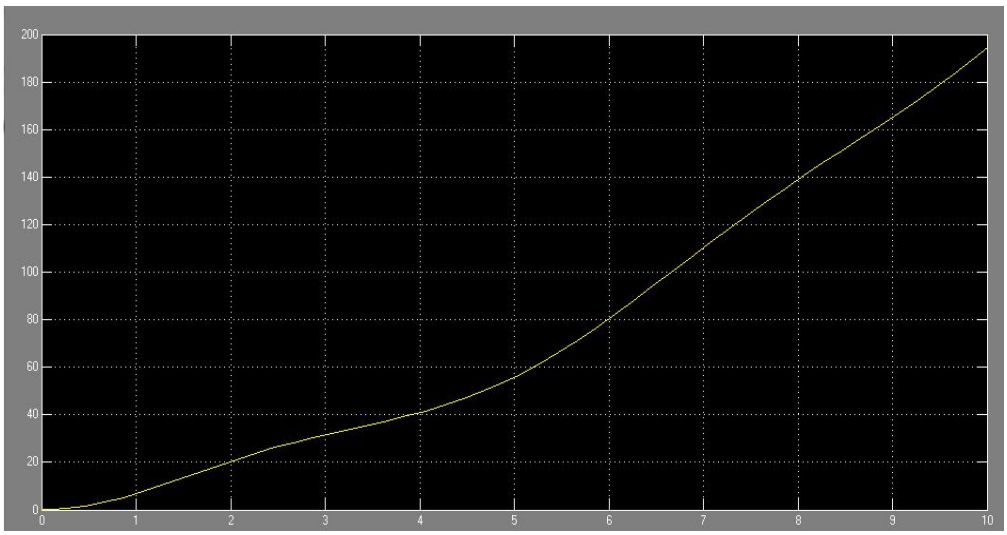


Figure 14: Simulated PID Controller Model

From the simulation graphs of the proposed models, it was noticed that the ANN model is more stable than the PID model.

Table 8: Speed of ANN and PID Controllers

S/N	Voltage of DC motor (Volt)	Speed (designed model) (m/s)	Speed (ANN controller) (m/s)	Speed (PID controller) (m/s)
1	0.900	202.10	202.50	205.90
2	1.220	273.70	274.50	278.80
3	1.280	287. 20	288.00	290.30
4	1.300	291.90	292.50	295.90
5	1.330	298.60	299.30	303.50
6	1.360	305.30	306.00	309.90
7	1.500	336.80	337.50	341.70
8	1.600	359.20	360.00	364.20
9	1.700	381.70	382.50	385.60
10	1.800	404.10	405.00	408.70

3.2 Percentage Error for Artificial Neural Network (ANN) and Proportional Integral Derivative (PID) controllers

The average % error for Artificial Neural Network (ANN) and Proportional Integral Derivative (PID) controllers was calculated from Table 8.0a as follows:

$$(Average\ \%\ Error\ for\ ANN) = \frac{2.36}{10} = 0.24$$

$$(Average\ \%\ Error\ for\ PID) = \frac{14.59}{10} = 1.46$$

Table 9: Percentage Error for ANN and PID

S/N	Speed from DC Motor model (m/s)	Speed from ANN model controller output (m/s)	Speed from PID model controller (m/s)	Percentage error in ANN controller (%)	Percentage error in PID controller (%)
1	202.1	202.5	205.9	0.19	1.88
2	273.7	274.5	278.8	0.29	1.86
3	287. 2	288.0	290.3	0.28	1.08
4	291.9	292.5	295.9	0.21	1.37
5	298.6	299.3	303.5	0.24	1.64
6	305.3	306.0	309.9	0.23	1.51
7	336.8	337.5	341.7	0.23	1.45
8	359.2	360.0	364.2	0.23	1.45
9	381.7	382.5	385.6	0.23	1.02
10	404.1	405.0	408.7	0.23	1.34

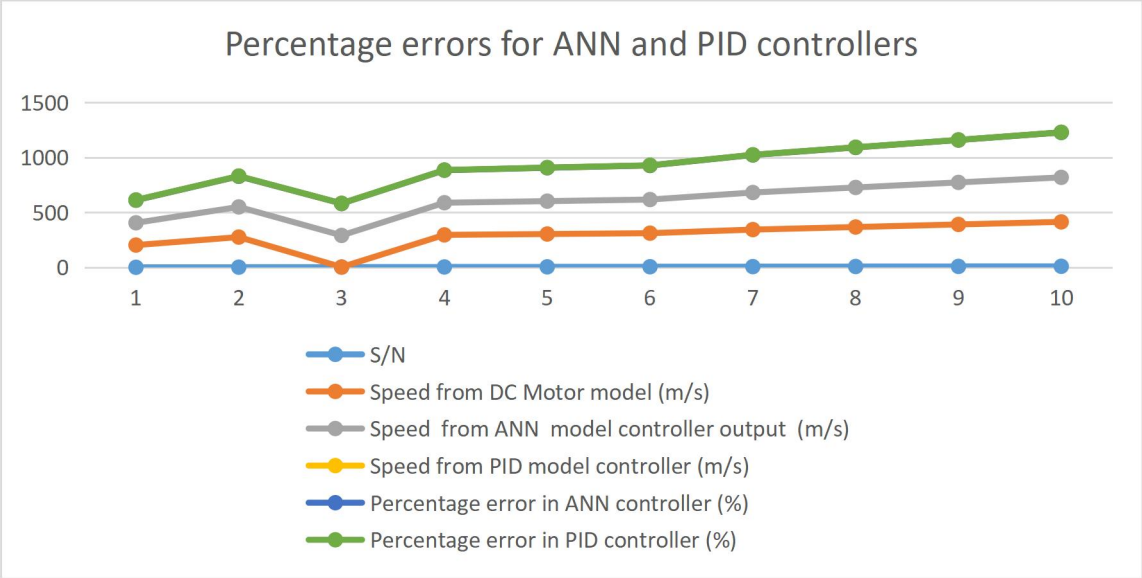


Figure 15: Analysis on Percentage Error for ANN and PID Controllers

IV. SUMMARY AND CONCLUSION

4.1 Summary

In this study, our primary objective was to enhance the control system for a DC motor. To accomplish this, two different control approaches were developed, an Artificial Neural Network (ANN) model controller and conventional Proportional-Integral-Derivative (PID) controller. Then simulations were conducted to compare the performance of these controllers, with a focus on stability and precision in regulating the motor's speed. The simulations showed that the ANN controller significantly outperformed the PID controller in terms of stability. When assessing the precision of speed control for the DC motor, the PID controller exhibited a deviation of 1.46%, while the ANN model achieved a remarkable deviation of just 0.24%. This substantial improvement in performance underscores the effectiveness of the ANN model in achieving precise and stable control.

4.2 Conclusion

In conclusion, the research has demonstrated that the implementation of an Artificial Neural Network (ANN) controller can result in substantial improvements in the control of a DC motor, particularly when compared to the traditional Proportional-Integral-Derivative (PID) controller. The ANN controller proved to be not only more stable but also significantly more precise in regulating the motor's speed. These findings highlight the potential for intelligent-based control systems to advance the field of motor control. The advantages of the ANN model in terms of stability and precision make it a promising technology for a wide range of applications. As technology continues to evolve, this research provides valuable insights into the potential for improved control systems, which can have significant real-world implications for industries reliant on precise motor control.

REFERENCES

Anis Hamza, and Noureddine Ben Yahia, (2023), "Artificial Neural Networks Controller of Active Suspension for Ambulance based on ISO Standards", Proceedings of the Institution of Mechanical Engineers, Part D Journal of Automobile Engineering 237(1):34-47, DOI:10.1177/09544070221075456

Basavarajappa Sukka, Rameshappa, Nagaraj Mudaka Pla Shadaksharappa (2022), "Applications of Artificial Neural Networks in Speed of Electrical Motors", International Journal of Electrical and Computer Engineering (IJECE), Vol. 12, No. 5, ISSN 2088-8708, PP 4700 – 4711

Caudill Maureen, Butler Charles (2019), "Naturally Intelligent Systems", Publisher: Bradford Books, Madonna Publishers Ltd, Accra

Entonia Eduardo de Barros Ruano (2022), "Application of Neural Network to Control System", Thesis Submitted to WALES University School of Electrical Engineering Science, for the degree of Doctor of Philosophy

Kenji Suzuki (2011), "Artificial Neural Networks", Tokyo Institute of Technology, Japan. Published11 April **2011**. Doi10.5772/644. ISBN978-953-307-243-2

Lewis, F.I., S. Jagannathan and A. Ildirek (2019). Artificial Intelligence in Engineering, [HCL07].C.-F..Hsu ... [LJY97].

Mano, Carlos. (2014). Neural Networks Explained. Retrieved from http://www.ehow.com/print/about_55853 09_neural-networks-explianed.htm.

Mehr, D, Richfield, S. (2017), "Neural Net Application to Optical Character Recognition", IEEE, 1st International Conference on Neural Network, volume 4, PP 711 -777

Wajeeha Ahmed, Areeshia Chaudhary, Gulfraz Naqvi (2022)," Role of Artificial Neural Networks in AI", Neuro Quantology, Volume 20, Issue 13, page 3365 – 3379.

