

Impact of Hidden Layer in Artificial Neural Networks

Maurya Vijayaramachandran¹, SiddiqueAfraaz N²

¹(Department of ECE, Sri Venkateswara College of Engineering, Sriperumbudur, India)

²(Department of ECE, Sri Venkateswara College of Engineering, Sriperumbudur, India)

Received 28 October 2020; Accepted 09 November 2020

Abstract: It's a technique for building a computer program that learns from data. It is based very loosely on how we think the human brain works. First, a collection of software "neurons" are created and connected, allowing them to send messages to each other. Next, the network is asked to solve a problem, which it attempts to do over and over, each time strengthening the connections that lead to success and diminishing those that lead to failure of the system. The paper focuses on such fail conditions and methods on how to improvise the success rate.

Background: As computers advanced into their infancy of the 1950s, it became possible to begin to model the rudiments of these theories concerning human thought. Nathaniel Rochester from the IBM research laboratories led the first effort to simulate a neural network. That first attempt failed. But later attempts were successful. It was during this time that traditional computing began to flower and, as it did, the emphasis on computing left the neural research in the background. Yet, throughout this time, advocates of "thinking machines" continued to argue their cases. In 1956 the Dartmouth Summer Research Project on Artificial Intelligence provided a boost to both artificial intelligence and neural networks. One of the outcomes of this process was to stimulate research in both the intelligent side, AI, as it is known throughout the industry, and in the much lower level neural processing part of the brain. In the years following the Dartmouth Project, John von Neumann suggested imitating simple neuron functions by using telegraph relays or vacuum tubes. Also, Frank Rosenblatt, a neurobiologist of Cornell, began work on the Perceptron. He was intrigued by the operation of the eye of a fly. Much of the processing which tells a fly to flee is done in its eye. The Perceptron, which resulted from this research, was built in hardware and is the oldest neural network still in use today. A single-layer perceptron was found to be useful in classifying a continuous-valued set of inputs into one of two classes. The perceptron computes a weighted sum of the inputs, subtracts a threshold, and passes one of two possible values out as the result. Thus the first neural network was born

Results: The **output** layer is responsible for producing the final result. There must always be one **output** layer in a **neural network**. The **output** layer takes in the inputs which are passed in from the layers before it and performs the calculations via its neurons, and then the **output** is computed.

Key Word:Neural network, latency reduction, efficiency increment, DNN, Regression model.

I. INTRODUCTION

Machine Learning has evolved to the extent that today it pretty much runs the world. Most services we use daily utilize machine learning to enhance user-experience like Google, Facebook, Twitter, Netflix, and much more. Machine Learning is a process of finding a pattern in a large amount of data and applying the pattern to the other set of data. Machine Learning is now known as the father of Deep Learning. Deep Learning is an enhanced version of Machine Learning which uses various techniques to find even the tiniest pattern and amplify it. This is achieved by a computing system called Artificial Neural Networks that mimics the working of a human brain. A Neural Network seeks to recognize the underlying relationship with a set of data with a series of algorithms to simulate the working of a human brain. Currently, Neural Networks and Deep Learning provide the best solution for Data Validation, Speech Recognition, Data Validation, etc.

1.1 Types of neural network

1.1.1 Perceptron:

Perceptron is a single layer neural network. It consists of an input layer and a single output neuron. It is used to classify inputs into two parts. Therefore, it is known as Binary Classifier. It accepts weighted inputs and applies the activation function to get the final result.

1.1.2 Feedforward Neural Networks:

Feedforward Networks are the most basic form of neural networks where the inputs only have a unidirectional flow through artificial neurons in the hidden layer(s) and to the output layer. The Number of Hidden Layers is directly proportional to the complexity of the function.

1.1.3 Multilayer Perceptron:

The Multilayer Perceptron is considered as a doorway to complex neural networks where the input data travels through a different layer of artificial neurons. The flow of data in this neural network is bi-directional - Forward and Backward. This neural network has at least three layers - an input layer, an output layer, and multiple hidden layers in between. The input data is multiplied with weights and fed to the activation function. With backward propagation, the weights are altered to reduce the loss.

1.1.4 Convolution Neural Network:

Convolution Neural Network consists of a three-dimensional arrangement of neurons as opposed to the standard two-dimensional arrangement of previously discussed networks. The first layer in this Network is the Convolutional Layer. This network can have one or more convolutional layers. Before moving onto the next layer, a convolutional operation is applied to the inputs. Thus, the network can be much deeper, consisting of fewer parameters. CNN shows promising results in image and video recognition.

1.1.5 Recurrent Neural Network - Long Short-Term Memory:

In a Recurrent Neural Network, the output of a specific layer is stored and fed back to the input. This influences the output of the next layer. The first layer of the recurrent neural network is constructed the same as the feedforward network, the recurrent process begins in the subsequent layers. Each layer computes the output and stores some value which may need to be used later. If the output is incorrect, then the neural network self learns with backward propagation. This makes the recurrent neural network very useful in text-to-speech conversion.

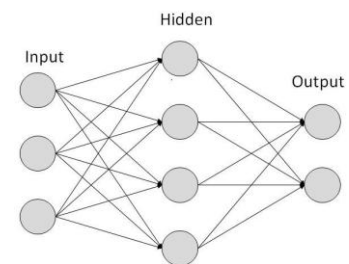
1.1.6 Modular Neural Network:

Modular Neural Network is multiple neural networks working to solve sub-tasks. These individual neural networks do not interfere with each other and work individually to achieve the output. This is seen to be computing large problems significantly faster by breaking it down into smaller components.

II. IMPLEMENTATION OF NEURAL NETWORK

2.1.1 Basic Implementation :

In the Diagram, each arrow from one neuron to another is a connection between those two neurons and acts as a channel for the flow of Information. Each connection has a weight that controls the signal between those to nodes. If the Neural Networks produces Desired or Accurate results, the weight for the connection is not changed. However, if the results produced by the neural network are Undesired or Inaccurate, then it alters the weights of the connections to improve the future results. Neural Networks are capable of learning and can be trained to produce desired results. There are several machine learning methods. The most used ones are - Supervised Learning, Unsupervised Learning, and Reinforcement Learning.



2.1.2. Supervised Learning- In this approach, the neural network is given with the set of inputs and desired outputs for those inputs. Thus, the neural network learns in a supervised manner.

2.1.3. Unsupervised Learning - The neural network is fed with a large set of data with no known outputs. It is made to find the hidden pattern in the set of data and group them accordingly.

2.1.4. Reinforcement Learning - In this approach of learning, The neural network has to achieve a goal with trial and error sequence and get to the desired solution. It tries a whole lot of different things and is rewarded or penalized based on its behavior helps or obstructs to reach the final desired result.

2.2. Hidden layer - From the flow of information in the neural network mentioned above, we know that all the computing and simulation of the human brain takes place in the 'mysterious' Hidden Layer. The hidden layer is located between input and output layers and performs transformations to the inputs that came into the network. A Neural Network can contain zero or more hidden layers. The name 'Hidden' layer represents that it is hidden from the end-user or external systems. Each hidden layer consists of the same no. of neurons that do a similar calculation and transmits it to the other neurons. This process divides a big impossible task into small possible tasks. The more the no. of hidden layers, the longer it takes to generate the output, and the more sample data it requires for self-learning.

III. LITERATURE SURVEY

The Existing algorithms for Neural Networks in use are:

3.1.1. The Feedforward algorithm

Where n is a neuron on layer l , and w is the weight value on layer l , and i is the value on the $l-1$ layer. All input

$$n_l = S \left[\sum_{l-1} (w_l i_{l-1}) \right]$$

values are set as the first layer of neurons. Then, each neuron on the following layers takes the sum of all the neurons on the previous layer multiplied by the weights that connect them to the relevant neuron on that following layer. This summed value is then activated.

3.1.2. A Common Activation Algorithm: Sigmoid

$$S(t) = \frac{1}{1+e^{-t}}$$

No matter how high or low the input value, it will get normalized to a proportional value between 0 and 1. It's considered a way of converting a value to a probability, which then reflects a neuron's weight or confidence. This introduces nonlinearity to a model, allowing it to pick up on observations with greater insight.

3.1.2.1 The Cost function

$$(E = \frac{1}{2}(n_L - t)^2)$$

The squared cost function lets you find the error by calculating the difference between the output values and target values. The target/desired values could be a binary vector for classification.

3.1.2.2 The Backpropagation

The error from the cost function is then passed back by being multiplied by the derivative of the sigmoid function S' . Thus, δ is first defined as the following:
at the output layer (beginning of backpropagation). Then we calculate the error through each layer which can be

$$\delta_L = (\Delta E_{n_L} * S'(n_L))$$

considered to represent the recursive accumulation of change so far that contributed to the error (from the perspective of each unique neuron). Past weight values must be transposed to fit the following layer of neurons. Finally, this change can be traced back to an individual weight by multiplying it by the weight's activated input

$$\frac{\partial E}{\partial n} = \delta_L = [T(w_{l+1}) * \delta_{l+1} * n_L(1 - n_L)]$$

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial n} * n_{l-1}$$

neuron value.

Applying the learning rate/weight updating

The change now needs to be used to adapt to the weight value. The eta represents the learning rate:

$$w = w - \left(\eta * \frac{\partial E}{\partial w} \right)$$

network running. These algorithms and their functions only scratch the surface of how powerful neural networks can be and how they can potentially impact various aspects of business and society alike. It is always important to overview how exciting technology is designed, and these five algorithms should be the perfect introduction.

IV. HIDDEN LAYER EFFECTS

Hidden Layers in neural networks make it far superior to any other machine learning strategies. This Hidden layer is present between the input and the output layer. There can be zero or more hidden layers in neural networks. Usually, one hidden layer is more than enough for most of the problems. Hidden layers consist of artificial neurons that are connected to every other neuron in the subsequent hidden layer. Mostly, the number of neurons in each hidden layer will be the same. The optimum number of neurons in a hidden layer can be determined with the following expression:

$$\text{Number Of Neurons} = \frac{\text{Trading Data Samples}}{\text{Factor} * (\text{Input Neurons} + \text{Output Neurons})}$$

Factor in the expression is an integer ranging from 1 to 10 and is used to prevent overfitting.

The neurons in the hidden layer do the same function of multiplying the input with a weighted bias and execute an activation function and pass it onto the next layer. Hidden layers allow us to divide the function of the neural network into parts. Each Hidden Layer builds on the work of the previous layer of neurons. Hidden layers are quite common in neural networks but their function varies from one case to another. As the number of hidden layers increases the problem complexity and computational time.

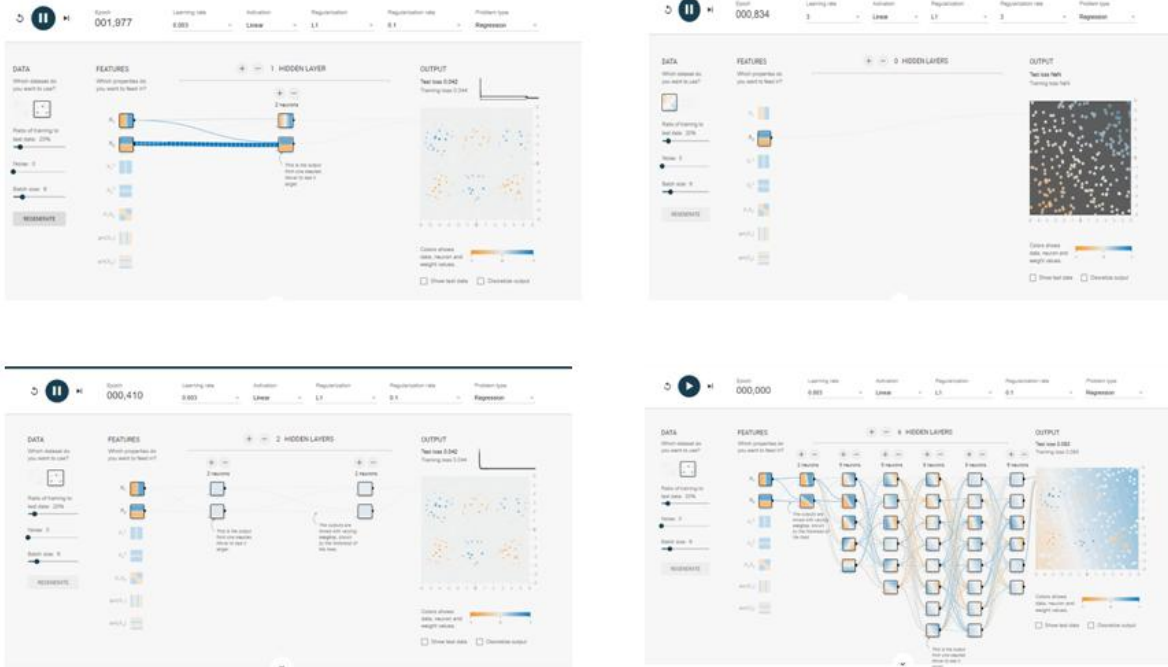
To produce the most accurate results with neural networks and reduce the losses, optimizing the neural network is a crucial part of training the network. Optimizers are used for optimizing the neural networks. Optimizers are algorithms or methods used to change the attributes of the neural network such as weights and learning rate to reduce the losses in the results. There are many optimizers like Gradient Descent, Momentum, AdaGrad, etc. Adam is the most widely used optimizer because it is fast and more efficient in a lot. Adam, *adaptive moment estimation*, uses the adaptive learning rate method to compute individual learning rates for different parameters. Adam uses estimations of first and second moments of the gradient to adapt the learning rate for each weight of the neural network.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

M(t) and V(t) are values of the first moment which is the Mean and the second moment which is the uncentered variance of the gradients respectively

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

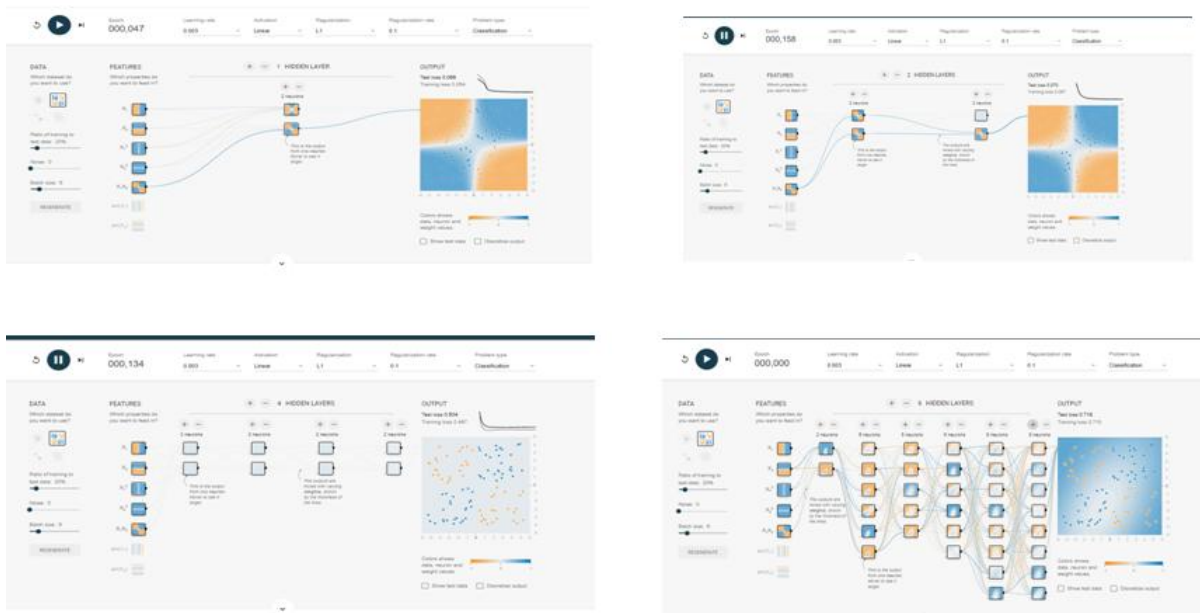
V. SIMULATIONS



VI. RESULTS

The above are the simulations for regression and classification models

- 6.1. As no. of the hidden layer increases, the training loss increases.
- 6.2. As no. of the hidden layer increases, the testing loss decreases.
- 6.3. Higher Epoch is needed for a sustained result.
- 6.4. The efficiency increases as well.
- 6.5. The training time decreases and testing time increases.



VII. CONCLUSION

The Greater the no of hidden layers, the greater is the latency, the greater is the result accuracy.
The problem can be counteracted using Artificial Neurons with GPU assisted computing capabilities.

REFERENCES

- [1]. <https://playground.tensorflow.org>
- [2]. <https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks-51466afa0d3e>
- [3]. Deep Learning in Computer Vision Principles and Applications, Mahmoud Hassaballah, Ali Ismail Awad
- [4]. Deep Neuro-Fuzzy Systems with Python, With Case Studies and Applications from the Industry, Himanshu Singh, Yunis Ahmad Lone
- [5]. Using Artificial Neural Networks for Analog Integrated Circuit Design Automation (SpringerBriefs in Applied Sciences and Technology) 1st ed. 2020 Edition

SiddiqueAfraaz N, et. al. "Impact of Hidden Layer in Artificial Neural Networks." *IOSR Journal of Engineering (IOSRJEN)*, 10(11), 2020, pp. 33-38.