World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(Review Article)

Mathematical modeling of neural networks: Bridging the gap between mathematics and neurobiology

Akinbusola Olushola [1,*] and Victoria Alao [2]

[1] *Department of Mathematics and Computer Science, Indiana University of Pennsylvania, Indiana, PA, USA.*
[2] *Department of Biology, Indiana University of Pennsylvania, Indiana, PA, USA.*

World Journal of Advanced Engineering Technology and Sciences, 2024, 13(01), 516–526

Publication history: Received on 14 August 2024; revised on 24 September 2024; accepted on 26 September 2024

Article DOI: https://doi.org/10.30574/wjaets.2024.13.1.0448

Abstract

This article explains how mathematical modeling is used in neural networks with much focus on artificial neural networks or ANN and the biological neural system. It offers an introduction to the objects defining the type of model – architecture of neural networks, the mathematical models of neuron behavior – and learning algorithms used for training. The article consolidates the progress and issues surrounding the enhancement of neural network usability toward more biological realism about artificial models and their biological counterparts. It further goes to the new methodologies, including neuromorphic computing, the hybrid model, and the ethical issues of AI. Using examples of particular cases and calculations in the article, the authors show examples of practical application and further research to address the gap between theory and practice. The conclusions made in this work stress the need for collaboration and integration of multiple fields and approaches in the development of neural nets and their adoptions.

Keywords: Neural Networks; Mathematical Modeling; Biological Neural Networks; Spiking Neural Networks; Deep Learning; Neuromorphic Computing

1. Introduction

The study of neural networks bridges two fascinating and complex fields. Its peculiarities are revealed in its connections with two rather unanticipated disciplines: mathematics and neurobiology. Neural networks have become one of the most used models of computation in artificial Intelligence as they mimic how the human brain works. On one side, the biological neural network in the human brain includes the complex neuron interconnection that results in cognition, perception, and learning. On the other hand, there are artificial neural networks (ANNs), which replicate these processes with the help of mathematical calculations for solving specific tasks in such areas as image or voice identification, language translation, and decision-making. By simulating the neural activity mathematically, the scientists sought to mimic the brain functions and move the studies on neuroscience and machine learning to a new level.

Mathematical modeling is used to capture the processes of the functioning of neural networks of both the natural and artificial types. In this regard, the perceptron, a simple neural model, can be easily explained in terms of weights, inputs, and activation equations. This simple model is the basis of consensus for more complex multilayer nodes in deep learning networks today. These models, known as backpropagation models and trained using optimization algorithms such as gradient descent, are developed based on how neurons in the brains adapt from their experiences. Like in system biology, where dendrites become repeatedly connected, artificial neurons work on a system of weights to reduce error at training.

[*] Corresponding author: Akinbusola Olushola

However, authors still notice a huge gap between artificial models of neurons and the real biological neurons that are the base for artificially created neurons. Even though the ANNs were proven to be very efficient in pattern recognition and decision-making, biological neurons operate at a much higher level, utilizing intricate biochemical and electric impulses. The problem, therefore, lies in finding ways to reduce this gap through adopting more realistic mathematical models that align with brain functionality. This article examines the utilization of mathematical modeling in Neuronal Networking and how it blends neurobiology and artificial Intelligence knowledge to enrich our understanding of thinking, learning, and computation.

2. Foundations of neural networks

Neural networks, as part of biological and artificial systems, have been researched for years, which brought a lot of valuable understanding of different forms of information processing. Biological neural networks' basic elements are neurons, which are the cells of the brain that are in charge of transmitting and processing data. A Real neuron's unit includes dendrites, which are receiving structures; axons, which are transmitting structures; and synapses, which are joining structures of neurons. This information is relayed via the emission of chemical substances across synapses, where electrical impulses are created and then spread across the network. Such interactions facilitate functions like learning, memory ops, and decision-making. In the human brain, billions of neurons create complicated networks; these networks provide the basis for the complex computations that underlie cognition, perception, and movement.

The discovery of biological neurons and their operation has greatly impacted the creation of artificial neural networks, which are electronic devices that emulate the functioning of neurons in biological neural networks. In its basic version, an ANN comprises nodes, neurons, and interconnections called edges, weighted like synapses. One of the simplest architectures of artificial neurons is known as perceptron. It requires several inputs, each multiplied by an assigned value. These results are then summed and passed through the activation function that decides the output of that neuron. If the production reaches a certain level, the perceptron "fires," much like a biological neuron where an action potential occurs after reaching the threshold. In equation form, this would be an input sum multiplied by weights plus a bias term passed through a specific activation function to yield an output. The perceptron model formed a basis for other higher-order ANNs that can learn from data.

Such limitations formed the basis of the multilayer networks, including the input layer, one or more hidden layers, and the output layer. The above-said multilayer networks are more effective than simple neural networks known as feedforward neural networks, which can model the non-linearity existing in data. In the case of multilayer perceptual networks, every individual layer of neurons receives the output from the higher layers and processes this output forward to the next level. Such networks can, therefore, innovate complicated patterns that simple perceptron networks cannot. Another important mathematical concept developed in the creation of ANNs was the backpropagation algorithm, which allows such networks to correct their errors by changing the weight of the neurons. Backpropagation helps in working out the partial derivative of the error function concerning every weight stored, and then applying the chain rule of calculus enables the network to learn through the process of weeding out several errors. Likewise, synaptic plasticity in the human brain is versatile in how the input data allows ANNs to learn and improve their performance.

Cognitive activation functions are important in the governmental neural network since they give the nonlinear transformations needed in a neural network. As was mentioned in the chapter about the nature of biological neurons, their functioning to inputs is not linear. Still, this conclusion was nonlinear and important for higher brain function. To accomplish this in ANNs, activation functions like the sigmoid, tanh, ReLu, and others are used. For example, the blond activation function transforms the input signal, resulting in an output that ranges from 0 to 1, while a real neuron fires only within an interval of frequencies. An example of such a function is the ReLU function for deep neural networks because it only passes negative values to an output of zero while passing all of the positive input values as they are so that the gradient is never too small, hence removing the restriction referred to as the vanishing gradient problem.

Artificial neural networks are a type of artificial intelligence model that mimics the workings of a biological brain and has its characteristics. Biological neural networks are much more extensive and include electrical and biochemical processes that control neurons' work. Neurons work in parallel; they provide different firing rates to various stimuli, while artificial neurons in any ANN are fed in parallel in a feedforward manner. Moreover, the synapses in the brain can always change, whereas, in the case of ANNs, the weights are trained and optimized during fixed training sessions. Nevertheless, due to their strong foundation of solving problems concerning pattern recognition, decision making, or even simple predictions, ANNs are rather effective, which proves the efficiency of utilizing mathematically modeled neural structures.

3. Mathematical models of neural networks

Neural network concepts are based on mathematical models and neural networks in computer science and biology are built according to these models. Imitating the principles of the traditional process flowcharting, at the core of such models is the concept of neurons – biological or artificial – as the black boxes that accept inputs and yield outputs. The goal of a mathematical model is to mirror the function of the neurons in terms of signal transmission, data learning, or changing with time. In ANNs, these processes are represented mathematically, where the interaction mechanisms of the inputs, weights, and activation functions are defined.

The perceptron is the simplest form of an artificial neural network, often considered the building block of more complex models. A single perceptron receives multiple inputs, each associated with a corresponding weight. The inputs are multiplied by these weights and summed, producing a net input. This net input is passed through an activation function to determine the output, which controls whether the neuron "fires." The mathematical equation representing a perceptron can be written as:

$$y = \sigma \left(\sum_{i=1}^{n} w_i x_i + b\right) \ldots\ldots\ldots\ldots \text{ (i)}$$

In this equation, x_i represents the input values, w_i the weights, b is the bias, and σ is the activation function. The bias term helps shift the activation function to control the neuron's firing threshold. This model allows for a simple binary decision-making process, akin to how biological neurons either fire or remain inactive depending on whether a certain threshold is exceeded. However, perceptrons are limited in handling more complex data, especially problems that are not linearly separable. This limitation led to the development of multilayer networks, where multiple perceptrons are stacked into layers to form a more powerful model.

In a multilayer neural network, also known as a feedforward neural network, neurons are organized into an input layer, one or more hidden layers, and an output layer. Each neuron in a given layer is connected to every neuron in the next layer, with the connections represented by weights. The network processes inputs through these layers by applying a series of matrix multiplications. Mathematically, the transformation in each layer can be expressed as:

$$\mathbf{Y} = \sigma (\mathbf{W} \,.\, \mathbf{X} + \mathbf{b}) \ldots\ldots\ldots\ldots \text{ (ii)}$$

Here, represents the matrix of weights, is the input vector, and is the bias vector. The activation function σ is applied element-wise to the matrix multiplication result. This structure allows the network to capture complex patterns in data, especially when deep architectures with multiple hidden layers are used.

For this reason, neural networks are intelligent: they are trained with data, and the training is enabled by a mathematical technique known as backpropagation. It uses the method known as backpropagation to minimize the error between the actual output and the target by adjusting the network weights. The error is defined by a loss function such as a mean squared error in the regression problems or a cross-entropy in the case of the classification problems. It has the smallest value of the loss function, where a weight change must be calculated in a direction that will minimize the error. This is done using gradient descent as an optimization technique, which seeks to estimate the gradient of the loss function about the weights.

Mathematically, the gradient descent algorithm updates the weights as follows:

$$\Delta w = -\eta \frac{\partial E}{\partial w} \ldots\ldots\ldots\ldots \text{ (iii)}$$

In this equation, η is the learning rate, E is the loss function, and $\frac{\partial E}{\partial w}$ is the gradient of the loss function concerning the weight w. The learning rate controls how large the weight updates are in each iteration. The gradient is calculated using the chain rule from calculus, allowing the error to be propagated backward through the network, layer by layer. The error gradient is computed for each weight in the network, and the weights are adjusted accordingly. This process is repeated for many iterations until the network converges to weights that minimize the error.

The choice of activation function plays a critical role in the performance of a neural network. Activation functions introduce non-linearity into the model, which is necessary for the network to learn complex patterns in data. Common activation functions include the sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU). The sigmoid activation function compresses the input into a range between 0 and 1, making it useful for binary classification tasks. Mathematically, it is represented as:

$$\sigma(x) = \frac{1}{1+e^{-x}} \text{.............} \quad \text{(iv)}$$

While sigmoid functions are biologically inspired, they suffer from the vanishing gradient problem, where the gradient becomes too small for effective learning in deep networks. The ReLU function has become widely used in modern neural networks to overcome this. ReLU is defined as:

$$f(x) = \max(0, x) \text{.............} \quad \text{(v)}$$

ReLU is computationally efficient and helps mitigate the vanishing gradient problem, enabling deep networks to learn more effectively. It allows only positive values to pass through, setting all negative inputs to zero, encouraging sparse activation, and improving learning efficiency.

Apart from feedforward networks, other more complex architectures include recurrent neural networks (RNNs) and convolutional neural networks (CNNs), which share the same mathematical framework but are developed to handle certain data types. RNNs are commonly applied to the sequence data due to their feedback mechanism, which enables certain information to pass in a loop. CNNs are supposed to handle a grid-structured input, such as images, using convolution operations that reflect only local patterns. These architectures are built from similar mathematical concepts of weights, activations, and optimization, but the architecture is chosen based on the task.

Another important aspect has been modeling neural networks in a mathematical sense, which has made them flourish in a number of applications. Neural networks' built-in formulation of inputs, weights, activation functions, and optimization algorithms has allowed ordinary perceptrons to complex deep multilayer structures for learning from data and making accurate predictions. Although these models take their strengths from real life biological systems, they are based and heavily dependent on mathematical notations to work properly. Although artificial neural networks have to do with computation, their strength and accuracy are seen in the mathematical model that forms their base.

4. Learning in neural networks

Memory is the unique procedure that helps biological and artificial neural networks to learn, generalize, and optimize their performance. Learning is identified as synaptic plasticity in biological systems, meaning a change of synaptic connections between the neurons. They are the basis of the development of artificial neural networks (ANNs) in which the connections, or weights, between artificial neurons are modified during the learning process. These adjustments enable ANNs to search data for a pattern and improve the network's classification, regression, and pattern analysis ability.

A foundational principle in biological learning is Hebbian learning, named after the psychologist Donald Hebb, who proposed that "neurons that fire together, wire together." This theory suggests that when two neurons are activated simultaneously, their synaptic connection strengthens, making it more likely that they will activate together in the future. Mathematically, Hebbian learning can be represented by the weight update rule:

$$\Delta w_{ij} = \eta x_i y_j \text{.............} \quad \text{(vi)}$$

In this equation, Δw_{ij} represents the change in the synaptic weight between neuron i and neuron j, η is the learning rate, and x_i and y_j are the activities of neurons i and j respectively. While this rule captures the basic idea of associative learning, it does not account for the complexity of tasks that modern artificial neural networks are designed to handle. Therefore, more sophisticated learning algorithms, such as backpropagation, are employed in ANNs to optimize performance.

In artificial neural networks, learning is typically framed as an optimization problem. The network must adjust its weights to minimize a loss function, which measures the difference between the predicted output and the actual target. The backpropagation algorithm is a key method used to achieve this. It works by calculating the gradient of the loss function concerning each weight in the network and then updating the weights in the direction that reduces the error. This gradient is computed using the chain rule from calculus, which allows the error to be propagated backward from the output layer to the input layer. This backward flow of error information is what gives backpropagation its name.

The core of learning in neural networks revolves around the gradient descent algorithm, which minimizes the loss function. The basic idea of gradient descent is to move iteratively in the direction of the steepest descent, which corresponds to the negative of the gradient of the loss function. Mathematically, the update rule for gradient descent is:

$$w^{(t+1)} = w^{(t)} - \eta \frac{\partial E}{\partial w^{(t)}} \text{.............} \quad \text{(vii)}$$

Here, $^{(t)}$ represents the weight at iteration t, η is the learning rate, and $\frac{\partial E}{\partial w^{(t)}}$ is the gradient of the loss function E concerning the weight. The learning rate controls the size of the weight updates: a small learning rate results in smaller updates, making learning more stable but slower. In contrast, a large learning rate speeds learning but risks overshooting the optimal solution.

When backpropagation is linked with gradient descent, it makes it possible for neural networks to learn through successive modification of the weights due to the mistakes they make during their computations. Training occurs generally through many cycles at the data sets, where the weights are adjusted at the end of each cycle. This process is called an epoch, and the training is done in conjugation several epochs until the network achieves the weights that minimize the error.

However, stochastic gradient descent is another version of the gradient descent optimization algorithm that is popular when training neural networks. In the actual implementation process of the gradient descent optimization, using the whole number of data points in the loss function to calculate its derivatives can be too complex. SGD, in each round of iteration, only takes a subset of the data. This makes it possible for the network to update the weights more frequently, which is very helpful in converging. However, since SGD uses only a subset of the data, the updates can be noisy, and therefore, the value of the loss function also steps up and down. Nevertheless, in the case of SGD, the network can get away with a local minimum while keeping the randomness, making it a powerful optimization technique.

Learning in neural networks also entails using activation functions, which adds non-linearity. When non-activation functions are not utilized in neural networks, they become just a linear model and cannot identify intricate patterns within data. Currently, activation functions are used in sigmoid, tanh, ReLU, and others. For instance, the sigmoid function starts from a range of input signals and scales it down to the probabilities range of 0 to 1. However, for deriving the gradient, sigmoid functions have a problem called the vanishing gradient problem, especially while training deep networks. To overcome this, rectified linear unit (ReLU) has emerged as a leading choice for activation functions. ReLU takes out the negative quantities and returns the input if the quantity is positive; if the amount is negative, ReLU returns a Quantity of zero; this makes it possible to avoid the vanishing gradient problem, and there is faster learning in the deep networks.

It is assumed that learning is more than a mere process of reducing error; it also includes the network's generalization ability, which is the ability of the segment to perform at an optimum level when put on unseen data. Training neural networks have certain drawbacks, such as overfitting, when the model learns the training data too well and cannot handle new data. Some methods, such as Dropout, regularization, and early stopping, are the solutions employed to prevent over-fitting. For instance, Dropout drops out some neurons during backpropagation, forcing the network to over-rely on some of its neurons, and in so doing, the network comes up with better features that generalize well.

Learning in neural networks can be defined as using certain mathematical elements interconnected with optimization principles. It's learned from the Hebbian rule through complex rule involving the backpropagation algorithm in the neural networks where weight is adjusted to enhance the reduction of the error. Stochastic gradient descent and other similar phenomena are the main methods used to optimize these weights. Activation functions enable the required nonlinear mapping of data. At the same time, regulatory acts help prevent overfitting during model training so that the model performs well on the unseen data later. When networks become more complex and effective, the base of mathematics for learning stays significant.

5. Biological models and mathematical equivalents

Biological neural networks, composed of billions of neurons and trillions of synaptic connections, serve as a key source of inspiration for artificial neural networks (ANNs). The complexity and adaptability of the human brain provide a powerful model for learning, memory, and decision-making, and researchers have long sought to bridge biological principles with mathematical models to understand both natural and artificial Intelligence better. While ANNs are rooted in mathematical abstractions, they are designed to emulate some of the core functions of biological neurons. Despite the differences between biological and artificial systems, significant progress has been made in constructing mathematical equivalents that capture key aspects of neural function.

In biological neural networks, neurons communicate via synapses, where signals are transmitted through electrical impulses and chemical neurotransmitters. Each neuron has dendrites that receive input signals from other neurons and an axon that sends the output to other neurons. The strength of a connection between two neurons is modulated by synaptic plasticity, which allows the brain to learn and adapt by changing the efficacy of synaptic transmission based on experience. The mathematical equivalent of this in ANNs is the weight associated with the connection between two artificial neurons. Weights determine one neuron has an influence on another, and they are adjusted during training to optimize the network's performance, similar to how synaptic strengths change in the brain during learning.

A mathematical model commonly used to describe how biological neurons function is the integrate-and-fire model. In this model, neurons are viewed as units that integrate incoming signals until they reach a certain threshold. When the input exceeds this threshold, the neuron fires an action potential, sending a signal down its axon to communicate with other neurons. Mathematically, the neuron's behavior can be described by a differential equation that models the membrane potential (t) as it evolves in response to incoming synaptic inputs. The membrane potential increases as inputs accumulate, and when the threshold is reached, the neuron "fires," and the potential resets. This model can be expressed as:

$$\frac{dV(t)}{dt} = -\frac{V\,(t)}{\tau} + I(t) \text{ .... (viii)}$$

where $V(t)$ is the membrane potential at time t, τ is the membrane time constant, and $I(t)$ represents the input current to the neuron. Once (t) reaches a threshold value, the neuron fires and the potential is reset to a baseline value. This model, while simple, captures the basic behavior of spiking neurons in biological systems.

Artificial neural networks adopt a simpler approach to neuron activation, where the behavior of a neuron is modeled as a weighted sum of inputs followed by an activation function. In the perceptron model, the input from multiple neurons is combined linearly, and the result is passed through an activation function that determines whether the neuron "fires" or not. The activation function in ANNs serves as a mathematical equivalent to the biological process of triggering an action potential. Common activation functions include the sigmoid, tanh, and rectified linear unit (ReLU). These functions introduce non-linearity into the network, allowing ANNs to model complex patterns and make decisions that go beyond simple linear separability.

Beyond single neurons, researchers have developed mathematical models that capture the dynamics of entire biological networks. One such model is the Hopfield network, a type of recurrent neural network inspired by the brain's associative memory. In a Hopfield network, neurons are fully connected, and the network can store patterns and retrieve them based on incomplete or noisy input. This behavior mimics the associative memory capabilities of the human brain, where partial information can trigger the recall of a full memory. Mathematically, the Hopfield network is described by an energy function that represents the state of the network, and the network evolves to minimize this energy. The energy function E is given by:

$$E = -\frac{1}{2}\sum_{i,j} w_{ij} s_i s_j \text{ .... (ix)}$$

where w_{ij} represents the weight between neurons i and j, and s_i is the state of neuron i, which can be either +1 or -1. The network's dynamics are governed by energy function minimization, similar to how biological networks settle into stable states corresponding to memories or learned patterns.

Another important biological model is the spiking neural network (SNN), which approximates how biological neurons function more closely than traditional ANNs. In spiking neural networks, information is transmitted in the form of discrete spikes or action potentials, rather than continuous values as in ANNs. These spikes are generated when a neuron's membrane potential exceeds a threshold, and the timing of these spikes carries crucial information. Spiking neurons operate asynchronously, meaning they fire at different times, which reflects the behavior of biological neurons more accurately. The mathematical description of spiking neurons involves complex differential equations that model both the timing and the intensity of spikes, and learning in SNNs often relies on mechanisms like spike-timing-dependent plasticity (STDP). STDP is a biological learning rule that adjusts synaptic strengths based on the relative timing of spikes between pre- and post-synaptic neurons.

Mathematically, STDP can be described by a function that modifies the synaptic weight w based on the time difference Δt between the pre-synaptic spike and the post-synaptic spike:

$$\Delta w = \begin{cases} A^{+}e^{-\Delta t/\tau^{+}} \; if \; \Delta t > 0 \\ -A^{-}e^{\Delta t/\tau^{-}} \; if \; \Delta t < 0 \end{cases} \text{ .... (x)}$$

Here, A^{+} and A^{-} are constants that determine the magnitude of weight changes, and τ^{+} and τ^{-} are time constants that control the learning rate. This rule captures the essence of Hebbian learning, where synaptic connections are strengthened if the pre-synaptic neuron fires shortly before the post-synaptic neuron and weakened otherwise.

In conclusion, biological neural networks provide the foundation for many mathematical models used in artificial Intelligence. The synaptic plasticity, action potentials, and associative memory observed in the brain have inspired key mathematical frameworks such as the perceptron, Hopfield networks, and spiking neural networks. Although ANNs simplify many biological processes, ongoing research continues to refine these models, making them more biologically plausible and improving their ability to emulate natural Intelligence. By bridging the gap between biology and mathematics, neural networks advance our understanding of artificial and biological Intelligence.

6. Modern approaches to bridging the gap

However, significant developments are afoot to close the gap between BNNs and ANNs because of recent developments in neuroanatomy and computational science. These approaches are used in an effort to bring artificial models closer to biological models and increase the capacity of artificial structures to mimic cognitive processes. Based on the principles of neurobiology, experts are improving the architecture of ANNs and enhancing the training process to make them as close as possible to the human brain.

One of the most noteworthy strategies is the emergence of neuromorphic computing, which aims to create form hardware and algorithms that resemble the human brain. Neuromorphic systems, therefore, employ analog circuits and Spiking Neural Network (SNN) to emulate how the brain functions. Unlike the conventional digital neural networks that use continuous values, the SNNs transmit data by spiking signals similar to those of biological neurons. Silicon devices like IBM TrueNorth or Intel Loihi are examples of neuromorphic hardware that resemble the biological brain and its efficient power use and parallel processing. These systems take advantage of the time and amplitude of the peak to sample and transmit information, and may have advantages in the speed and the power requirements over conventional 'baseline-shift' systems.

Another contemporary approach includes the incorporation of learning algorithms that are inspired by the brain and is an advancement of backpropagation. Spike-timing-dependent plasticity (STDP), one of the algorithms, is based on the biological three-factor learning rule of synaptic plasticity. STDP, however, provides that the change in synaptic weights depends on the relative timing of pre and post-synaptic spikes as a paradigm of the brain's ability to learn in temporal domains. They are considering that scientists have modified STDP for implementation in artificial neural networks in the framework of SNN for more realistic behavioral modeling. By incorporating STDP, the learning process in neural networks is made more adaptive and less sensitive to noise and is made to mimic how biological systems adjust the synaptic strength of neurons.

Another example of this has also been witnessed in deep learning models, particularly deep convolutional neural networks (CNN). CNNs are founded on the kind of visual cortex whereby neurons in several layers react to the features of the visual stimuli. When employing convolutional layers and where the pooling operations are applied, the CNNs can capture spatial hierarchies and features of data in the images. This process mimics how the brain can dissect the visual details and hierarchy, thus making it possible to attain extraordinary feats in image and especially video analysis.

In addition, more studies have focused on creating a new model that combines a realistic biological neuronal network with an artificial one. For instance, integrating molecular structures such as attention mechanisms and self-organizing maps with deep learning models enhances the latter's ability to attend to relevant knowledge and address new data. The embedded mechanisms include attentiveness that sharpens the model inputs and improves the total performance, particularly where selection or decision-making is involved based on the information content.

Similar to closing one's eyes to the tremendous development of BMIs, it also closes the gap by providing direct links between neural circuits and artificial apparatus. MEs enable the capture and analysis of neural signals' activity in the brain to operate outside equipment or give feedback to neural nets. Apart from the revelations about human brain activity, it presents various possibilities for interacting with biological and artificial systems. This can lead to a better design of the complex neural network.

7. Calculations and case study examples

In examining the mathematical modeling of neural networks and their biological counterparts, calculations play a crucial role in understanding and validating these models. By applying mathematical equations to real-world data, researchers can assess the performance of neural networks and refine their algorithms. This section will present some key calculations involved in neural network operations and illustrate their application through a case study example.

7.1. Calculations in Neural Networks

One fundamental calculation in neural networks involves determining the output of a neuron. For a single-layer perceptron, this output is computed by calculating the weighted sum of the inputs and then applying an activation function. Suppose a neuron receives inputs $x_{1,2}$, …, x_n with corresponding weights w_1, w_2, …, w_n, and a bias term b. The net input z to the neuron is:

$$z = \sum_{i=1}^{n} w_i x_i + b \text{…………..} \quad \text{(xi)}$$

The activation function σ then determines the neuron's output y. For example, using the sigmoid activation function, the output is given by:

$$y = \sigma(z) = \frac{1}{1+e^{-z}} \text{…………..} \quad \text{(xii)}$$

To illustrate the training process, we use gradient descent for optimizing the weights. The gradient of the loss function concerning a weight w_j is calculated to update the weight. Consider the loss function L, which measures the difference between the predicted output and the true target. The weight update rule for gradient descent is:

$$w_j \leftarrow w_j - \eta \frac{\partial L}{\partial w_j} \text{…………..} \quad \text{(xiii)}$$

Where η is the learning rate, this update is applied iteratively to minimize the loss function over multiple training epochs.

7.2. Case Study Example

To demonstrate these calculations, let's consider a simplified case study involving a neural network designed to classify handwritten digits using the MNIST dataset. This dataset consists of 28x28 pixel grayscale images of digits 0 through 9, and the network is tasked with categorizing each image into one of these ten categories.

7.3. Network Architecture

Assume a basic neural network architecture with one hidden layer:

- Input layer: 784 neurons (one for each pixel)
- Hidden layer: 128 neurons
- Output layer: 10 neurons (one for each digit class)

Training the Network

7.3.1. Forward Propagation

- Calculate the weighted sum and activation for each neuron in the hidden layer.
- For an input vector x, the weighted sum for a hidden neuron j is:

$$z_j = \sum_{i=1}^{784} w_{ji} x_i + b_j \text{…………..} \quad \text{(xiv)}$$

- Apply the activation function (ReLU or sigmoid) to obtain the output of the hidden layer.
- For the output layer, calculate:

$$z_k = \sum_{j=1}^{128} w_{kj} h_j + b_k \text{…………..} \quad \text{(xv)}$$

Where h_j is the output from the hidden layer, and w_{kj} are the weights connecting the hidden layer to the output layer.

- Apply the softmax function to get the predicted probabilities for each class:

$$P(y = k|x) = \frac{e^{zk}}{\sum_{i=1}^{10} e^{zi}} \text{………….. (xvi)}$$

7.3.2. Backpropagation

- Compute the error (loss) using a loss function such as cross-entropy:

$$L = \sum_{k=1}^{10} y_k \log(P(y = k|x)) \text{………….. (xvii)}$$

- Using the chain rule, calculate the gradients of the loss with respect to weights and biases in both hidden and output layers.
- Update weights using gradient descent:

$$w_{kj} \leftarrow w_{kj} - \eta \frac{\partial L}{\partial w_{kj}} \text{ ………….. (xviii)}$$

7.4. Case Study Results

For simplicity, train the network on a subset of the MNIST dataset with 1,000 images. After ten epochs of training with a learning rate η = 0.01, the network might achieve an accuracy of around 95% on a validation set. This result demonstrates the effectiveness of the neural network model in learning to classify digits.

Mathematical calculations underpin the operation and training of neural networks. These calculations are essential for optimizing network performance, from basic operations like weighted sums and activation functions to more complex procedures like gradient descent. The case study illustrates how these calculations are applied in practice, showcasing the process of training a neural network on real-world data and evaluating its performance.

8. Challenges and future directions

As neural networks and their mathematical models continue to evolve, several challenges remain, and numerous opportunities for advancement lie ahead. Addressing these challenges is crucial for enhancing neural networks' performance and applicability across various domains. The following section outlines some primary challenges and explores future directions in the field.

8.1. Challenges

- Scalability and Computational Complexity: Neural networks require substantial computational resources, particularly deep learning models with numerous layers and parameters. Training these models involves extensive matrix operations and gradient calculations, which can be computationally expensive and time-consuming. As models grow in size and complexity, the demand for computational power and memory increases, posing scalability challenges. Optimizing algorithms to be more efficient and utilizing advanced hardware, such as GPUs and TPUs, are essential for addressing these issues.
- Interpretability and Transparency: Despite their effectiveness, neural networks often operate as "black boxes," making it difficult to understand how they arrive at specific decisions. This lack of interpretability can be problematic, especially in critical applications such as healthcare and finance, where understanding the rationale behind decisions is crucial. Developing methods for explaining neural network decisions, such as attention mechanisms and visualization techniques, is an ongoing challenge. Enhancing model transparency is essential for building trust and ensuring ethical use of AI technologies.
- Data Quality and Quantity: The performance of neural networks heavily relies on the quality and quantity of training data. High-quality, diverse, and representative datasets are necessary for training effective models. However, acquiring and labeling large datasets can be resource-intensive and costly. Moreover, biases in training data can lead to biased model predictions. Addressing data-related issues through better data collection practices, synthetic data generation, and bias mitigation strategies is crucial for developing fair and accurate models.

8.2. Future Directions

- Neuromorphic Computing: Neuromorphic computing aims to create hardware and algorithms more closely mimic the brain's functioning. This approach involves developing systems that use spiking neural networks and analog circuits to achieve energy-efficient and highly parallel processing. Neuromorphic chips, such as

Intel's Loihi and IBM's TrueNorth, offer promising directions for advancing AI capabilities while reducing energy consumption. Continued research in neuromorphic computing could lead to more brain-like artificial systems and improved performance in tasks requiring real-time processing.

- Hybrid Models: Integrating biological principles with artificial neural networks is an exciting area of research. Hybrid models combining biological learning rules, such as spike-timing-dependent plasticity (STDP), with traditional deep learning techniques can enhance neural networks' adaptability and robustness. Exploring these hybrid approaches could lead to more biologically plausible models and better performance on complex tasks involving temporal and spatial patterns.
- Ethical AI and Fairness: As neural networks become more integrated into various aspects of society, ensuring their ethical use and fairness is paramount. Research into ethical AI focuses on developing guidelines and tools for creating transparent, accountable, and unbiased AI systems. Incorporating fairness-aware algorithms, improving data diversity, and implementing rigorous testing are essential for addressing ethical concerns and ensuring that AI technologies benefit all individuals equitably.

9. Conclusion

Indeed, the mathematical modeling of neural networks is a fluid interface between a mathematical system at one end of the spectrum and an application at the other. The origin of ANN goes back to mathematicians' ability to model complex behaviors of biological neurons and enhance Artificial and Natural Intelligence. From the basic architecture of the neural networks to such functions as learning and optimization, mathematics deeply impacts the development of the NNs.

Even so, some problems remain open, for example, in computational efficiency, transparency of models, and quality of data. There is a need to continue researching and innovating to correct these literacies. Neuromorphic computing and hybrid models paved the way for the development of better and biologically realistic neural architecture, and there is also a need to increase the model's fairness and interpretability for the ethical use of AI.

The future of neural network research can be built on a cross-disciplinary approach based on the cooperation of neuroscientists, computer scientists, and cognitive scientists. This will help researchers build models that more closely resemble the mechanics of the brain and fix real-world problems with greater accuracy.

The relationship between mathematics and neurobiology will further fuel breakthroughs and enhance existing neural computations with better adaptive networks. Thus, it is possible to gain a better understanding of neural networks' capabilities and make a valuable impact on various domains, including healthcare or self-driving cars. In the long run, the combined effort between the mathematical model and real biological concepts for designing AI will lead to even smarter, clearer, and more beneficial AI systems.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.

[2] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770-778). IEEE.

[3] Hinton, G. E., & Salakhutdinov, R. R. (2006). We are reducing the dimensionality of data with neural networks. Science, 313(5786), 504-507.

[4] Kording, K. P., & Wolpert, D. M. (2004). Bayesian integration in sensorimotor control. Trends in Cognitive Sciences, 8(6), 294-302.

[5] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[6] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). I am learning representations by back-propagating errors. Nature, 323(6088), 533-536.

[7] Olushola, A., Mart, J., (2022). Fraud Detection Using Machine Learning Techniques.10.13140/RG.2.2.33044.88961/1

[8] Olushola, A., Mart, J., Alao, V. , (2023). Implementations of Artificial Intelligence in Health Care. 10.13140/RG.2.2.36344.62729/1

[9] Olushola, A., Mart, J., Alao, V., (2023). Predictive Modelling For Disease Outbreak Prediction. 10.13140/RG.2.2.17470.259

[10] Spiking Neural Network Group. (2020). Spiking neural networks and their applications. Journal of Computational Neuroscience, 48(1), 1-3.

[11] Zhang, X., Zhao, H., & Zhang, Y. (2020). Spiking neural networks for brain-inspired computations: A review. Neural Networks, 126, 190-211