Chapter 1

Introduction

An artificial neural network is a set of algorithms that attempts to recognize underlying relationships within a set of data using a process similar to how a brain works. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. The purpose of the ANNs is to model how the human brain performs a particular task in a computer program. In addition to their functions, the biological neurons play a significant role in motivating the mechanism's working principle. Artificial neural networks mimic the behavior of biological neurons by making connections between input and output nodes. These connections can be adjusted and trained to allow the network to learn from its environment and improve its performance. In this thesis, we study about the stability and synchronization problems of neural networks. Neural network dynamics play a crucial role in implementing real-world problems such as associative memory, artificial intelligence, secure communication, signal processing, and optimization [2, 3, 4]. As an example, chaos synchronization of neural networks is used to enhance the security of signals transmitted between transmitters and receivers.

Control theory has important role in the synchronization problems of neural networks that is needed to stabilize the error systems. A wide range of controllers have been developed so far, including intermittent control [5], linear feedback control [6],

impulse control [7], and adaptive controller [8]. As a result, linear feedback control and adaptive control have a lower control cost; therefore, they are more effective and concise than discontinuous controllers. In this thesis, different kinds of synchronization criteria are used to study the dynamics of complex valued recurrent neural networks. The focus of the thesis is on the recurrent neural networks with feedback controllers. It is found that the feedback controllers are successful in achieving synchronization among the neurons of the network. Furthermore, the results show that the linear feedback control has the best performance in terms of the synchronization criterion. Overall, the present research concludes that the feedback controllers are effective and reliable way to achieve synchronization in complex valued recurrent neural networks.

The first section of the introductory chapter provides an overview of the biological neurons that motivates artificial neural networks. A discussion of artificial neural networks is described in the following sections. Basic mathematical concepts are introduced and then applied to investigate neural networks' dynamics in subsequent chapters. In the final section of this chapter, different definitions of synchronization are presented. An overview of the relationship between neural networks and synchronization is provided, and show how synchronization can be used to improve a neural network's performance. Finally, the implications of synchronization in neural networks have been discussed.

1.1 Biological Neural Network

The composition of a vast number of interconnected cellular units (nerve cells or neurons) and glial cells form the human brain and the nervous system. The glial cells provide physical and functional supports and protection for neurons. Although

neurons are found in a wide variety of sizes, shapes and locations, most of those are of uniform structures and neural signals are transmitted based on the electrical and chemical principles [9, 10]. The brain's essential features are the connectivity of neurons and the mechanism of transmitting electrochemical signals. Complex tasks can be performed by the brain with the help of both of those. The signals travel from one neuron to another, allow the brain to process the information and coordinate activities. The brain's ability to learn and remember information is also a result of this complex signaling system.

A prototypical neuron is given in the Figure 1.1. The central part of neuron is the cell body (soma) which contains the nucleus and other organelles that are essential for functioning all the cells. From the cell body of the neuron many root-like extensions called dendrites, which are receiving zones of synaptic signals or information from other neurons. A typical cell has many dendrites which are highly branched. The synapses (receiving zones of signals) are on the cell body and dendrites. Axons are long, fiber-like extensions of the cell body that carry signals to target neurons through specialized terminals (synaptic endings). Axons propagate self-generating electrical waves from the point of initiation at the cell body to its terminals as electrical signals. This process, known as action potential, is essential for neurons to communicate with each other. Axons are also important for the structural integrity of the brain, as those provide physical support to the neurons. Neurons have two types of signaling mechanisms, electrical and chemical. An electrical signal prevails in the interior of neurons, while chemical signals operate at synaptic ends. This difference allows neurons to communicate with each other quickly and efficiently. Synapses also allow neurons to form complex networks, which are essential for higher-level cognitive functions.

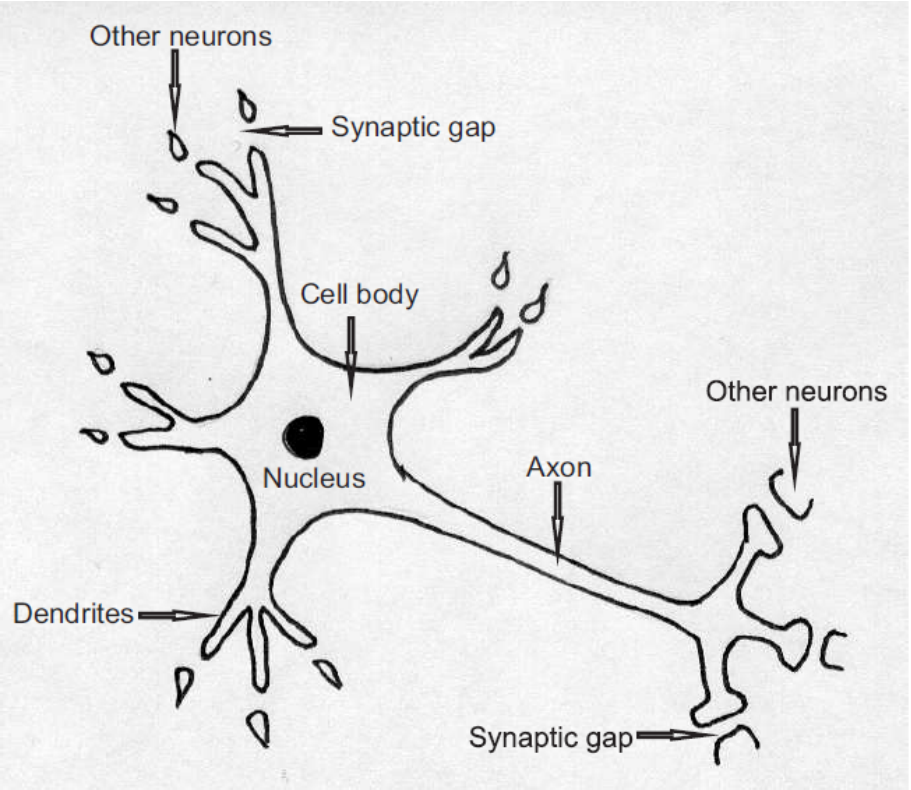

Figure 1.1: Schematic structure of a typical neuron [1].

Next, it described that how an electric current propagates across the axon membrane and how it can be modeled in a mathematical equation. Neuron membranes are composed of two thin layers of lipid molecules. The lipid molecules form a barrier between the inside and outside of the cell are known as the axon membrane. An electric current can flow across the axon membrane, which is responsible for the propagation of electrical signals within the neuron. Diffusion selectivity of the membrane varies with the length and time along an axon. In membranes, selective permeability is largely caused by ion channels, which allows certain types of ions to pass through along electrochemical gradients and concentration gradients. Ion channels are selective for certain ions, which allow for the diffusion selectivity of the membrane, and thus for the transmission of electrochemical signals to occur. This selectivity helps to create the action potential that is essential for nerve impulses.

There are ion concentrations of K^+, Na^+, Cl^-, and Ca^{2+}, and their differences across the membrane of the axon cause an electrical potential in the neurons. Across the axon membrane, the potential difference in a neuron in equilibrium is $-70mV$ (resting potential). This resting potential is maintained by the sodium-potassium pump, which pumps $3K^+$ ions out for every two Na^+ ions. This creates a difference in concentration and thus in potential across the membrane. When neurons are inactive, the ions distributed across the thin layers of the membrane make the interior or cytoplasm of neurons negatively charged relative to extracellular fluid. It's caused by a biological ion pump that works when the axon's interior becomes more positive. This is known as the resting potential, which helps to maintain the neuron's ability to fire action potentials. When a neuron is stimulated, the ions are redistributed, causing the interior to become more positive and creating an action potential. This flow of ions is what causes neurons to transmit the signals.

When an electric pulse injected in the axon of the neuron, the resting potential across the membrane is deviated. This deviation changes ions concentrations across the membrane that vary the potential difference. This change in potential difference generates an action potential or nerve impulse which moves from the axon hillock to the terminal end of the neuron. This impulse is then transmitted to the next neuron in the circuit. Thus, the membrane of the axon has the property of capacitance, i.e., separation of charges. Membranes play a similar role as resistor-capacitor (RC) circuits which will be discussed in the next section [11, 12]. The membrane permeability allows Na^+ ions to enter in interior of the neuron through the ion channels. As the voltage across the membrane increases above the threshold, the injected current produces a single pulse which propagates through the axon terminals. This propagated pulse triggers the release of neuro-transmitters which bind to receptors on the post-synaptic membrane. These neuro-transmitters then

produce an excitatory or inhibitory effect on the post-synaptic neuron. Due to the synaptic gap, the pulse signal travels through the axon terminals but stops at the synaptic ends. Through a special chemical mechanism called synaptic transmission, it is transferred from the synaptic ends to the target neuron. During synaptic transmission, substances called neuro-transmitters are released from synaptic ends and reach to the target neurons. These neuro-transmitters then bind to receptors on the surface of the target neuron, triggering an electrical signal that can be passed on to other neurons. This electrical signal can be either excitatory or inhibitory, depending on the type of neuro-transmitter released. The signal pulse acts as a current pulse in the target neuron, which follows the same mechanism as the presynaptic neuron. The current pulse in the target neuron triggers an action potential, which is a change in the voltage across the neuron's membrane. This change in voltage causes an electrical signal to be sent to the next neuron, continuing the chain of communication.

1.1.1 Biological Model

As discussed previously, neurons transmit electrical pulses along their axons to other neurons. Our brain performs a particular task of interest by transmitting signals from one neuron to another. This process is known as neural networking, and it is how our brains process and interpret information. This is how we are able to think, remember, and make decisions. Neural networks can also be used to create Artificial Intelligence(AI) systems that can learn and adapt to new situations. In order to understand the brain functioning, it is necessary to model the biological networks of neurons into the mathematical equation. Now, the general mathematical model of biological neural networks is discussed that was first formulated by Robert L. Harvey in his book Neural Networks Principles [10] published in 1994. Let us assume that

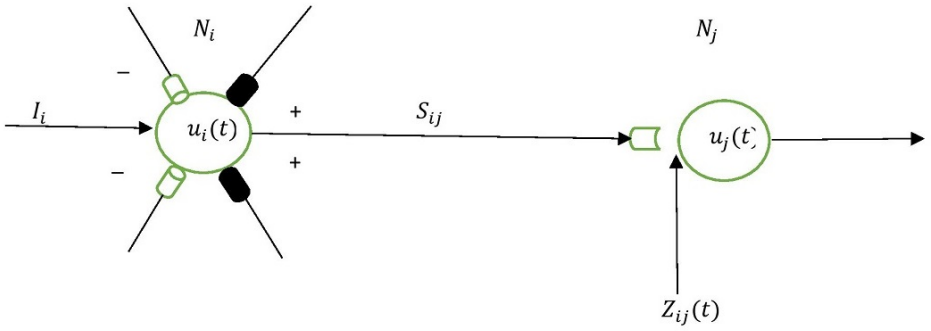

Figure 1.2: Schematic diagram of a network of neurons.

$N_1, N_2, ..., N_n$ are n neurons in the networks as shown in Figure 1.2. Consider a variable $u_i(t)$ to describe i-th neuron's state and a variable $Z_{ij}(t)$ describes the coupling strength between the neurons N_i and N_j. More precisely, let

$$u_i(t) = \text{the deviation of the } i\text{-th neuron from the resting potential.}$$

The variable $u_i(t)$ consists of the activation level of the i-th neuron. It is known as short term memory (STM) trace or axon potential.

The variable $Z_{ij}(t)$ connected with strength of interaction between the neurons N_i and N_j is defined as

$Z_{ij}(t) =$ the average release rate of neurotransmitter per unit axonal signal frequency.

This is called the synaptic coupling coefficient or long term memory(LTM) trace. It can be negative or positive depending on the signal that inhibits or excites the neurons to fire. If $Z_{ij}(t) > 0$, it means that i-th neuron is excited to send a signal to j-th neuron. If $Z_{ij}(t) < 0$, it means that i-th neuron is inhibited to send a signal to j-th neuron.

Assume a deviation in neuron's potential from equilibrium due to internal and external processes in neural network. The rate of change in neuron's potential is described

by the following differential equation.

$$\frac{du_i(t)}{dt} = \left(\frac{du_i(t)}{dt}\right)_{external} + \left(\frac{du_i(t)}{dt}\right)_{internal}, \quad \forall i. \tag{1.1.1}$$

Let us consider that the inputs from stimuli and other neurons are additive. Then, we get

$$\frac{du_i(t)}{dt} = \left(\frac{du_i(t)}{dt}\right)_{internal} + \left(\frac{du_i(t)}{dt}\right)_{excitatory} - \left(\frac{du_i(t)}{dt}\right)_{inhibitory} + \left(\frac{du_i(t)}{dt}\right)_{stimuli}, \forall i. \tag{1.1.2}$$

Assuming further that neuron's potential is decaying exponentially to equilibrium state without having external processes, we have

$$\left(\frac{du_i(t)}{dt}\right)_{internal} = -\alpha_i(u_i(t))u_i(t), \quad where \quad \alpha_i(u_i(t)) > 0, \quad \forall i. \tag{1.1.3}$$

Assume that the additive synaptic excitation is proportional to the pulse train frequency i.e.,

$$\left(\frac{du_i(t)}{dt}\right)_{excitatory} \propto \sum_{otherneurons} (frequnecy\ of signal)(synaptic\ coupling\ strengths), \tag{1.1.4}$$

which can be written in mathematical form as

$$\left(\frac{du_i(t)}{dt}\right)_{excitatory} = \sum_{l=1, l\neq i}^{n} S_{li}(t) Z_{li}(t), \quad \forall i, \tag{1.1.5}$$

where $S_{li}(t)$ is the average frequency of signal in the axon from neuron N_l to N_i, evaluated at N_i. The average signal frequency $S_{li}(t)$ depends on the propagation time delay τ_{li} taken by the signal to reach neuron N_i from N_l, and also depends on

the threshold value Γ_l for firing of N_l in the following manner

$$S_{li}(t) = f_l(u_l(t - \tau_{li}) - \Gamma_l), \tag{1.1.6}$$

where $f_l : R \to [0, \infty)$ is a given non-negative function called signal function. There are many different forms of signal functions which are commonly used in neural networks, which will be discussed later in detail.
Substituting (1.1.6) in equation (1.1.5), we find

$$\left(\frac{du_i(t)}{dt}\right)_{excitatory} = \sum_{l=1, l\neq i}^{n} Z_{li}(t) f_l(u_l(t - \tau_{li}) - \Gamma_l), \quad \forall i. \tag{1.1.7}$$

Assume hardwiring of the inhibitory inputs from the other neurons, i.e., the coupling strengths between the inhibited neurons are constant. Then have

$$\left(\frac{du_i(t)}{dt}\right)_{inhibitory} = \sum_{l=1, l\neq i}^{n} C_{li}, \quad \forall i, \tag{1.1.8}$$

where $C_{li} = b_{li}(t)h_l(u_l(t - \tau_{li}) - \Gamma_l)$, and b_{li} is constant coupling strength. The function h_l is a signal function. Generally, the threshold value Γ_l is same for every neuron in the network.

The stimuli are other external sources to change the neuron's potential, so we have

$$\left(\frac{du_i(t)}{dt}\right)_{external} = I_i, \quad \forall i. \tag{1.1.9}$$

Now, substituting the equations (1.1.3), (1.1.7), (1.1.8), and (1.1.9) in the equation (1.1.2), we get the so-called additive STM trace equation for $i = 1, 2, ..., n$, as

$$\begin{aligned}\frac{du_i(t)}{dt} =& -\alpha_i(u_i(t))u_i(t) + \sum_{l=1,l\neq i}^{n} Z_{li}(t)f_l(u_l(t-\tau_{li}) - \Gamma_l) \\ & - \sum_{l=1,l\neq i}^{n} b_{li}h_l(u_l(t-\tau_{li}) - \gamma_l) + I_i. \end{aligned} \tag{1.1.10}$$

As we have assumed that the excitatory synaptic coupling is varying with time, so we have the following equation based on Hebb's law.

$$\frac{dZ_{ij}(t)}{dt} = -A_{ij}(z_{ij}(t))Z_{ij}(t) + P_{ij}(t)[u_j(t)]^+, A_{ij}(Z_{ij}(t)) > 0, \quad \forall i, j, \tag{1.1.11}$$

where $P_{ij}(t) = \beta_{ij}f_i(u_i(t-\tau_{ij}) - \Gamma_i), \beta_{ij} \geq 0$, and

$$[u_j(t)]^+ = \begin{cases} u_j(t), if \quad u_j \geq 0, \\ 0, if \quad u_j < 0. \end{cases}$$

The second term of the equation (1.1.1) shows that to increase $Z_{ij}(t)$, neuron N_i must send a signal $P_{ij}(t)$ to neuron N_j, and at the same time N_j must be activated, i.e., $u_j(t) > 0$.

Note that the STM and LTM trace equations (1.1.10) and (1.1.11), respectively are not solvable until the coefficients $\alpha_i, A_{ij}, C_{li}, S_{li}, P_{li}$ and the external stimuli I_i are given.

1.2 Artificial Neural Networks(ANNs)

Artificial neural networks are a branch of machine learning models that are built using principles of neuronal organization discovered by connectionism in the biological neural networks constituting human brains. ANNs are used to replicate the way the human brain processes information, making them useful for solving complex problems such as image recognition and natural language processing. ANNs are also used in applications such as robotics and autonomous vehicles. In most of the cases, neural networks are simulated in software on digital computers or are implemented using electronic components [13]. There are computational units, which correspond to neurons, and interconnections, which correspond to synapses. Biological neuron models have inspired the architecture of artificial neurons, as well as the synaptic connections between them. By connecting artificial neurons into a network, machine learning algorithms can be used to process data and generate insights. Neural networks can be trained on large datasets to learn and recognize patterns, and can be used to solve complex problems. A comparison of an ANN with a biological network is shown in Table 1.1. The definition of an ANN may be the following, defined by Simon Haykin in his book [13] Neural Network: *A Comprehensive foundation*.

Definition 1.2.1. A massively parallel distributed processor called a neural network is made up of basic processing units and has a built-in tendency to store and utilise information from experience. It resembles the brain in two respects:

- Neural networks acquire knowledge from its environment through a learning process.
- Interconnection strengths, known as synaptic weights, are used to store the acquired knowledge.

Biological neural network	**Artificial neural network**
Neuron	Processing unit
Dendrite	Input unit
Cell Body	Processing function
Axons	Output unit
Synapse	Weights
External stimulus	Bias

Table 1.1: Comparison between a human brain and an artificial neural network.

The unique characteristic of a neural network makes it a powerful computing machine due to its massively parallel distributed architecture, as well as its ability to learn and generalize. As a generalization mechanism, it can produce reasonable outputs for inputs those have not been encountered during the learning process. Neural networks can be used for a variety of tasks, including image recognition, natural language processing, and pattern recognition. This makes them ideal for use in a variety of applications, such as autonomous vehicles, medical diagnostics, and robotics. A neural network performs to compute a complex problem in integrated manner. It cannot perform working individually.

Figure 1.3 shows the block diagram of a neuron of the network. The input signals of the neuron are represented by $u_1, ..., u_r, ..., u_n$ and the corresponding synaptic weights of the k-th neuron is defined by $w_{k1}, ..., w_{kr}, ..., w_{kn}$. Depending on whether a signal is inhibitory or excitatory, synaptic weights can be either positive or negative. Positive weights tend to increase the strength of the connection between two neurons, while negative weights do the opposite, weakening the connection. This

allows neurons to form complex networks and pathways, which is essential for the functioning of the brain. The manner of writing the subscript in synaptic weights w_{kr} is important to note, the first subscript refers to the neuron that is receiver of the signals, and the second subscript refers to the input ends of the synapse to which the weight refers. Unlike biological neurons, the ANNs may have synaptic weights these are based on interval of real numbers. This allows for more precise calculations and can result in a more accurate output. Furthermore, this allows for more complex and layered neural networks, which can be used to better model real-world data.

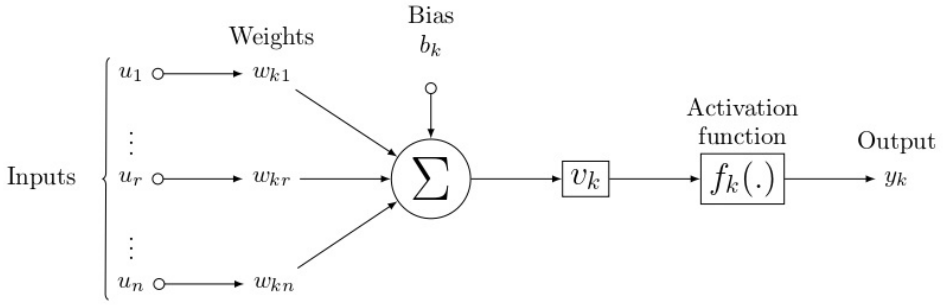

Figure 1.3: A diagram of an artificial neuron of the network.

The summing junction acts as a linear combiner of inputs, i.e., input signal u_r is multiplied by its respective synaptic weight w_{kr}. The bias b_k is an external stimulus that has the effects of decreasing or increasing the input of activation function $f_k(.)$, it depends on whether the bias is negative or positive, respectively. Mathematical expression for the linear combiner of input signals is given as

$$v_k = \sum_{r=1}^{n} u_r w_{kr} + b_k. \tag{1.2.1}$$

In equation (1.2.1), the output of the linear combiner u_k is an input value or an induced local field for the activation function. Neurons use activation functions to get the desired output. This is also known as a squashing function because it squashes the amplitude of the output of a neuron to some finite value. This finite value is usually between 0 and 1, and can be used to normalize the output of a neuron. Squashing functions are used in neural networks to ensure that the output of the neuron is within a certain range. In Figure 1.3, the output of a neuron in presence of the activation function is given as

$$y_k = f_k(v_k) = f_k(\sum_{r=1}^{n} u_r w_{kr} + b_k). \tag{1.2.2}$$

An activation function of a neuron can be linear or nonlinear in nature. Typically, there are three types of activation functions those are usually considered in designing neural networks:

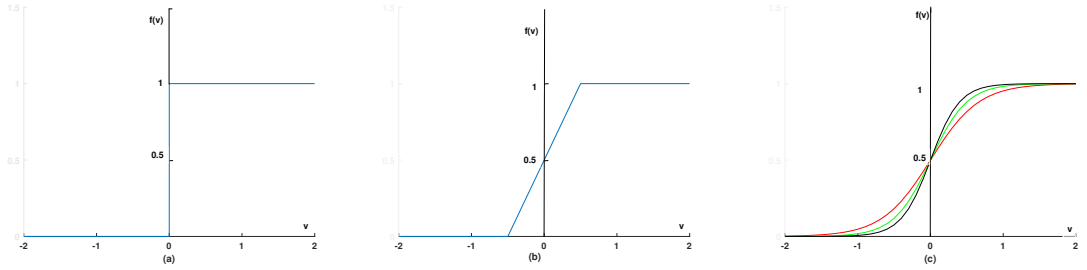

Figure 1.4: (a) A Step function, (b) Piecewise linear function, and (c) Sigmoid activation function for β = 3,4 and 5.

(a) *Step function(Threshold function).* For the activation function shown in Figure 1.4(a), we shall have the following output for a neuron of a Figure 1.3.

$$y_k = f_k(v_k) = \begin{cases} 1, if & v_k \geq 0, \\ 0, if & v_k < 0. \end{cases} \tag{1.2.3}$$

The model involving threshold function as an activation is known as McCulloch-Pitts model [14] in recognition of their pioneer work done by S. McCulloch and W. Pitts in 1943. The basic concept of this model is, a neuron fires a signal indicating "yes", i.e., "1" if the induced local field of that neuron is non-negative, and 0 otherwise. In the McCulloch-Pitts model, the activation function describes the all-or-none property of a neuron. This model is the foundation of ANNs, which are used in machine learning and deep learning algorithms. It is also the basis of understanding the functioning of a biological neuron. This model is the foundation of artificial intelligence.

(b) *Piecewise linear function.* A piecewise linear function is defined by the following function.

$$f(v) = \begin{cases} 0, if \quad v \leq -\frac{1}{\beta}, \\ 0.5, if \quad v_k < -\frac{1}{\beta} < v < \frac{1}{\beta}, \\ 1, if \quad v \leq \frac{1}{\beta}, \end{cases} \tag{1.2.4}$$

and shown in Figure 1.4(b). Here β is the amplification factor or the neural gain. Such type of activation functions has been widely implemented in cellular neural network [15, 16]. Figure 1.4(b) shows a piecewise linear function for $\beta = 2$. According to function given in (1.2.4), the piecewise linear function reduces to a threshold function as the amplification factor $\beta = 2$ approaches infinity.

(c) *Sigmoid function.* The sigmoid function shown in Figure 1.4(c). It is the most commonly used activation function in the construction of ANN models satisfying certain concavity and asymptotic properties. It is a strictly increasing, smooth, and bounded function. One of the examples of a sigmoid function is a logistic function defined by

$$f(v) = \frac{1}{1 + exp(-\beta v)}, \tag{1.2.5}$$

where β is an amplification factor, which is described as the slope parameter of the curve. For different values of β, we can get different slopes of the curve which as shown in Figure 1.4(c). From the function (1.2.5), It consists of the sigmoid function, which asymptotically approaches to 1 and 0 when $\beta \to \infty$ and $\beta \to -\infty$, respectively. Thus, sigmoid functions reduce to threshold functions when slope parameter β approaches infinity. This allows sigmoid functions to be used in a variety of applications, such as binary classification and image processing. The sigmoid function can also be used to model nonlinear relationships between inputs and outputs. Whereas a threshold function has the range of only two elements 0 and 1, and thus this function has the range in open interval (0,1).

It is worth noting that a threshold function is not smooth whereas a sigmoid function is a smooth function. As far as applications are concerned, the smoothness of sigmoid functions is crucial, which can be used for analog signal processing, as well as many mathematical theories [17]. Sigmoid functions can also be used in many machine learning algorithms, such as logistic regression and neural networks. In addition, they are useful for modeling nonlinear relationships between variables. If we consider random variables for the firing threshold of an all-or-non neuron with a Gaussian normal distribution function, then the expected output signal value is a sigmoid function of activity [18]. The model of this type of neurons is called stochastic model. ANN models increasingly use sigmoid activation functions for this and other reasons. Tangent hyperbolic and inverse tangent functions are also popular examples of sigmoid activation functions.

In ANNs, the network architecture of neurons is very important in terms of their applications or learning characteristics. There are different structures of neurons in networks designed for different purposes. For example, convolutional neural networks(CNNs) are used for image processing applications such as object detection.

On the other hand, recurrent neural networks(RNNs) are used for tasks such as natural language processing. Both of these architectures have distinct advantages and disadvantages those should be taken into consideration when designing a network. In general, for layered neural networks, there are two types of network architectures:

(a) *Feedforward network*. Figure 1.5 depicts a feedforward diagram of neural network

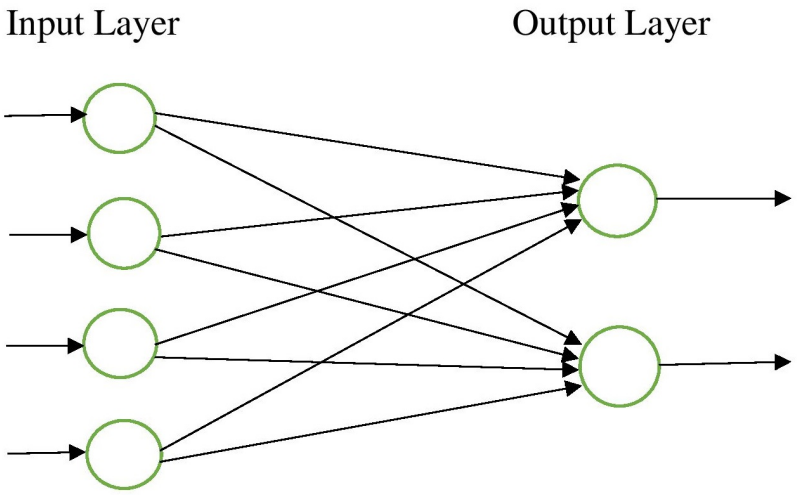

Figure 1.5: A feedforward network without hidden layer.

which has input layer of 4 neurons that projects onto output layer of 2 neurons. Nodes in input layer are source of signals sending towards the neurons of output layer, and not vice versa. Input neurons are not counted since they have no connections to process information. This type of neural network is referred to as a single layer neural network. Figure 1.6 shows multilayer feedforward networks which is more complicated feedforward neural network as compared to ordinary neural network. An input layer consists of five neurons, a hidden layer consists of three neurons, and an output layer consists of one neuron. This network is also known as a 5-3-1 network. In general, a multilayer feedforward neural network consists of input layer

of p neurons, first layer of q neurons, second layer of h neurons, and output layer of h_2 neurons, and is known as a $p - q - h_1 - h_2$ network.

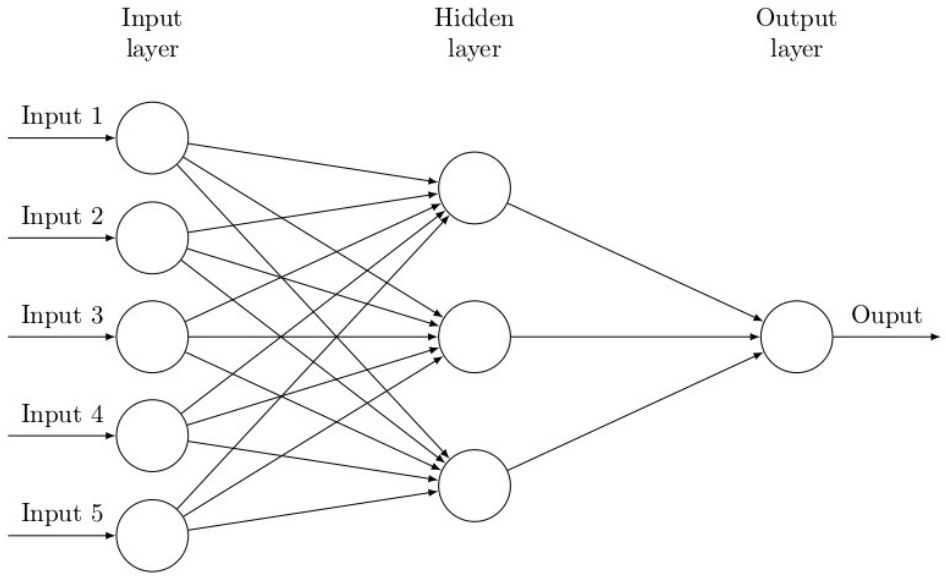

Figure 1.6: Multilayer feedforward neural network with one hidden layer.

The neurons in the hidden layer function primarily to intervene between the external inputs and the network output in a specific way. The network is able to extract higher-order statistics if we add one or more hidden layers. The hidden layers can be adjusted to different degrees of complexity, depending on the desired output. This allows the network to better adapt to new data and extract complex features from the input data [19]. The mechanism of forwarding the signals in a multilayer neural network is started from the neurons of input layer which supply the respective elements of the activation pattern (input vector) to the neurons of the first layer (i.e., the first hidden layer). Further, the output signals of the first hidden layer are forwarded to the neurons of the second hidden layer, and so on. The output signals of the last hidden layer are then sent to the output layer. Finally, the output layer produces the output vector that contains the class of the input vectors. Typically, neurons in each layer of the network receive only the output signals from

the previous layer. The overall response to the activation patterns of the neurons in the input layer is constituted by the set of output signals of the output (final) layer's neurons. Each neuron in the network has its own weights and thresholds, and the final output of the network depends on the combination of these values. By adjusting these weights and thresholds, the network can be trained to recognize patterns and responds to inputs in meaningful ways.

(b) *Feedback or Recurrent network.* RNNs are neural networks that contain feedback

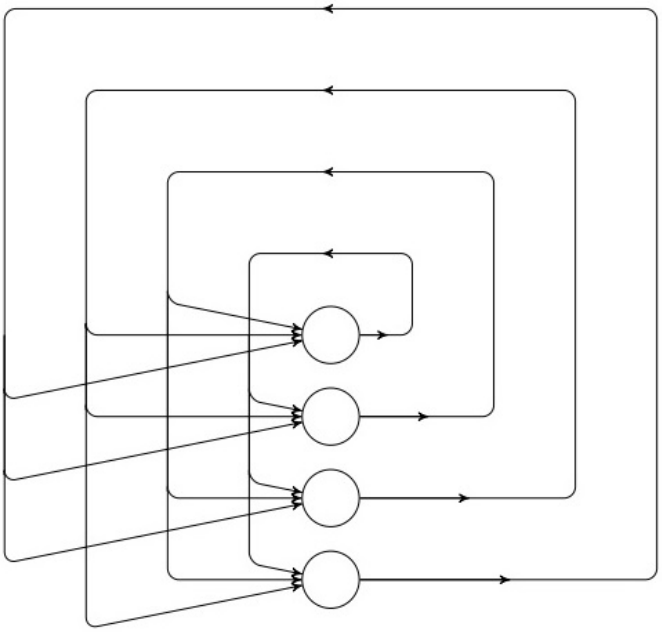

Figure 1.7: A recurrent neural network without a hidden layer

from output signals as inputs. Figure 1.7 depicts a recurrent network of a single layer without a hidden layer. There are four neurons in Figure 1.7, and each of them feeds its output signal back to the inputs of the others. Note that each neuron in the network is not feeding back its output signal to the input of itself, i.e., network does not contain a self-feedback loop. Instead, each output signal is feedback to the

inputs of the other three neurons. This creates a RNNs, which is capable of learning from its environment and making decisions based on its experience. Existing of self-feedback loops in the network has a profound impact on the learning capabilities, and its performance. Another class of RNNs is presented in Figure 1.8, where the network has a single hidden layer.

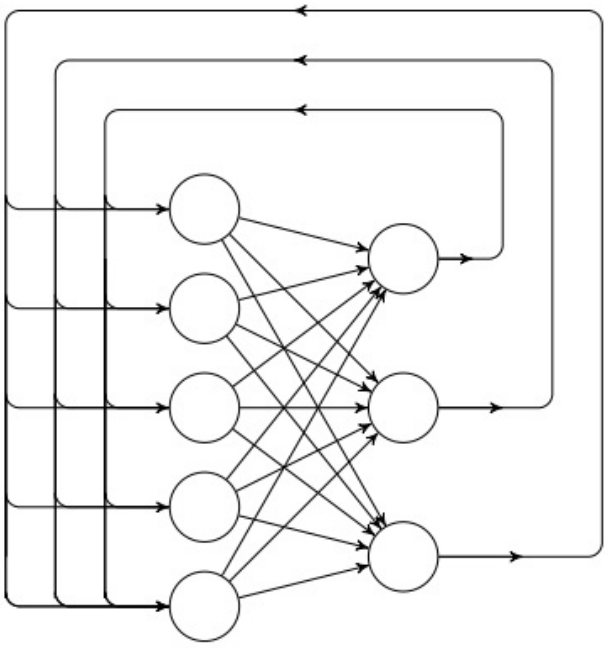

Figure 1.8: A recurrent network with one hidden layer of neurons.

Figure 1.8 shows the feedforward neural network with a single hidden layer. The network is known fully connected if every neuron of each layer in the network is connected to every neuron in the adjacent forward layer. In other case, if some of the synaptic connections from the network is missing then it is called a partially connected network. An example of a partially connected network with one hidden layer is presented in Figure 1.9, where each neuron of the hidden layer has a local field of interconnections with the neurons of input layer.

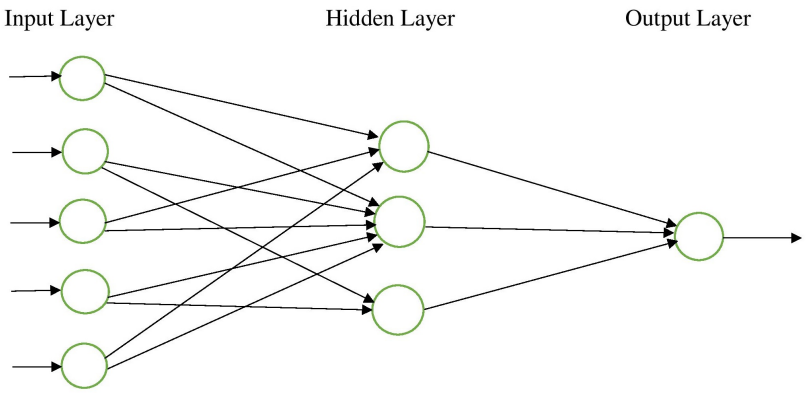

Figure 1.9: Partially connected neural network.

From the above discussions, it is concluded that the performance of a neural network is affected by the following important factors.
(1) External inputs, (2) Activation functions, (3) Internal decay rates,
(4) Propagation delays, (5) Synaptic weights, (6) Connections topology/Network architecture.

In deterministic models of neural networks, the future levels of activation and synaptic coupling coefficients can be calculated if their initial counterpart is known and all other factors are assumed as constants. Therefore, these models can predict the behavior of the neural networks in given situations. However, they do not take into account the effects of random fluctuations in the environment or the dynamics of learning and adaptation. As such, their predictive accuracy is limited. This is called the joint activation-weight dynamics. As a network learns, its synaptic weights are adaptively determined to perform some particular task, such as pattern recognition or to obtain some desired network outputs. In the space of matrices of synaptic coupling coefficients, such a scheme determines a discrete (a system of difference equations) or a continuous dynamical system (a system of differential equations).

This is called weight dynamics.

Back-propagation is a common training algorithm used for neural networks. It involves iteratively adjusting the weights of the network based on the error between the predicted and actual outputs, using a gradient descent algorithm. This error is then propagated backward through the network, allowing the weights to be updated so as to reduce the error.

Now in the next section, the famous models of neural networks such as Hopfield model, Complex-valued model, and Memristor-based model have been discussed.

1.3 Some Additive Neural Networks

One way to take into account the time-varying inputs is illustrated in Figure 1.10, where the respective inputs $u_1(t), u_2(t), ..., u_n(t)$ are represented by potentials (i.e., voltages) and synaptic weights $w_{k1}, w_{k2}, ..., w_{kn}$ are represented by conductance (i.e., reciprocal of resistance). The inputs multiplied by their respective synaptic weights are summed up in the current summing junction characterized as low input resistance, unity current gain, and high output resistance. Total current flowing to the

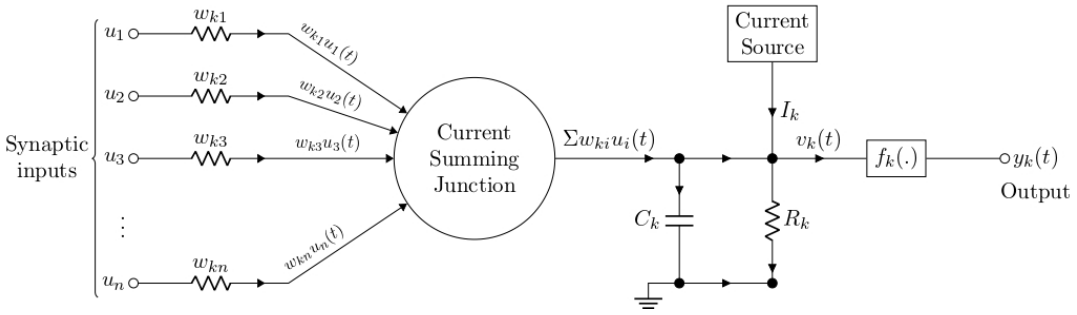

Figure 1.10: Additive model of a neuron.

input node of the activation function $f_k(.)$ is

$$\sum_{i=1}^{n} w_{ki} u_i(t) + I_k, \tag{1.3.1}$$

where the first term is due to the input signals of k-th neuron and the second term is due to the external source of current applied as a bias to the neuron. Let $v_k(t)$ denote the induced local field at the input node of the activation function, then the total current flowing away from the input node of the activation function is as follows:

$$\frac{v_k(t)}{R_k} + C_k \frac{dv_k(t)}{dt}, \tag{1.3.2}$$

where the first term presents the current across the resistor R_k and the second term presents the current due to potential drop across the capacitor C_k. According to the Krichoff's current law, the total current flowing through the input node of the activation function given in Figure 1.10 is zero. Thus, from the equations (1.3.1) and (1.3.2), we have

$$\frac{v_k(t)}{R_k} + C_k \frac{dv_k(t)}{dt} = \sum_{i=1}^{n} w_{ki} u_i(t) + I_k. \tag{1.3.3}$$

Given the induced input field $v_k(t)$, the output of the neuron k is determined by the activation function as

$$u_k(t) = f_k(v_k(t)). \tag{1.3.4}$$

The activation function in the additive model is generally chosen to be bounded and differentiable with asymptotic behavior like logistic function shown in Figure 1.4(c)

as

$$f_k(v_k) = \frac{1}{1 + exp(-v_k)}. \tag{1.3.5}$$

The model described by the differential equation (1.3.3) is known as an additive model of a neuron, where the name "additive" is used to discriminate it from the multiplicative (shunting) models which contain state dependent synaptic weights [20].

1.3.1 Hopfield Neural Networks

Let us consider a fully connected recurrent network consisting of n neurons with a structure like Figure 1.10. In Figure 1.11, the basic diagram of inter-connections shows each neuron feeding back its output via a delay element to the inputs of the other neurons. More precisely, there is no self feedback loop in the network. Thus,

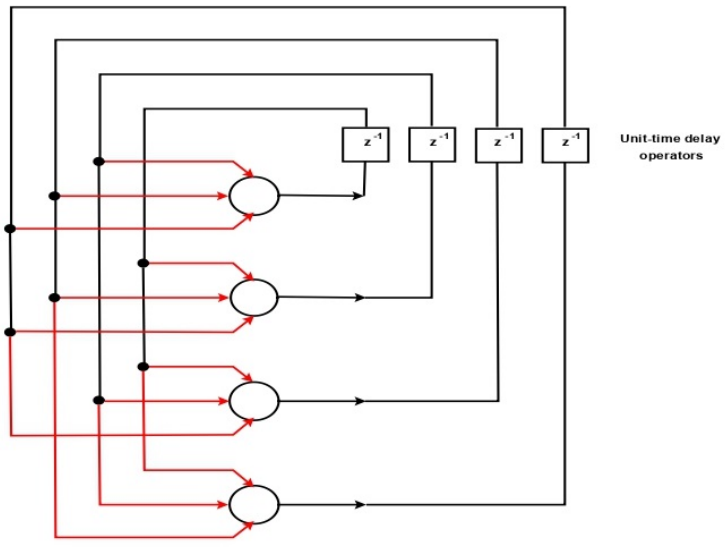

Figure 1.11: Hopfield neural network for n=4 neurons.

considering the instantaneous propagation of signals between the neurons, and the

equation (1.3.3), we get

$$C_k \frac{dv_k(t)}{dt} = -\frac{v_k(t)}{R_k} + \sum_{i=1}^{n} w_{ki} u_i(t) + I_k, \quad for, \quad k = 1, 2, ..., n, \tag{1.3.6}$$

where $u_i(t) = f_i(v_i(t))$, i.e., each neuron has its own activation function. An additive model (1.3.6) represents the Hopfield neural network without a time delay [21]. A Hopefield model typically consists of two types of flow viz., discrete and continuous. Discrete flow processes data in individual units, while continuous flow processes data as a continuous stream. The Hopfield model has been successfully used to solve a variety of optimization problems. The solution of differential equation (1.3.6) is the continuous flow of the Hopfield neural network. The discrete flow of Hopfield network is based on the McCulloch-Pitts model where the state of each neuron has the values -1 and 1 depending on the signs of induced local field, i.e., $u_i(t) = -1$ if $v_i(t) > 0$, and $u_i(t) = 1$ if $v_i(t) < 0, \forall i$. This thesis is based on the continuous flow of neural networks rather than the discrete one.

As an associative neural network, the Hopfield neural network has attracted a great deal of attention [22, 23]. The concept of associative memory can be defined as a content-addressable memory in which patterns are stored in fixed points along a network. A network retrieves patterns stored in a memory based on incomplete or noisy information about those patterns. As a result, retrieving the stored pattern with a reasonable subpart or subset of its information is an important property of a content addressable memory.

The dynamics of Hopfield network was being studied by John Hopfield in his article [21] published in 1984. He had defined an energy function for the network by assuming the following conditions.

(i) The matrix of synaptic weights is symmetric, i.e., $w_{ki} = w_{ik}$ for all i and k.

(ii) There exists an inverse of the activation function (1.3.5), so we may write $v = f_k^{-1}(u)$.

From the equation (1.3.6), we obtain

$$f_k^{-1}(u) = -\ln\left(\frac{1-u}{1+u}\right). \tag{1.3.7}$$

The energy or Lyapunov function for the Hopfield model (1.3.6) is defined as

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} w_{ik}u_i(t)u_k(t) + \sum_{k=1}^{n}\frac{1}{R_k}\int_0^{u_k} f_k^{-1}(u)du - \sum_{k=1}^{n} I_k u_k. \tag{1.3.8}$$

In this subsection, we call it only the energy function because we will discuss the Lyapunov function later. There may be many minima points in the energy function defined in equation (1.3.8), which represent the stable fixed points in the Hopfield model (1.3.6). The dynamics of the network is to find out those minima. The energy function can be used to analyze the dynamical behavior of the Hopfield network. This can provide useful insight into the network's stability and behavior.

Hence, differentiating E with respect to time t and using the equations (1.3.6) and (1.3.7), we obtain

$$\frac{dE}{dt} = -\sum_{k=1}^{n} C_k\left(\frac{du_k}{dt}\right)^2\left[\frac{df_k^{-1}(u_k)}{du_k}\right]. \tag{1.3.9}$$

From the equation (1.3.7), we can see that the inverse activation function is monotonically non-decreasing function of the output x_k. Thus it follows that

$$\frac{df_k^{-1}(u_k)}{du_k} \geq 0, \quad \forall \quad u_k. \tag{1.3.10}$$

Also note that

$$\left(\frac{du_k}{dt}\right)^2 \geq 0, \quad \forall \quad u_k. \tag{1.3.11}$$

Hence, we conclude from the equation (1.3.9) that

$$\frac{dE}{dt} \leq 0. \tag{1.3.12}$$

According to the Lyapunov's method of stability, the inequality (1.3.12) implies that the time-evolution of the continuous Hopfield neural network described by the equation (1.3.6) represents a trajectory, which seeks out the minimum of the energy function E and comes to a stop at such fixed points. It is also observable from the equation (1.3.9), using the inequalities (1.3.10) and (1.3.11), that the inequality (1.3.12) is zero only if

$$\frac{du_k}{dt} = 0, \quad \forall \quad k. \tag{1.3.13}$$

Therefore, we have

$$\frac{dE}{dt} < 0, \tag{1.3.14}$$

except at fixed points.

As a result, the trajectory of the continuous Hopfield neural network converges asymptotically to the fixed points.

1.3.2 Complex-valued Neural Networks

An ANN that processes information with complex-valued variables and parameters is known as a complex-valued neural network (CVNN) [24]. Their advocacy stems from the difference between how complex numbers are represented arithmetically, especially multiplication. Multiplication functions that result in phase rotation and amplitude modulation reduce the degree of freedom in an advantageous way [25]. This allows for the representation of complex numbers in a more efficient and intuitive way. As a result, this reduces the number of operations needed and simplifies the implementation of mathematical algorithms. This is the main advantage of complex numbers in the field of mathematics. A major advantage of ANNs is their self-organization and freedom in learning. CVNNs can minimize a potentially dangerous portion of freedom by knowing a priori the amplitude and phase of data.

It should be noted, however, that many current implementations of artificial neural networks and machine learning frameworks use real numbers instead of complex numbers. It has become increasingly popular to construct ANNs using complex numbers, and to explore the potential advantages of CVNNs over their counterparts with real values.

While the individual components of complex numbers have been treated independently as real numbers in many analyses involving complex numbers, it is erroneous to assume that CVNNs are equivalent to real-valued neural networks in two dimensions. CVNNs are fundamentally different from real-valued neural networks as they are able to capture complex relationships between inputs and outputs, allowing for more sophisticated modeling. Furthermore, in many analyses involving complex numbers, the individual components of the complex number have been treated independently as real numbers, it would be erroneous to apply the same concept to

CVNNs by assuming that a CVNN is equivalent to a two-dimensional real-valued neural network. In fact, it has been shown that this is not the case[26], because the CVNNs' synaptic weighting is dependent on the operation of complex multiplication limits. As a result, phase-rotational dynamics appears to exert a strong influence on the learning process.

In neural networks, activation functions introduce nonlinearity to affine transformations. The model becomes more expressive as a result. Suppose an input $x \in \mathbb{C}^M$ and weights $W \in \mathbb{C}^{N \times M}$ are given, then the output $y \in \mathbb{C}^N$ of any neuron is determined by

$$y = f(x) = Wx,$$

where f is a nonlinear activation-function. It has been shown that neural networks are neural approximators using sigmoid squashing activation functions that are monotonic and bound [27, 28, 29, 30]. The majority of the activation functions, which have been proposed for CVNNs in the literature, are summarized in Table 1.2. There are two types of activation functions: holomorphic and non-holomorphic. Complex functions that are holomorphic everywhere are called "entire functions ". Entire functions are useful as activation functions because they can be applied to any input. Non-holomorphic functions, on the other hand, are limited in their application and are not suitable for use as activation functions. It is not possible to have a complex-differentiable bounded complex activation function in the complex domain. According to Liouville's theorem, all bounded entire functions are constants. Hence, it is impossible to have an CVNN that uses squashing activation functions and is entire. Therefore, the use of a complex-valued neural network (CVNN) with squashing activation functions is limited to being real-valued. This limits the expressive power of CVNNs, as they cannot represent the full range of complex functions.

Activation functions	**Corresponding Publications**
Split-type A	[26, 31, 32]
Split-type B	[25, 26, 33]
Fully Complex (ETF)	[34, 35, 36]
Non Parametric	[37]
Energy Functions	[38, 39]
Complex ReLU	[40]
Nonlinear Phase	[41]
Linear Activation	[42]
Split Kernel Activation	[37]
Complex Kernel Activation	[37]
Absolute Value Activation	[43]
Hybrid Activation	[44]
Mobius Activation	[44]

Table 1.2: Complex valued activation functions.

1.3.3 Memristor based Neural Networks

The use of memristors, currently emerging devices to build nanoscale neural networks, is referred to as the Memristor Based Neural Network.

In 1971 a memristor was proposed to be the fourth essential passive circuit element by Leon Chua [45]. Until in 2008, the Williams group experimentally validated the first solid-state memristor in an HP lab. It was modeled as a thin semiconductor

film (TiO2) sandwiched between two metal contacts. Memristors, which are short for memory-resistors, exhibit a unique memory effect that sets them apart from regular resistors. Applying a bias voltage to a memristor changes its resistance, known as memristance. The memristor's non-volatility is demonstrated by retaining memristance even when power supply is removed.

Researchers are studying memristor-based neural networks as a potential replacement for metal-oxide-semiconductor (CMOS) devices in neuromorphic circuits and exploring their potential applications. Current research mainly concentrates on using memristors as synaptic connections between neurons. However, in any application, it may be possible to allow memristors to perform computation naturally while avoiding additional CMOS devices. This thesis presents examples of neural network methods, including memristor-based structures, memristive spiking-time dependent plasticity (STDP) model, and their potential applications.

The memristor has been found in the nanoscale by several research groups using various materials, including titanium-dioxide-based [46], ferroelectric [47], tungsten-oxide-based [48, 49], and diamond-like carbon-based memristors [50]. Research into the application of the memristor spans across several scientific fields. The memristor is particularly studied for its intrinsic features in artificial neural networks. The memristor has been implemented to reduce the sizes and power consumptions of content-addressable memory (CAM) [51, 52] and ternary CAM (TCAM) [53]. These few examples demonstrate the potential applications of memristors.

Characteristics of Memristor: Memristors are a new type of hardware that offer unique advantages over traditional equipment and are an ideal choice for next-generation storage devices. There are some specific features as follows:

(i) The memristor's input and output are nonlinear, yet continuous, leading to theoretically infinite storage accuracy [54].

(ii) The memristor serves as a fundamental component that can be utilized directly in hybrid circuits.

(iii) The memristor can maintain a new state despite changes in its internal structure while the charge flows through it, making it non-volatile [55].

(iv) The memristor is ideal for neural network due to its small size and low power consumption. Additionally, it works similarly to sensor simulation calculations [56].

1.4 An Overview of Mathematical Concepts

In this section, the mathematical concepts which are applied in the chapters of this thesis have been summarized. The focuses are only on delay differential equations and matrix measure theory.

1.4.1 Delay Differential Equations

The delay differential equation (DDE) is a differential equation in which the argument of the highest order derivative is not less than the argument of the unknown function and its lower order derivatives in the equation. This equation is used to model the behavior of the systems in which the present state of the system depends on the previous state of the system. For example, the delay differential equation is

defined as

$$\dot{u}(t) = u^2(t-5) + u(t), \tag{1.4.1}$$

but

$$\dot{u}(t-6) = u^2(t) + u^3(t), \tag{1.4.2}$$

is not a DDE. Let $\mathcal{C} = C([-\tau, 0], R^n)$ is a Banach space of continuous functions mapping interval $[-\tau, 0]$ into R^n. The norm for the space $\mathcal{C}$ is defined as

$$\|\psi\|_\tau = \sup_{-\tau \le s \le 0} \|\psi(s)\|, \tag{1.4.3}$$

where $\|.\|$ is an Euclidean norm on R^n. Suppose $u(t)$ is a function defined on at least $[t-\tau, t]$, then we can define a new function u_t as

$$u_t(s) = u(t+s), \forall s \in [-\tau, 0]. \tag{1.4.4}$$

For $E \subset R^n$, we construct $\mathcal{C}_E = C([-\tau, 0], E)$ is a Banach space of continuous functions' mapping interval $[-\tau, 0]$ into E.

Definition 1.4.1. Suppose $I \subset R$ and $f : I \times \mathcal{C}_E \to R^n$ is a function, and "." denotes the right hand derivative with respect to time t then the following relation

$$\dot{u}(t) = f(t, u_t), \tag{1.4.5}$$

is a DDE on $I \times \mathcal{C}_E$.

The right hand derivative of a function $u(t) : R \to R$ is defined as

$$\overline{\lim}_{h\to 0^+} \frac{u(t+h) - u(t)}{h}. \tag{1.4.6}$$

For the prescribed $t_0 \in I$ and $\psi_0 \in \mathcal{C}_E$, the initial value problem (IVP) related to the DDE (1.4.5) is

$$\begin{cases} \dot{u}(t) = f(t, u_t), t > t_0, \\ u(s) = \psi_0(s - t_0), \forall s \in [t_0 - \tau, t_0]. \end{cases} \tag{1.4.7}$$

It means that the initial value at t_0 must specify the solution $u(t)$ for the whole past $[t_0 - \tau, t_0]$. Note that the initial function or history ψ_0 is a continuous function but it is not necessarily compatible with the DDE (1.4.7). Thus the solution might not be differential at initial instant t_0. For example, take the simple model of DDE given as

$$\begin{cases} \dot{u}(t) = u(t-1), t > 0, \\ u(s) = 1, \forall s \in [-1, 0]. \end{cases} \tag{1.4.8}$$

$u(t) = t+1$ is the solution of DDE (1.4.8) for interval $t \in (0, 1]$ with initial condition $u(0) = 1$. From the solution conclude that $\dot{u}(0^+) = 1 \neq \dot{u}(0^-) = 0$. It means that first derivative of $u(t)$ at $t_0 = 0$ is not continuous. The equation (1.4.5) is a general form of the following differential equations.

1. When $\tau = 0$, then differential equation

$$\dot{u}(t) = f(t, u), \tag{1.4.9}$$

is ordinary differential equation.

2. DDE with discrete delays defined as

$$\dot{u}(t) = f(t, u(t), u(t-\tau_1), ..., u(t-\tau_n)), \ \ where\ \tau = \max_{1\le i\le n} \tau_i. \tag{1.4.10}$$

3. DDE with distributed delay given as

$$\dot{u}(t) = \int_{-\tau}^{0} f(t, s, u(t+s))ds. \tag{1.4.11}$$

When delay is infinity, then DDE with infinity delay describe as

$$\dot{u}(t) = \int_{-\infty}^{0} f(t, s, u(t+s))ds, \tag{1.4.12}$$

associated with the history function $u(s) = \psi_0(s - t_0), \forall s \in (-\infty, t_0]$.

Now we define the solutions of DDE (1.4.5).

Definition 1.4.2. Let $t_0, \delta \in R$ with $\delta > t_0$ such that $u \in C([t_0-\tau, \delta), E), [t_0, \delta) \subset I$, and $u(t)$ satisfies the equation (1.4.5) for $t \in [t_0, \delta)$. Then the function $u(t)$ is a solution of the equation (1.4.5) on $[t_0 - \tau, \delta)$.

In an ODE, the solution of IVP (1.4.7) is equivalent to find the solution of the following integral equation.

$$\begin{cases} u(t) = \psi_0(t_0) + \int_{t_0}^{t} f(s, u_s)ds, \ \forall\ t \in [t_0, \delta), \\ u_{t_0}(s) = \psi_0(s), \quad \forall s \in [t_0 - \tau, t_0]. \end{cases} \tag{1.4.13}$$

The concept of existence and uniqueness, is similar to the nonlinear ODE [57]. We will introduce the theorems on existence and uniqueness of DDE with finite delay. The detailed proofs can be seen in [58]. Before stating the theorems, the following definition is required.

Definition 1.4.3. Let $f : I \times \mathcal{C}_E \to R^n$ and $S \subset I \times \mathcal{C}_E$. Then f satisfies Lipschitz condition on S if there exists $K > 0$ such that

$$\|f(t, \psi_1) - f(t, \psi_2)\| \leq K\|\psi_1 - \psi_2\|_\tau, \tag{1.4.14}$$

whenever $(t, \psi_1), (t, \psi_2) \in S$.

Theorem 1.1. *[57](Local Existence) Let D be an open subset in $R \times \mathcal{C}$, and $f : \omega \to R^n$ is continuous on D. If $(t_0, \psi_0) \in D$, then there is a solution of the IVP (1.4.7) passing through (t_0, ψ_0) that contained in $[t_0 - \tau, t_0 + \nu]$ for some $\nu > 0$.*

Theorem 1.2. *[57](Uniqueness) Let D be an open subset in $R \times \mathcal{C}$ and $f : \omega \to R^n$ is continuous function which satisfies Lipschitz condition on each compact set in D. If $(t_0, \psi_0) \in D$, then there exists a unique solution to the IVP (1.4.7) passing through (t_0, ψ_0).*

In neural networks, time-delay plays a key role in dynamical behavior. As discussed in the previous section, the neuron transmit signals are an instantaneous process, but in practical, it is not always true. There are different shapes and sizes of axons due to which signal transmission from one neuron to others is not instantaneous [59]. Taking these facts into account, the additive Hopfield model represented by the equation (1.3.6) can be transformed into the following DDE.

$$C_k \frac{dv_k(t)}{dt} = -\frac{v_k(t)}{R_k} + \sum_{i=1}^{n} w_{ki} f_i(v_i(t - \tau_{ki})) + I_k, \quad for \quad k = 1, 2, ..., n, \tag{1.4.15}$$

where τ_{ki} is the time taken by the signals to transmit from i-th to k-th neurons in the network. This type of delay is called discrete delay. Since neural networks have a spatial structure due to the presence of many parallel pathways connected among the neurons it is not possible to model those with discrete delay. Such networks

can be better modeled through a continuous delay which takes into account the temporal dynamics of the neurons. Continuous delay can account for the changing connection strengths, delays, and weights over time. There will be a distribution of propagation time delay. Thus, it is desirable to introduce continuously distributed delays in order to model those [60, 61, 62, 63, 64, 65]. The equation (1.3.6) can be represented in the form of DDE with distributive delays as

$$C_k \frac{dv_k(t)}{dt} = -\frac{v_k(t)}{R_k} + \sum_{i=1}^{n} w_{ki} f_i\left(\int_0^{\infty} v_i(t-u) g_{ki}(u) du \right) + I_k, \quad for \quad k = 1, 2, ..., n, \tag{1.4.16}$$

where u is a signal delay from i-th to j-th neuron which occurs with probability distribution function $g_{ki}(u)$ with mean delay $\tau_{ki} = \int_0^{\infty} u g_{ki}(u) du$. A DDE that contains discrete and distributed delays is known as a mixed-time delay DDE. The time derivative of state variable v_k depends on v_i for the past $(-\infty, t]$. There are literature involving results on delayed neural networks, which have been published in the recent past [66, 67, 68, 69, 70, 71].

1.4.2 Matrix Measure Theory

In order to analyze the stability and synchronization of neural networks, there are many algebraic methods available such as Linear Matrix Inequality (LMI)[72, 73], M-Matrices [74], H-Matrices [75], and Matrix measure theory [76, 77, 78], etc. In other words, we can say that the concept of measuring a matrix derived from the concept of normed linear space. A norm is a function that assigns a length or size to each vector in a vector space. It defines the concept of distance in linear space, which in turn, allows us to measure a matrix.

Let us consider R^n is a finite dimensional Euclidean linear space over real field R.

The norm $\|(.)\|_p$ in R^n for $1 \leq p < \infty$ is defined as

$$\|u\|_p = (\sum_{i=1}^{n} |u_i|^p)^{\frac{1}{p}}. \tag{1.4.17}$$

The normed linear space is linear space R^n associated with the norm defined in (1.4.17) . Now define the following norm for $p = 1, 2, \infty$.

$$\|u\|_1 = \sum_{i=1}^{n} |u_i|, \quad \|u\|_2 = (\sum_{i=1}^{n} |u_i|^2)^{\frac{1}{2}}, \quad \|u\|_\infty = \max_{1 \leq i \leq n} |u_i|, \tag{1.4.18}$$

for any $u \in R^n$.

Similarly, we can define the matrix norm in linear space $R^{n \times n}$ over the real field R. Thus, for any matrix $M = [m_{ij}]_{n \times n} \in R^{n \times n}$, we can get the following matrix norms as

$$\|M\|_\infty = max_i \sum_{j=1}^{n} |m_{ij}|, \|M\|_1 = max_j \sum_{i=1}^{n} |m_{ij}|, \|M\|_2 = \sqrt{\lambda_{max}(M^T M)}, \tag{1.4.19}$$

for $p = 1, 2, \infty$.

Definition 1.4.4. The function $\|(.)\| : R^{n \times n} \to R^+$ is uniformly continuous and convex in nature. The one-sided directional derivative of the norm function $\|(.)\|$ at point $I \in R^{n \times n}$ in the direction of M is called the matrix measure of M and it is denoted by $\mu(M)$, and defined by

$$\mu(M) = \lim_{h \to 0^+} \frac{\|I + hM\| - 1}{h}. \tag{1.4.20}$$

Let us show the existence of limit of the function $f(h) = \frac{\|I+hM\|_p - 1}{h}$. If $f(h)$ is decreasing w.r.t h and bounded below then the limit of function (1.4.20) must exists

for all $M \in R^{n\times n}$. Let $k \in (0,1)$, then we get

$$\begin{aligned} khf(kh) = \|I + khM\| - 1 =& \|k(I + hM) + (1-k)I\| - 1 \leq k\|I + kM\| + 1 - k - 1 \\ \leq& k(\|I + hM\| - 1). \end{aligned} \tag{1.4.21}$$

That is $f(kh) \leq f(h)$ implies $f(h)$ is decreasing w.r.t. h. Note that $f(h) \geq -\|M\|$. Hence the existence of the limit in (1.4.20) is proved. The matrix measure of the matrix M over the norm $\|(.)\|_p$ for $p = 1, 2, \infty$ is defined as

$$\mu_p(M) = \lim_{h\to 0^+} \frac{\|I + hM\|_p - 1}{h}. \tag{1.4.22}$$

Thus we have the following matrix measures as

$$\begin{aligned} \mu_1(M) &= max_j \left\{ m_{jj} + \sum_{i=1, i\neq j}^{n} |m_{ij}| \right\}, \\ \mu_2(M) &= \frac{1}{2}\lambda_{max}(M^T + M), \\ \mu_\infty(M) &= max_i \left\{ m_{ii} + \sum_{j=1, j\neq i}^{n} |m_{ij}| \right\}. \end{aligned}$$

Example 1.4.1. If matrix $M \in R^{n\times n}$ is a skew-symmetric and $M \neq 0$, then $\mu_2(M) = 0$.

Lemma 1.3. *In the view of Definition 1.4.4, let $M, N \in R^{n\times n}$, then the matrix measure $\mu_p(.)$ has the following properties:*

(i) $-\|M\|_p \leq \mu_p(M) \leq \|M\|_p, \forall \;\; M \in R^{n\times n}$.

(ii) $\mu_p(\alpha M) = \alpha\mu_p(M), \forall \alpha > 0, \forall \;\; M \in R^{n\times n}$.

(iii) $\mu_p(M + N) \leq \mu_p(M) + \mu_p(N), \forall \;\; M, N \in R^{n\times n}$.

(iv) $max[\mu(M) - \mu(-N), -\mu(-M) + \mu(N)] \leq \mu(M + N) \leq \mu(M) + \mu(N)$.

(v) $\mu : R^{n\times n} \to R$ *is convex on* $R^{n\times n}$*, i.e.,*

$$\mu[\lambda M + (1-\lambda)N] \leq \lambda\mu(M) + (1-\lambda)\mu(N), \quad \forall\lambda \in (0,1).$$

(vi) $|\mu(M) - \mu(N)| \leq |\mu(M-N)| \leq \|M-N\|$.

(vii) $-\mu(-M) \leq Re\lambda_i(M) \leq \mu(M)$ *for all* $i \in \{1,2,...,n\}$.

(viii) If M *is non-singular, then* $-\mu(-M) \leq (\|M^{-1}\|)^{-1} \leq \|M\|$.

Proof. In [78], the matrix measure is shown to possess the above properties. □

In order to analyze neural network stability, matrix measure method offers the following advantages:

1. The matrix measure can be zero/negative/positive, but the norm is always positive.

2. Constructing Lyapunov function for the system is a hard task due to the lack of general methods, but matrix measures allow us to construct Lyapunov functions easily.

3. The matrix measure can help us to identify the stability of the neural network in a much simpler and straightforward manner.

4. It is also useful for detecting and evaluating the stability of a system with multiple inputs and outputs.

In view of these advantages, many authors have focused their attention on the stability analysis of neural networks via the matrix measure method [76, 77, 78, 79, 80].

This analytical method has shown to be efficient for the analysis of dynamical systems with a wide range of parameters. It has been widely applied in engineering and physics, and has been proven to be an effective tool for the analysis of neural networks.

1.5 Synchronization

Chaos is an important characteristic of nonlinear dynamical systems. Chaos occurs when the trajectory of a nonlinear system is highly sensitive to its initial conditions. This means that small variations in the initial conditions of the system can lead to significantly different outcomes over time. In other words, chaotic systems are unpredictable. Many researchers have demonstrated chaotic behavior of neural networks in their articles [81, 82, 83]. Due to their extremely sensitive dependence on initial conditions, chaotic systems are difficult to synchronize. From very close initial conditions, any initial correlation between identical systems exponentially decreases to zero over time. This means that chaotic systems are very hard to predict and control, making them even more unpredictable than non-chaotic systems. This unpredictability is one of the core features of chaotic systems, and makes them both interesting and challenging to study. Therefore, any initial synchronization between the systems is likely to disappear rapidly. However, some methods have recently been proposed for achieving synchronized behavior between chaotic systems. These methods involve coupling the systems through a communication channel. This coupling allows for the exchange of information, allowing the systems to synchronize their behavior. However, this synchronization is only temporary and must be constantly updated. The concept of a response system locking onto a driver system

has been pioneered by Pecora and Carroll [84]. Until now, these studies have focused on driving a response system with a single driver. The information learned from investigating such basic systems might not be sufficient to predict how complex systems made up of multiple independent driving systems that compete with one another to synchronize the same response system would behave. This is because the complexity of the interactions between the drive and response systems is not easily predictable. To gain insights into such complex systems, experimentation is key. Advanced mathematical modelling and computer simulation can also help to understand the dynamics of the system. In a network of chaotic elements, the Pecora-Carroll driving mechanism has the 'strong-coupling' limit of a general scheme of directionally-oriented coupling.

There are many types of synchronization which have been reported in literature [85, 86, 87]. Some of those are described below.

Definition 1.5.1. Let us assume $f : R^n \to R^n$ is a continuous vector field and $V(u, v)$ is a coupling term. Then two coupled identical systems as given by

$$\dot{u}(t) = f(u(t)), \tag{1.5.1}$$

$$\dot{v}(t) = f(v(t)) + V(u(t), v(t)), \tag{1.5.2}$$

where $u, v \in R^n$, are said to be synchronized completely if $\|v(t) - u(t)\| \to 0$ as $t \to \infty$.

Definition 1.5.2. Let us consider the coupled systems as

$$\dot{u}(t) = f(u(t)), \tag{1.5.3}$$

$$\dot{v}(t) = g(v(t)) + V(u(t), v(t)), \tag{1.5.4}$$

where $f, g : R^n \to R^n$ is a continuous vector field and $V(u, v)$ is a coupling term. Now define the error system as

$$e(t) = v(t) - \alpha u(t), \tag{1.5.5}$$

where $\alpha \neq 0$ is projective constant. Then the systems (1.5.3) and (1.5.4) are said to be quasi-projective synchronized if there exists $T > 0$ such that $e(t) \in D = \{e(t) : \|e(t)\| < \epsilon\}$ for all $t > T$, where ϵ is a synchronization error bound.

Remark 1.4. There are some another type of synchronization follows as:

(i) In general, if $\alpha = 1$, then the error system (1.5.5) is quasi-synchronized.

(ii) If $\alpha = -1$, then the error system (1.5.5) is quasi-anti synchronized.

(iii) If α is replaced by a diagonal matrix $\Omega = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ with constant $\alpha_i \neq 0, \forall i$, then it is called modified projective synchronization.

(iv) If α is replaced by a diagonal matrix $\Omega(t) = \{\alpha_1(t), \alpha_2(t), ..., \alpha_n(t)\}$ with constant $\alpha_i(t) \neq 0$, is bounded and continuously differentiable function for all i, then it is called modified function projective synchronization.

(v) If $\alpha_1(t) = \alpha_2(t) = ... = \alpha_n(t)$, then it is called the function projective synchronization.

In the literature, there have also been reports of various synchronization scheme kinds, including anti-synchronization, phase synchronization, generalized synchronization, see [88, 89, 90] and the references listed therein.

1.6 Summary of the thesis

In this thesis, four different problems are presented in which complex-valued recurrent neural network models are considered. All the problems are mainly focused on the matrix measure method. The first problem is related to the global exponential synchronization of said model in the presence of uncertainty and bounded and unbounded delays. Based on Halanay inequality and matrix measure approach, global exponential synchronization is studied for two cases. The first case is the synchronization of CVRNNs in the presence of uncertain parameters with time-varying bounded and unbounded delay terms and the second one is the concerned synchronization in the absence of uncertain terms with same bounded and unbounded time-varying delay terms. The second problem is extended to the global quasi-synchronization of CVRNNs with interaction terms with time-varying delay. When the scaling factor consists of a unit constant then the synchronization scheme becomes global quasi synchronization. Since time delays in the neural network may affect their dynamics, the CVRNNs models are considered with time-varying delays. It is shown that global quasi-synchronization of CVRNNs can not be completed, i.e., the trajectory of the error system fluctuates in a small compact domain of radius equal to synchronization error bound. Thus, we call this type of synchronization a global quasi-synchronization. A study of quasi-projective synchronization on memristor-based CVRNNs model has been presented in the third problem. Here, the quasi-projective synchronization is presented in two ways based on the Halanay inequality and some useful techniques. Some novel sufficient conditions are firstly derived by using the matrix measure method. And then, by using the Lyapunov-Krasovskii function, another sufficient stabilization condition for quasi-projective synchronization of the concerned model is presented. Again, in the fourth problem, the same synchronization analysis on a second-ordered model of CVRNN is studied.

By using an appropriate variable transformation, the order of the ICVRNN differential system is reduced, and then, by applying the real decomposition method, it is separated into real and imaginary components. The matrix measure approach with the nonlinear Lipschitz activation functions is employed in the ICVRNN model.

****