1-1-2023

Bio-Inspired Memory Device Based Physical Reservoir Computing System to Solve Temporal and Classification Problems

Md Razuan Hossain
University of Mississippi

BIO-INSPIRED MEMORY DEVICE BASED PHYSICAL RESERVOIR COMPUTING SYSTEM TO SOLVE TEMPORAL AND CLASSIFICATION PROBLEMS

A Dissertation
presented in partial fulfillment of requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
The University of Mississippi

by

Md Razuan Hossain

August 2023

ABSTRACT

In recent years, technological advancements in the field of computing have been limited by the slowing down of Moore's law, which predicts the doubling of the number of transistors on a microchip every two years. The limitations of traditional solid-state electronics have become increasingly evident, as it is becoming difficult to increase the number of transistors while maintaining their reliability and performance. The breakdown of Dennard Scaling, which describes the power-density relationship in CMOS transistors, and the von Neumann bottleneck, which refers to the limited bandwidth between the central processing unit (CPU) and memory, have further exacerbated the situation. Moreover, the ever-increasing computing demands have led researchers to explore alternative computing mechanisms, such as neuromorphic computing.

In this context, there is a growing interest in developing novel memory devices that possess non-linear current-voltage characteristics and inherent memory properties, such as bio-inspired memory devices. These devices are inspired by the functioning of biological synapses in the brain, which enable neurons to communicate with each other and form complex networks. Bio-mem devices use biomolecules, such as proteins or DNA, as their active material and can exhibit a range of non-linear behaviors, including hysteresis, threshold switching, and negative differential resistance. Additionally, they have inherent memory properties that allow them to retain their state even after the power is turned off.

The unique features of bio-mem devices make them a promising candidate for solving classification and temporal pattern recognition tasks. Unlike their solid-state counterparts, bio-mem devices feature a similar structure, switching mechanism, and ionic transport modality as biological synapses, while consuming considerably lower power. The use of bio-mem devices in computing can lead to the development of more efficient and powerful computing systems that can handle complex tasks more effectively.

One promising computing paradigm that has emerged in recent years is Reservoir Computing (RC), which is a type of recurrent neural network that uses a fixed, random, and sparse connectivity matrix called a ”reservoir.” The reservoir provides a non-linear mapping of the input data, and the output is obtained by training a linear readout layer using a simple learning algorithm, such as ridge regression or support vector regression. RC has been shown to be effective in solving a range of classification and temporal problems, such as speech recognition, time-series prediction, and image classification. This work aims to explore the use of biomolecular devices in RC to solve classification and temporal problems, which could lead to the development of more efficient and powerful computing systems.

DEDICATION

To my loving family, who has been my constant support and source of inspiration throughout my academic journey. Thank you for always believing in me, encouraging me to pursue my dreams, and providing me with the love and emotional support I needed to overcome the challenges of doctoral research. Your unwavering commitment to my success has been the driving force behind my achievements, and I am grateful for your presence in my life.

To my thesis advisor, Md Sakib Hasan, who has been a mentor, guide, and friend. Your expertise, guidance, and encouragement have been invaluable to me, and I could not have completed this research without your support. Your patience, insights, and feedback have pushed me to become a better researcher, and I am grateful for the opportunity to work with you.

To my colleagues and friends, who have made this journey enjoyable and memorable. Thank you for the camaraderie, laughter, and support that we shared during this challenging but rewarding experience. Your enthusiasm, feedback, and encouragement have significantly impacted my research and personal growth.

Finally, to all the people who have contributed to my academic and personal growth, directly or indirectly, thank you. Your support, guidance, and encouragement have been invaluable, and I am grateful for the role you played in my success. This dissertation is dedicated to you all.

ACKNOWLEDGEMENTS

Completing a Ph.D. is not a solitary task, and I am deeply grateful to the many people who have helped me along the way.

First and foremost, I would like to express my sincere gratitude to my thesis advisor, Md Sakib Hasan, for her invaluable guidance, support, and encouragement. His vast knowledge, keen insights, and unwavering commitment to excellence have inspired me to push the boundaries of my research and achieve the highest possible standards.

I would also like to thank my dissertation committee members for their expert feedback and critical insights. Their constructive criticism, challenging questions, and encouragement have helped me to refine my research and strengthen my arguments.

I am grateful to the faculty and staff at the University of Mississippi, who have created a supportive and intellectually stimulating academic environment. The collaboration, networking, and professional development opportunities have enriched my research and broadened my horizons.

My deepest gratitude goes to my family and friends, who have been my constant source of love, encouragement, and emotional support. Their unwavering belief in me has sustained me through the ups and downs of this challenging journey. I am also grateful to my wife, Fowzia, whose unwavering support, patience, and encouragement have been the bedrock of my success.

I would also like to thank Dr. Joseph Najem for his continuous support and guidance. Also, I want to say gratitude to my colleagues Nicholas Armendarez, Ahmed Salah Mohamed Kassem, Anurag Dhungel, Maisha Sadia, and Partha Sarathi Paul for their assistance in my research. I am also thankful to Ashiq Adnan, Sudipto Saha, and Md Faisal Kabir for their support and assistance during my Olemiss journey.

To everyone who has contributed to my academic and personal growth, directly or indirectly, thank you. This dissertation would not have been possible without you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 Motivation

In recent years, traditional computing architectures have evolved into sophisticated systems capable of large-scale parallel computation using billions of transistors. However, the demand for higher speed and lower power consumption has presented challenges to these multi-core architectures. One of the major challenges is the slowing down of Moore's law, which states that the number of transistors on a chip doubles approximately every two years, increasing computing power [55].

The slowing down of Moore's law has been attributed to several factors, including the physical limitations of transistor miniaturization and the increasing difficulty of improving the performance of multicore architectures. As a result, researchers have explored alternative approaches to improving computing performance, such as developing new architectures and using novel materials. For example, novel materials such as graphene and carbon nanotubes have been proposed to improve computing performance by enabling faster switching speeds and reducing power consumption [71] [154].

Moreover, researchers have also explored the use of neuromorphic computing, which is inspired by the structure and function of the human brain. This approach involves the use of artificial neural networks to perform computing tasks, which can be more energy-efficient and capable of parallel processing compared to traditional computing architectures [1].

Despite these efforts, it remains a challenge to maintain the performance improvement rate seen in earlier years. Nevertheless, the development of new computing architectures and materials and the exploration of alternative approaches, such as neuromorphic computing, may offer promising solutions to these challenges in the future. One promising technology that has emerged in recent years is the field of nanotechnology, which involves the design and manufacture of materials at the nanoscale level. Nanotechnology holds tremendous promise for developing novel materials and architectures that can significantly improve computing performance. For instance, researchers are exploring the use of nanophotonics, which uses light instead of electricity to transmit data, to develop faster and more energy-efficient computing systems.

To conclude, the slowing down of Moore's law has presented significant challenges to traditional computing architectures. However, researchers are exploring alternative approaches to enhance computing performance, such as developing new materials and architectures and exploring neuromorphic computing and quantum computing. Nanotechnology also holds tremendous promise for developing novel materials and architectures that can significantly improve computing performance. With these alternative approaches, it is possible to overcome the limitations of traditional computing and pave the way for a new era of computing.

1.2 Artificial Neural Network

Artificial intelligence (AI) and artificial neural networks (ANNs) are closely correlated as ANNs are a key component of AI. ANNs are a subset of machine learning algorithms that are inspired by the structure and function of the human brain. ANNs are a type of machine learning algorithm that are inspired by the structure and function of the human brain. ANNs are designed to learn patterns and relationships from data and use this knowledge to make predictions or decisions.

A typical ANN consists of multiple layers of interconnected neurons, with each layer performing a specific function. The input layer receives the input data, which is then passed through one or more hidden layers, and finally to the output layer, which produces the final

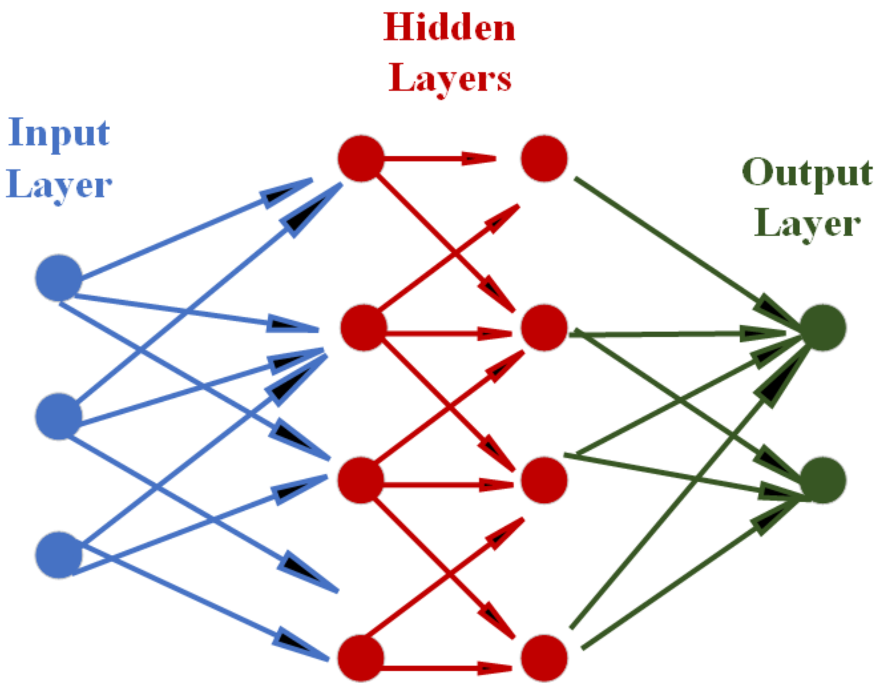


Figure 1.1: Schematic of Feed Forward Network.

output [38]. The diagram below shows a simple feed-forward neural network, which is the most common type of ANN.

In this diagram (1.1), the circles represent neurons, and the lines connecting them represent the connections between the neurons. There are three layers: the input, hidden, and output layers. Each neuron in the hidden layers and the output layer performs a weighted sum of the inputs it receives and applies an activation function to the sum to produce its output. The weights are learned during the training process, which involves adjusting the weights to minimize the difference between the network's output and the desired output.

Overall, the ANN learns to map input data to output data by adjusting the weights of its connections in response to training data. Once trained, the ANN can be used to predict new input data.

ANNs can be classified into different types based on their architecture, learning methods, and application areas. Below are the most common types of ANNs:

Feedforward Neural Networks (FFNN): Feedforward neural networks [145] are the simplest and most common type of neural network. They are also known as multilayer perceptrons (MLPs). These networks consist of multiple layers of neurons, with each neuron in one layer connected to all neurons in the next layer. The input data is fed into the first layer, and the output is obtained from the last layer. FFNNs are used for both classification and regression tasks.

Recurrent Neural Networks (RNN): Recurrent neural networks [151] are a type of neural network where the output of one neuron is fed back into the network as input to another neuron. These networks are particularly useful for processing sequential data, such as time series data, natural language processing, and speech recognition.

Convolutional Neural Networks (CNN): Convolutional neural networks [178] are designed to process data with a grid-like structure, such as images and videos. These networks use convolutional layers to extract features from the input data, followed by pooling layers to reduce the dimensionality of the data. CNNs are widely used in image classification, object detection, and facial recognition.

Self-Organizing Maps (SOM): Self-organizing maps [72] are a type of unsupervised neural network used for clustering and dimensionality reduction. SOMs use a grid of neurons that are arranged in a two-dimensional space. Each neuron in the grid is connected to the input data, and the neurons closer to each other are more likely to be activated together. SOMs are widely used in data visualization, image compression, and feature extraction.

Radial Basis Function Networks (RBF): Radial basis function networks [112] are neural networks that use radial basis functions as activation functions. These networks consist of an input layer, a hidden layer with radial basis functions, and an output layer. RBF networks are used for function approximation, classification, and regression tasks.

Deep Neural Networks (DNN): Deep neural networks [183] are neural networks with many hidden layers. DNNs are used for complex tasks, such as speech recognition, image recognition, and natural language processing. DNNs require a large amount of training data and computational resources to achieve high accuracy.

In summary, artificial neural networks are versatile machine learning algorithms that can be applied to a wide range of applications. The type of neural network used depends on the nature of the problem, the type of input data, and the desired output.

A literature review on ANNs has been given in the following:

The history of artificial neural networks can be traced back to the 1940s, when Warren McCulloch and Walter Pitts proposed the first mathematical model of a neural network. However, it was not until the 1980s and 1990s that ANNs gained popularity due to advancements in computing power and the availability of large datasets. Since then, ANNs have been used in a wide range of applications, from image and speech recognition to medical diagnosis and financial forecasting.

Artificial neural networks consist of interconnected nodes, called neurons, which are organized in layers. Each neuron receives input from other neurons and computes an output using an activation function. The output of one layer of neurons is used as input to the next layer until the final output layer produces a prediction or decision. The process of adjusting the weights of the connections between neurons to optimize the network's performance is called training.

There are several types of ANNs, including feedforward neural networks, recurrent neural networks, convolutional neural networks, and deep neural networks. Feedforward neural networks are the simplest type of ANN, where information flows only in one direction from input to output. On the other hand, Recurrent neural networks allow information to be stored and processed over time, making them suitable for applications such as speech recognition and natural language processing. Convolutional neural networks are commonly used for image and video recognition, while deep neural networks are a type of neural network with multiple hidden layers that are used for complex tasks such as autonomous driving.

The applications of ANNs are numerous and varied. One of the most well-known applications is image recognition, where ANNs are used to identify and classify objects in images. ANNs are also used in speech recognition, natural language processing, and machine translation. In the medical field, ANNs are used for medical diagnosis, drug discovery, and personalized medicine. ANNs are also used in financial forecasting, fraud detection, and predictive maintenance in various industries. Some scholarly work on ANNs has been explored in the following:

Laith Alzubaidi et al. (2021) [5] describe a deep learning approach for image classification using Convolutional Neural Networks (CNNs). Rodrigo Neves (2018) [108] provided an overview of deep learning techniques for time series forecasting. The authors discussed various deep learning models such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Convolutional Neural Networks (CNNs) for time series forecasting. The article also discussed the challenges and opportunities of using deep learning for time series forecasting. Pan and Yang (2010) [116] provided a survey on transfer learning for image classification. The authors discussed various transfer learning techniques such as fine-tuning, feature extraction, and domain adaptation for image classification. The article also discussed the advantages and limitations of transfer learning for image classification. Li et al. [80] proposed an approach for anomaly detection in time series data using Autoencoders. The authors tested their approach on the Yahoo Webscope dataset and achieved an accuracy of 99.63%. The results showed that their approach outperformed traditional machine learning methods for anomaly detection in time series data. Zhang et al. [187] provided a comprehensive survey of deep learning for sentiment analysis. The authors discussed various deep learning models such as RNNs, LSTM networks, and CNNs for sentiment analysis. The article also discussed the challenges and opportunities of using deep learning for sentiment analysis. Zhao et al. [191] provided a review of deep learning techniques for object detection. The authors discussed various deep learning models such as Faster R-CNN, YOLO, and SSD for object detection. The article also discussed the recent developments in deep learning techniques for object detection and their future directions. Zhang et al. (2019) [188] provided an overview of deep learning models for recommendation systems. The authors discussed various deep learning models such as Collaborative Filtering (CF), Matrix Factorization (MF), and Neural Collaborative Filtering (NCF) for recommendation systems. The article also discussed the challenges and opportunities.

1.3 Neuromorphic Computing

Neuromorphic computing is a rapidly emerging field that aims to develop computer architectures and systems that simulate the structure and function of the human brain. This type of computing is designed to provide a new way of processing information that can potentially lead to better and more efficient computing systems. Neuromorphic computing is based on the principles of neuroscience and is inspired by the workings of the brain. This essay will explore the working principle of neuromorphic computing, its popularity, and its applications.

Neuromorphic computing is based on the idea of developing computing systems that are modeled after the human brain. This means that instead of using traditional computing architectures that are based on binary digits, neuromorphic computing systems use networks of artificial neurons that communicate with each other using electrical signals. These networks of artificial neurons are designed to simulate the function of biological neurons in the brain, and the signals they use to communicate are similar to the electrical impulses transmitted by neurons in the brain.

A biological neuron is the fundamental unit of the nervous system. It consists of three main parts: the cell body, the dendrites, and the axon.

Cell body (Soma): The cell body, or soma, contains the nucleus and other organelles that are essential for the neuron's function. The soma processes incoming signals and produces the output signal that is transmitted to other neurons.

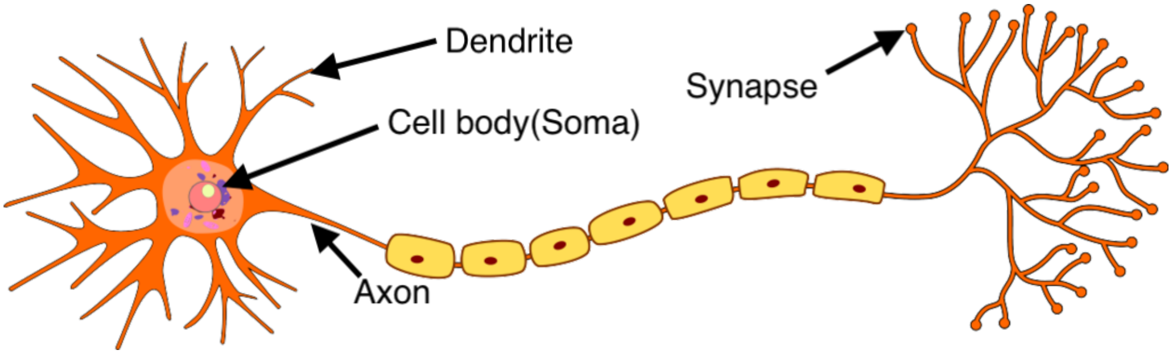


Figure 1.2: Schematic of Biological Neuron. (Image source: Wikimedia Commons.)

Dendrites: Dendrites are branching structures that extend from the cell body. They receive input signals from other neurons or sensory receptors and transmit them to the cell body. Dendrites are covered with tiny projections called spines, which increase the surface area of the dendrites and allow for more input signals to be received.

Axon: The axon is a long, thin projection that extends from the cell body. It carries the output signal, or action potential, from the cell body to the axon terminals. The axon is covered with a myelin sheath, increasing the action potential's speed.

Synapse: The synapse is the junction between two neurons, where the axon terminal of one neuron meets the dendrite or cell body of another neuron. The synapse allows for communication between neurons through the release of chemical messengers called neurotransmitters.

The diagram 1.2 shows a simplified diagram of a biological neuron. In this diagram (Figure 1.2), the cell body is located at the neuron's center and connected to multiple dendrites. The axon extends from the cell body and ends in axon terminals, which form synapses with other neurons. The dendrites receive input signals, which are transmitted to the cell body. If the input signals are strong enough, the cell body produces an output signal, or action potential, which travels down the axon and is transmitted to other neurons through synapses.

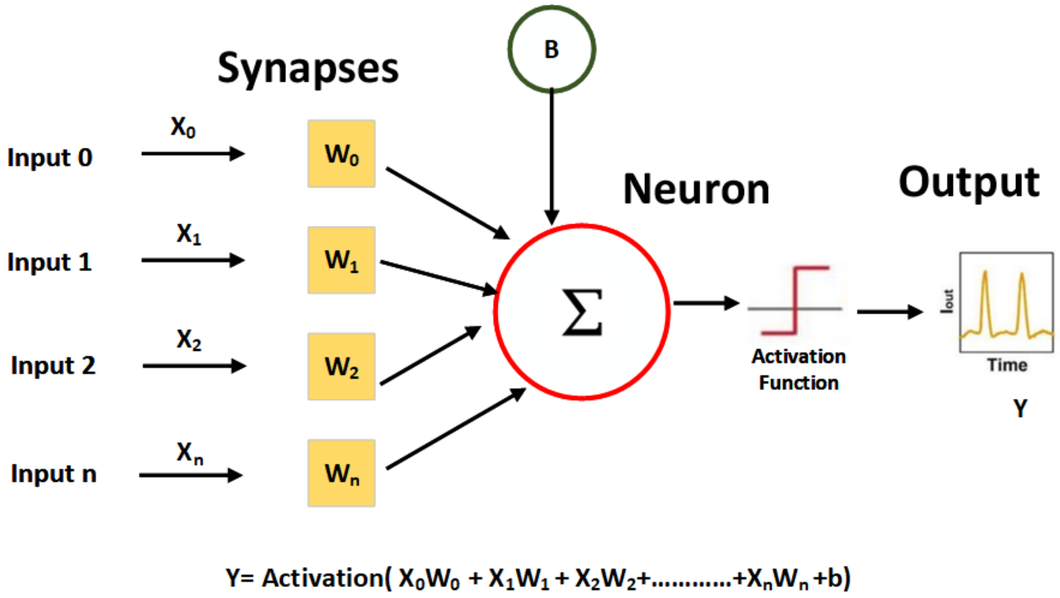


Figure 1.3: Schematic of the equivalent electronic model of the neuromorphic system.

Figure 1.3 shows the electronic model of the neuromorphic system. Inputs are multiplied by synaptic weights and summed up in the neuron layer. After adding the bias, it's triggered by an activation function (sigmoid, relu, step, etc.), and we get the output signal.

Overall, the biological neuron works by receiving input signals, processing them, and producing an output signal. The output signal is then transmitted to other neurons through synapses, allowing for communication between neurons and the function of the nervous system.

Neuromorphic computing can be classified into the following categories:

Spiking Neural Networks (SNNs): Spiking neural networks [40] are a type of neural network that simulate the timing of spikes in biological neurons. These networks use a "pulse" or "spike" to represent the activation of a neuron. SNNs are used for tasks that require temporal processing, such as speech recognition, image processing, and robotics.

Liquid State Machines (LSMs): Liquid state machines [90] are a type of spiking neural network that uses a pool of neurons, known as a "liquid", to process input data. The neurons in the liquid are randomly connected, and their activation states represent the system's current state. LSMs are used for classification, prediction, and control tasks.

Neural Engineering Frameworks (NEFs): Neural engineering frameworks [156] are a type of neuromorphic computing that is designed to bridge the gap between neuroscience and engineering. NEFs use mathematical models of neurons and synapses to design and implement neural networks. NEFs are used for sensory processing, motor control, and decision-making tasks.

Neuromorphic Hardware: Neuromorphic hardware [186] is a computing hardware designed to implement neural networks using analog or digital circuits. Neuromorphic hardware can be used for real-time processing of sensory data, such as vision and audition, and for cognitive computing applications.

Neuroevolution: Neuroevolution [34] is a type of neuromorphic computing that uses evolutionary algorithms to design and optimize neural networks. Neuroevolution is used for game playing, robotics, and optimization tasks.

Memristive Networks: Memristive networks [26] [14] are a type of neuromorphic computing that use memristors, which are resistive devices that can change their resistance based on the history of the applied voltage. These networks are used for tasks such as pattern recognition, signal processing, and memory storage.

Memristive networks have gained popularity in recent years due to their unique properties that make them well-suited for neuromorphic computing applications. Memristors are resistive devices that can change their resistance based on the history of the applied voltage. This allows for creating synapses that can learn and adapt to the input data, making them similar to biological synapses in the human brain.

There are several advantages [180] [19] [85]to using memristive networks for neuromorphic computing:

Energy Efficiency: Memristive networks are highly energy-efficient, which is important for many neuromorphic computing applications. The devices have low power consumption and can be integrated into compact circuits, making them suitable for mobile and embedded systems.

Scalability: Memristive networks can be scaled to large sizes, making them suitable for complex neural network architectures. The devices can be fabricated using standard semiconductor manufacturing processes, which makes them easy to integrate with other electronic components.

Robustness: Memristive networks are robust and can withstand a wide range of operating conditions. The devices are resistant to noise and can operate in harsh environments, which makes them suitable for real-world applications.

Learning and Adaptation: Memristive networks can learn and adapt to input data, making them suitable for pattern recognition and classification tasks. The devices can be used to create synapses that can adjust their strength based on the history of the input signals, which allows for the creation of adaptive networks.

Non-volatile Memory: Memristive networks can be used to create non-volatile memory, which is the memory that retains its contents even when power is turned off. This makes them suitable for applications such as data storage and retrieval.

In summary, memristive networks have gained popularity in recent years due to their unique properties that make them well-suited for neuromorphic computing applications. The devices are highly energy-efficient, scalable, robust, and can learn and adapt to input data, making them suitable for various applications.

Neuromorphic computing systems are designed to be highly parallel, meaning they can process multiple pieces of information simultaneously. This is different from traditional computing systems that are designed to process information in a serial manner, one piece at a time. Neuromorphic computing systems also have the ability to learn from the data that they process, just like the human brain. This means they can adapt to new data and change their behavior over time.

Neuromorphic computing has gained popularity in recent years due to the significant advancements that have been made in the field. These advancements have led to the development of powerful computing systems that can simulate the behavior of biological neurons with high accuracy. This can potentially revolutionize many fields, including artificial intelligence, robotics, healthcare applications, [32] [31], and even medicine.

Neuromorphic computing is also becoming more popular because it offers a more energy-efficient way of processing information. Traditional computing systems are known for their high energy consumption, and this has become a major concern as the demand for computing power continues to increase. Neuromorphic computing systems, on the other hand, are designed to be more energy-efficient, which makes them an attractive alternative to traditional computing systems.

Neuromorphic computing has a wide range of applications and is expected to significantly impact many fields in the coming years. One of the most promising applications of neuromorphic computing is in the field of artificial intelligence [147]. Neuromorphic computing systems can be used to train machine learning algorithms, which can then be used to perform tasks such as image recognition, speech recognition, and natural language processing.

Neuromorphic computing is also being used in the field of robotics [143]. Robots that are designed to mimic the behavior of living organisms can benefit from the principles of neuromorphic computing. By using networks of artificial neurons, robots can learn from their environment and adapt to new situations in real time. Neuromorphic computing has the potential to revolutionize the field of medicine. Researchers are currently exploring the use of neuromorphic computing systems to develop better diagnostic tools and to improve the treatment of neurological disorders such as Alzheimer's disease and Parkinson's disease. Some literature reviews on neuromorphic computing has done in following

Indiveri et al. (2015) [60] provide a comprehensive overview of neuromorphic computing and its applications. The authors discuss the various approaches to neuromorphic computing, including neuromorphic hardware, neuromorphic algorithms, and neuromorphic systems. They also provide examples of neuromorphic computing applications in areas such as image and speech recognition, robotics, and sensor networks. Schemmel et al. [146] present a wafer-scale neuromorphic hardware system for large-scale neural modeling. The authors describe the system's architecture, which includes 384 analog circuits, each containing 256 neurons and 1024 synapses. The authors demonstrate the system's ability to simulate large-scale neural networks and provide examples of its application in areas such as image recognition and robotics. Merolla et al. [100] describe the design and implementation of a neuromorphic computing system that contains one million spiking neurons and is capable of simulating large-scale neural networks. The system is based on an integrated circuit architecture that includes both digital and analog components and uses a communication network to connect multiple chips. Furber [36] provides an overview of large-scale neuromorphic computing systems, including the challenges and opportunities associated with developing such systems. The author discusses various hardware and software platforms used in neuromorphic computing, such as spiking neural networks, memristors, and neuromorphic processors. Zhang et.al. [189] describe the use of memristor crossbars in neuromorphic computing systems. The authors discuss the advantages and challenges associated with using memristors and provide examples of their applications in spiking neural networks. They also discuss the various modeling techniques used to simulate memristor-based systems. Hasler et. al. [51] discusses the challenges and opportunities associated with developing large-scale neuromorphic hardware systems. The authors discuss various approaches to achieving such systems, such as using hierarchical architectures and neuromorphic processors. They also discuss the potential applications of large-scale neuromorphic systems in areas such as robotics and sensor networks. Merolla et. al. [100] developed a million spiking-neuron integrated circuit with a scalable communication network and interface. The authors demonstrated the scalability of the system by simulating a model of 1 billion neurons with 10 trillion synapses. The authors also highlighted the potential of neuromorphic computing for energy-efficient and parallel computing applications.

1.4 Memristors

Memristors are a relatively new addition to the field of electronics, and they have quickly become an exciting area of research. A memristor is a passive two-terminal electrical component that has the ability to remember the amount of charge that has flowed through it in the past. This unique feature makes memristors different from other basic electrical components like resistors, capacitors, and inductors.

Memristors were first theorized by Professor Leon Chua of the University of California, Berkeley, in 1971. However, it wasn't until 2008 that HP Labs announced that they had created a physical memristor. Since then, researchers worldwide have been studying this new component's potential uses and applications.

One of the most promising applications of memristors is in the field of computing. Memristors have the potential to revolutionize computer memory by providing an alternative to traditional DRAM (Dynamic Random Access Memory) and flash memory. Memristors are non-volatile, which means that they don't require power to maintain their state. This makes them ideal for use as long-term storage devices. In addition, memristors have much faster read and write times than traditional memory technologies, which means that they could lead to significant improvements in computer performance.

Another potential use for memristors is in artificial intelligence. Memristors can be used to create neural networks modeled after the human brain's structure. This is because memristors can mimic the behavior of synapses, which are the connections between neurons in the brain. Researchers hope to develop more efficient and powerful AI systems by using memristors to create neural networks.

In addition to their potential applications in computing and AI, memristors also have potential uses in other fields. For example, medical devices could use memristors to create implantable sensors that can monitor and control various physiological processes. They could also be used in energy storage systems to create more efficient and long-lasting batteries.

Despite the potential of memristors, some challenges still need to be addressed. For example, memristors are currently difficult and expensive to produce, which makes them impractical for widespread use. In addition, there are still some questions about the reliability and durability of memristors, which could limit their usefulness in certain applications. The following includes a literature review that has been conducted on Memristors:

Strukov et al. [158] describe the first experimental realization of memristive devices and their unique properties. The researchers demonstrate the use of memristors as electronic synapses in neuromorphic circuits. Yang et al. [181] discuss the potential of memristive devices in computing applications, including logic circuits and memory storage. Jo et al. [64] describe the use of memristors as synapses in neuromorphic circuits and show that they exhibit short-term and long-term plasticity, similar to biological synapses. Pershin et al. [127] propose a memristive learning model inspired by amoeba behavior. The model demonstrates the potential of memristors for implementing learning mechanisms in artificial systems. Wang et al. [172] describe the development of flexible memristive devices based on nanocomposite materials and their potential use in brain-inspired computing. Yang et al. [182] describe the use of nanoscale metallic inclusions in dielectric materials to achieve resistive switching in memristive devices. Kim et al. [70] describe the development of a memristive device that exhibits intrinsic diode characteristics and long endurance, making it a promising candidate for memory storage applications. Chua [20] proposed the existence of memristors as a fourth fundamental circuit element, which can store information and exhibit non-linear behavior. Struck et al. [158] describe the first experimental realization of memristive devices and their unique properties. The researchers demonstrated the use of memristors as electronic synapses in neuromorphic circuits. Alibart et al. [2] demonstrate the use of memristive crossbar circuits for pattern classification tasks, using both ex situ and in situ training. Kuzum et al. [74] describe using phase change materials in memristive devices to implement programmable synapses in brain-inspired computing. Burr et al. [16] discuss the potential of memristive devices in neuromorphic computing applications, including pattern recognition and machine learning, in the review article.

Memristors are a promising new component in the field of electronics. They have the potential to revolutionize computing and AI, as well as a wide range of other areas. Although some challenges still need to be overcome, researchers worldwide are working to improve the performance and reliability of memristors, and we will likely see more widespread use of these components in the coming years.

1.5 Memcapacitors

Memcapacitors, also known as memory capacitors, are passive electronic components that can remember the amount of charge that has passed through them. They are similar to memristors, which can remember the amount of control that has flowed through them, but instead of changing resistance, they change the capacitance.

The concept of a memcapacitor was first theorized in 1971 by Leon Chua [135]. Chua proposed that a memcapacitor could be created by taking two parallel plates and placing a layer of suitable dielectric material between them. When a voltage is applied to the plates, the dielectric material becomes polarized, causing a change in capacitance that depends on the history of the voltage applied.

Like memristors, memcapacitors have the potential to revolutionize a number of different fields. One area where they could be particularly useful is in memory storage. Current memory technologies, such as DRAM and flash memory, have speed, durability, and power consumption limitations. Memcapacitors have the potential to overcome these limitations by providing a more efficient and durable means of storing information.

Another potential application of memcapacitors is in the field of energy storage. Capacitors are currently used in a number of energy storage applications, but they have limitations in terms of the amount of energy they can store and the rate at which they can discharge. On the other hand, Memcapacitors have the potential to store much larger amounts of energy and discharge that point much more quickly.

In addition to their potential applications in memory and energy storage, memcapacitors could be used in other fields. For example, they could be used in sensors to detect and store changes in environmental conditions. They could also be used in medical devices to store and release small amounts of medication over time.

Despite the potential of memcapacitors, some challenges still need to be addressed. One challenge is developing a suitable dielectric material that can be used in memcapacitors. The dielectric material needs to be able to withstand the high voltages that are typically used in electronic circuits, and it needs to be able to maintain its properties over time. Another challenge is developing a reliable and cost-effective method for manufacturing memcapacitors.

In recent years, the memcapacitive device, which operates on the capacitive principle, has garnered attention alongside the memristive device [135]. Although various research groups have proposed theoretical models for memcapacitive devices [28] [97] [101] [126] [68], only a few practical implementations have been demonstrated thus far [174] [75] [185] [192]. While most neuromorphic functions are achieved through emulating electric pulse patterns using solid-state devices, it is still a challenge to implement biological synapse-based devices, which hold superior neuromorphic system functionality [179].

Memcapacitors are a promising new type of electronic component that has the potential to revolutionize several different fields. They can store and remember information more efficiently and durably than current memory technologies, and they have the potential to store much more significant amounts of energy than current capacitors. While some challenges still need to be overcome, researchers are actively working to improve the properties and manufacturing processes of memcapacitors, and we will likely see more widespread use of these components in the coming years.

1.6 Research Objective

The research objective of a biomolecular device-based (memristive/memcapacitive) reservoir computing system is to investigate the potential of using biomolecular devices to build efficient and high-performance reservoir computing systems. The main goal is to develop a biomolecular device-based reservoir computing system to perform complex computations with high accuracy and low power consumption. Specific research objectives may include designing and implementing a biomolecular device-based reservoir computing system, analyzing its precision and power consumption performance for solving temporal and classification problems, optimizing its architecture and parameters, and comparing its performance with other reservoir computing systems. The research also explores potential applications of biomolecular device-based reservoir computing systems in fields such as chaotic signal prediction, image classification, and temporal data recognition. Overall, this research aims to contribute to developing more efficient and effective computing systems that can enable new applications and advance the field of neuromorphic computing. I should emphasize that the majority of my analysis was carried out utilizing RC systems based on biomolecular memristors, while for certain issues, I employed memcapacitor-based RC systems. Consequently, I have used both memristors and memcapacitors to address some problems.

1.7 Organization of the Dissertation

The dissertation work has been organized into five chapters. Chapter One will introduce the research problem, including a brief overview of the background, objectives, research questions, and significance of the study. Chapter two will review the literature on biomolecular devices, focusing on the theoretical framework and previous research studies. Chapter three will describe the reservoir computing system, including the research design, sample selection, data collection, and analysis procedures. Chapters four, five, and six will present

Temporal and classification data analysis using biomolecular Memory based reservoir computing system. Finally, chapter seven will summarize the study's main findings, conclusions, and recommendations for future research.

2 BIOMOLECULAR DEVICES

A biomolecular device is a device that is based on biological molecules, such as DNA or proteins, and uses their properties to perform specific functions. These devices can be designed and engineered to perform various tasks, such as sensing, computation, and actuation, and can be used in a wide range of applications, including biotechnology, medicine, and environmental monitoring. Biomolecular devices can be made using techniques such as genetic engineering, protein engineering, and self-assembly, and they can be integrated with other devices and systems to create complex biomolecular systems. Examples of biomolecular devices include DNA nanodevices, protein sensors, and biosensors. The development of biomolecular devices has the potential to revolutionize many fields, as they offer unique advantages such as high specificity, low cost, and compatibility with biological systems.

2.1 Biomolecular Memristor

The looming conclusion of Moore's Law, the collapse of Dennard Scaling, and the limited bandwidth between the CPU and memory, known as the von Neumann bottleneck, along with the continuous rise in computing requirements, are pushing researchers to investigate alternative computing paradigms [79]. In recent years, neuromorphic computing has surfaced as a supplementary architecture to von Neumann systems. Neuromorphic computers, consisting of artificial neurons and synapses, offer a more effective platform for implementing neural network algorithms compared to conventional architectures.

Despite substantial advancements in very-large-scale-integration (VLSI) circuits [100], neuromorphic networks have yet to achieve the intricacy, neuronal density, and energy efficiency of the human brain. The brain employs complex molecular mechanisms to constantly modify connections between neurons, and the resulting synaptic plasticity [99] allows it to retain patterns, adapt to new information, and execute a vast number of parallel operations with remarkably low power consumption [165]. On the other hand, VLSI networks simulate synaptic activities using transistors, which bear little resemblance to their biological counterparts in terms of mechanisms and necessitate sizable, power-intensive complementary metal-oxide-semiconductor (CMOS) circuitry.

The majority of cutting-edge solid-state devices used in neuromorphic circuits have been designed primarily for high integration density and energy-efficient computation, largely ignoring biological realism in terms of structure and function. A potential alternative is to develop systems that more closely resemble biological systems, which are energy-efficient, flexible, probabilistic, fault-tolerant, and ideally biological in nature. To achieve this, soft-matter biomolecular artificial synapses with memristive and memcapacitive behavior have been explored for emulating the behavior of chemical and electrical synapses for neuromorphic computing and signal processing applications [107, 50, 73, 105, 11, 23, 95, 57, 56, 94]. Najem et al. [107] recently introduced a biomolecular memristor (memory + resistor) with a composition, structure, switching mechanism, and ionic transport that closely resembles bio-synapses. This device can mimic essential synaptic functions, such as paired-pulse facilitation and depression, thanks to a generic memristive property, allowing for learning and computation with significantly reduced power consumption [107, 175]. Additionally, the synapse-like dynamic features of the device enable streamlined learning circuit implementations [175, 50].

In this section, a brief introduction to bio synaptic memristor has been presented.

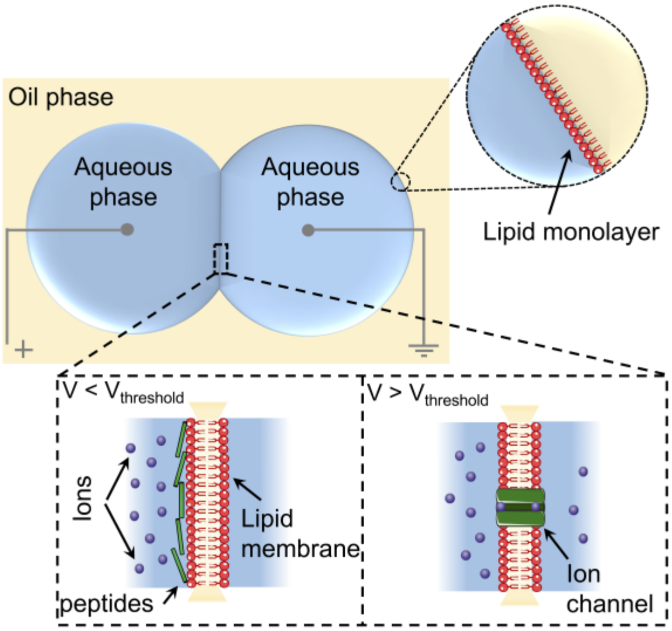


Figure 2.1: Biomolecular memristors featuring alamethicin ion channels for voltage-regulated signal transmission. (Figure from [50])

2.2 Characteristics of Biomimetic Memristive Device

Biological neurons and synapses function by concurrently storing and processing information while maintaining adaptability. These unique features of the brain, such as the co-location of computing and memory along with plasticity, enable it to carry out massive computational tasks while using as little as 20 watts [165]. Traditional silicon-based neuromorphic circuits fall short in replicating the fundamental transport properties of biological synapses and neural networks, consequently necessitating more complex neural networks and power to achieve comparable computational capabilities.

Gaining a profound understanding of how the brain processes and responds to information in intricate environments requires the development of easily configurable devices that imitate nature's biological neural design and the elaborate biomolecular processes responsible for brain memory and computing. To address this need, a soft, two-terminal biomolecular memristor device that emulates the physical structure, switching mechanism, and ion transport of bio-synapses has been recently demonstrated [107, 106]. This device is composed of an alamethicin-doped synthetic biomembrane that is 3-5 nm thick (Fig. 2.1). In brief, the highly insulating ($\sim 10G\Omega$) lipid membrane self-assembles at the interface of two contacting, lipid-encased aqueous droplets placed in hexadecane oil. In the presence of alm peptides and sufficient transmembrane voltage, conductive and memristive ionic pathways are formed through volatile, voltage-driven insertion of alamethicin peptides (alm) into the insulating lipid membrane (Fig. 2.1). At low voltages, where alm peptides are surface-bound, the device is considered to be in the resting state. However, the device suddenly transitions into a voltage-dependent conductive state when voltages exceed a specific potential, $V_{threshold}$. This response closely mimics the voltage-modulated variable conductance observed in biosynapses. The device's current-voltage relationship in response to harmonic voltage input displays pinched hysteresis, which demonstrates the memristive nature of the two-terminal system.

Experiments and simulations have illustrated voltage-dependent threshold switching and volatile memristive behavior governed by two voltage-dependent state variables: the areal density of alamethicin channels and the increase in membrane area due to electrowetting. These variables, in turn, dictate the total number of ion channels and, consequently, the device's net conductance. As a result, the two-terminal device exhibits switching dynamics comparable to depolarizing pulses in actual nerve cells, with short- and long-term plasticity such as paired-pulse facilitation (PPF), paired-pulse depression (PPD), and, when paired with a non-volatile memristor, spike timing-dependent plasticity (STDP) [107, 47]. In comparison to previous memristive devices, this biomolecular memristor consumes significantly less power (0.1-10 nW) and is easier to fabricate . Figure 2.2 shows the hysteresis characteristics of the biomolecular memristor. The response is pinched off at $0mV$.

Figure 2.3 displays the voltage-current response of the biomolecular memristor. Here the voltage pulse train of $170mV$ has been applied to the memristor with on time of 10 ms and off time of 5 ms.

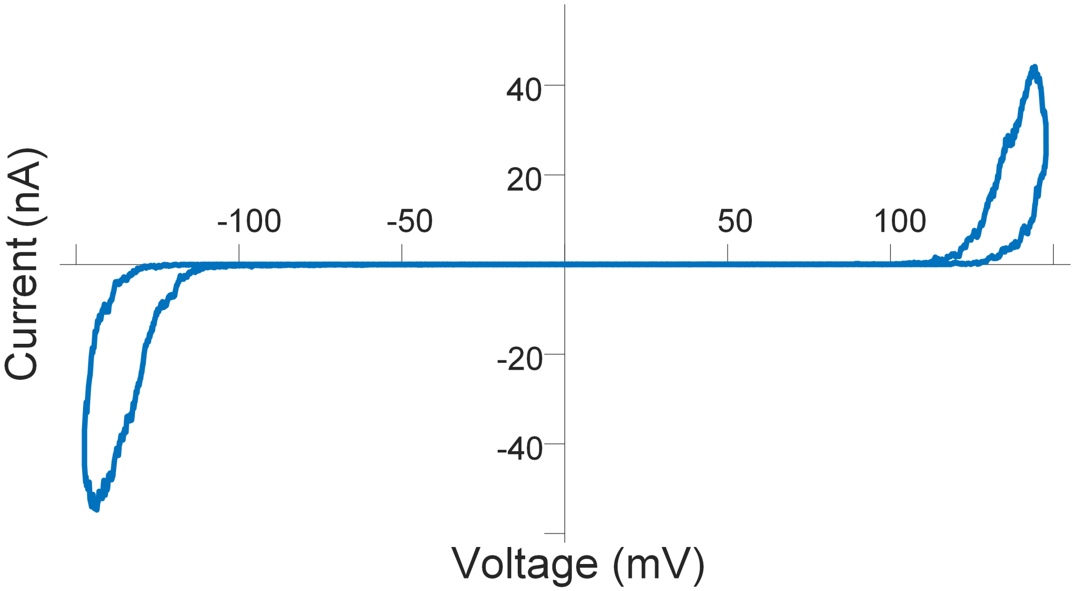


Figure 2.2: Hysteresis characteristics of Biomolecular memristor.

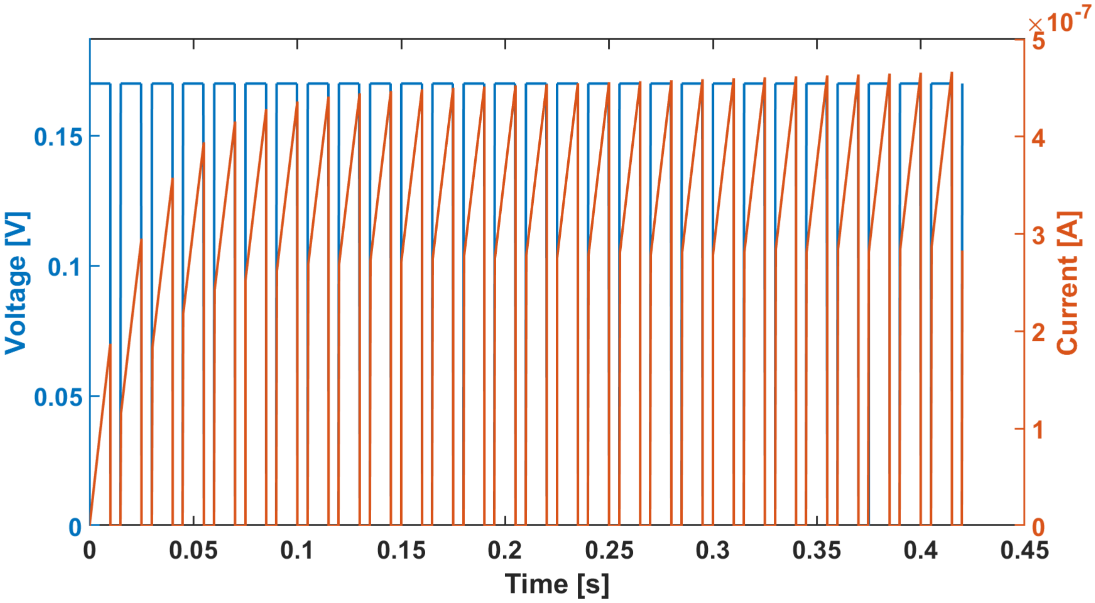


Figure 2.3: Voltage and Current response of Biomolecular memristor.

2.3 Mathematical Expression of Biomimetic Memristive Device

The current-voltage relationship of a standard voltage-controlled memristor can be expressed as:

$$I = G(x)V \tag{2.3.1}$$

$$\frac{dx}{dt} = f(\mathbf{x}; V) \tag{2.3.2}$$

Here, G denotes the nominal memory conductance, and $\mathbf{x}$ represents one or more voltage-controlled state variables that regulate the conductance. The conductance of this biomolecular memristor depends on the total number of alm pores forming ion-channels across the insulating membrane, which in turn is influenced by the areal density of alm pores and the nominal size of the bilayer. Both these factors are voltage-controlled, giving the device its voltage-controlled memristive nature. By selecting the number of open alm pores per unit area, N_a, and the fractional increase in bilayer area, A_m, as two state variables, Eq. 2.3.1 can be reformulated as:

$$I = G(N_a, A_m)V \tag{2.3.3}$$

For an applied voltage V, the state equations for N_a and A_m, as derived in [107], are:

$$\frac{dN_a}{dt} = \frac{1}{\tau_0 exp(|V|/V_t)}(N_0 exp(|V|/V_e) - N_a) \tag{2.3.4}$$

$$\frac{dA_m}{dt} = \frac{1}{\tau_{ew}}(\alpha V^2 - A_m) \tag{2.3.5}$$

Here, V_e, N_0, V_τ, and τ_0 represent the voltage required to cause an e-fold increase in the number of alm pores, a proportionality constant representing the number of alm pores at zero volts, the voltage required to induce an e-fold increase in τ, and the time constant

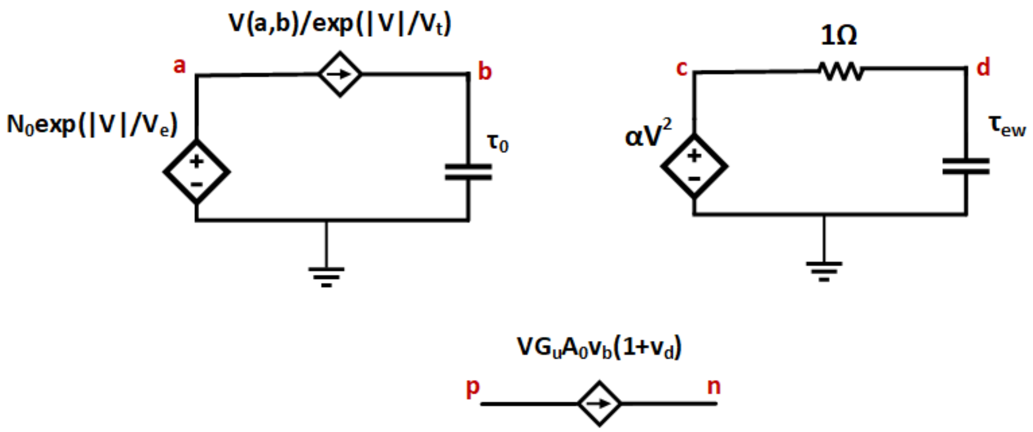


Figure 2.4: Spice Implementation Biomolecular Memristor [50].

for pore closure at zero volts, respectively. α and τ_{ew} are voltage-sensitivity constant and characteristic time constant describing the electrowetting process, respectively. Combining these results, the overall conductance can be expressed as:

$$G(t) = G_u(t)N_a(t)A_0(1 + A_m(t)) \tag{2.3.6}$$

where A_0 and G_u are the bilayer area at zero volts and the average unit conductance of a single alm pore determined by both the structure of the alm pore and the conductivity of the electrolyte solution within the droplets. τ_{ew} and α were determined by fitting numerical solutions of Eq. 2.3.5 to the measured change in membrane area during voltage sweeps, and these, along with measured relaxation time constants (τ_0), were used in another fitting routine to estimate the parameters for alm insertion by fitting numerical solutions of Eq. 2.3.4 to measured I-V responses. The current response over time and I-V hysteresis curve of DPhPC memristor for 0.17 Hz triangular wave of amplitude 160 mV is shown in Fig. 2. As seen in this figure, and the model fits reasonably well with the experimental results. More details on the analytical modeling of the device response for the pulse and sinusoidal inputs can be found in [47].

A subcircuit, depicted in Fig 2.4, was developed to implement the model in SPICE. Here, p and n represent the positive and negative terminals, which are arbitrary since this is a bidirectional symmetric device where the terminals can always be interchanged. The node voltage v_b and v_d are not physical voltages. They represent N_a and A_m, respectively. The primary underlying concept is to reformulate equations 2.3.1 and 2.3.2 in terms of RC circuit. For a series RC circuit driven by a voltage source V_{in}, the equation describing the voltage across the capacitor is:

$$\frac{dv_c}{dt} = \frac{V_{in}}{RC} - \frac{v_c}{RC} \tag{2.3.7}$$

If we replace R, C, and V_{in} with $exp(V/V_\tau)$, (τ_0), and $N_0 exp(V/V_e)$, we get back Eq. 2.3.4, and if we replace those with 1, τ_{ew}, and αV^2, we get back Eq. 2.3.5. We use a voltage-controlled voltage source for V_{in} and a behavioral current source to implement voltage-dependent resistance. Finally, we combine Eq. 2.3.6 and Eq. 2.3.3 using a behavioral current source to obtain the current through the device.

2.4 Biomolecular Memcapacitor

The biomolecular memcapacitor is an innovative and groundbreaking development in the realm of bioelectronic devices, harnessing the unique properties of biological molecules to achieve advanced functionality. By mimicking the natural processes of living organisms, biomolecular memcapacitors are designed to store and process electrical energy in a highly efficient manner, much like how neurons store and transmit information in our brains. This cutting-edge technology combines the advantages of capacitors and memory elements, resulting in a device exhibiting non-linear, hysteretic, and adaptive behavior. Biomolecular memcapacitors have the potential to revolutionize various fields, including medicine, biocomputing, and energy storage, by providing a new platform for designing smart, responsive, and eco-friendly bioelectronic systems.

Najem et al. [105] evidenced that an artificial biomembrane, particularly a lipid bilayer embedded with voltage-activated ion channels, exhibits volatile memory resistance,

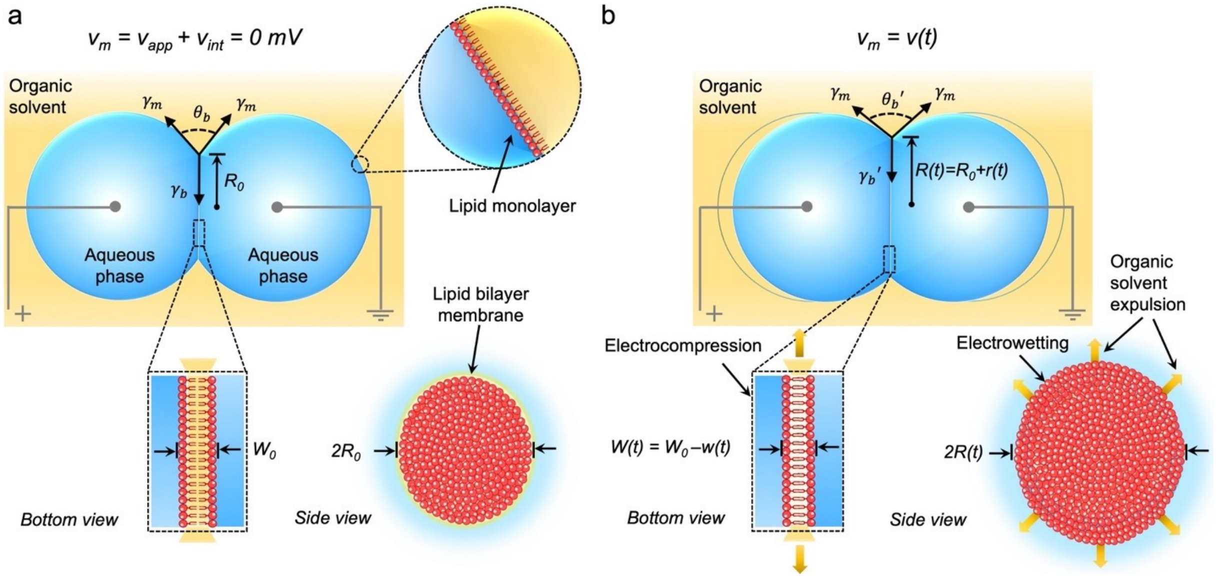


Figure 2.5: Biomolecular memcapacitor assembly and electromechanical behavior. The structural change of the memcapacitor for a) no voltage, $v_m = 0mV$, b) with voltage, $v_m = VmV$. The figure has been taken from [105]

which is governed by two voltage-sensitive parameters: ion channel spatial density and membrane area expansion due to electrowetting [107]. This seminal research prompted us to hypothesize that a non-conductive lipid bilayer, void of ion channels, could display capacitive memory solely influenced by voltage-mediated changes in dielectric dimensions, predominantly steered by the hydrophobic core of the bilayer. Our research revealed that lipid-coated aqueous droplets, upon adhering in oil, form a biomimetic membrane interface (3-5nm in thickness) that showcases volatile, analog memcapacitance via voltage-controlled geometric flexibility. A pinched hysteresis on the charge-voltage plane is attributable to dynamic alterations in the interfacial area and hydrophobic thickness, both of which are nonlinearly reliant on voltage. With the integration of experimental and theoretical approaches, we demonstrate this structure operates as a volatile, second-order, universal memcapacitor [21], capable of executing synapse-like temporal filtering and learning via short-term plasticity. Our research predictions foresee the development of novel, low-power biomimetic memelements, designed from soft, organic materials and biomolecules, that may further our understanding of capacitive memory and susceptibility in neuronal membranes.

Figure 2.5 shows the structural change of the memecapacitor for with/without voltage. Further details has been demonstrated in the following sections.

2.5 Characteristics of Biomimetic Memcapacitor Device

The memcapacitor displays pinched hysteresis loops for its capacitance, bilayer interfacial area (Electrowetting,EW), and hydrophobic thickness (Electrocompression, EC) (figure 2.6. For each loop, the time constant of the memcapacitor along the increasing-voltage trajectory differs from that along the decreasing-voltage trajectory. This indicates that the EW and EC processes of the device are non-reversible as functions of sinusoidal transmembrane potential, which serves as a hallmark of memory. Moreover, pinched $Q - v$ for this particular device can be referenced in prior research [105].

In figure 2.7, the device's fading memory property has been demonstrated. The memcapacitance decays in $\sim 2s$ when a 150mV voltage stimulation is removed.

2.6 Mathematical Expression of Biomimetic Memcapacitor Device

Memory capacitors, or memcapacitors in short, are two terminal nonlinear energy storage elements that exhibit memory properties. The capacitance magnitude nonlinearly depends on one or more internal states and can be regulated based on present and past external stimulation. Like memristors, memcapacitors can be categorized as either non-volatile if the memcapacitors' states are maintained or volatile if their states are unmaintained upon removal of an electrical stimulus [181]. Herein, replicating a device developed by Najem *et al.* [105] formerly, we constructed a lipid bilayer-based parallel-plate memcapacitor $(\sim 0.1 - 1)\mu F \cdot cm^{-2}$ [144, 107] by interfacing two lipid monolayer-encased aqueous droplets ($\sim 200nL$ each) immersed in an oil phase. At the interface of the two droplets, an elliptical, planar lipid bilayer ($\sim 100\mu m$ in radius) spontaneously forms with a highly insulative ($> 100M\Omega \cdot cm^2$) core that is comprised of a mixture of hydrophobic lipid tails and residual entrapped oil. Upon transmembrane voltage application, the ionically charged

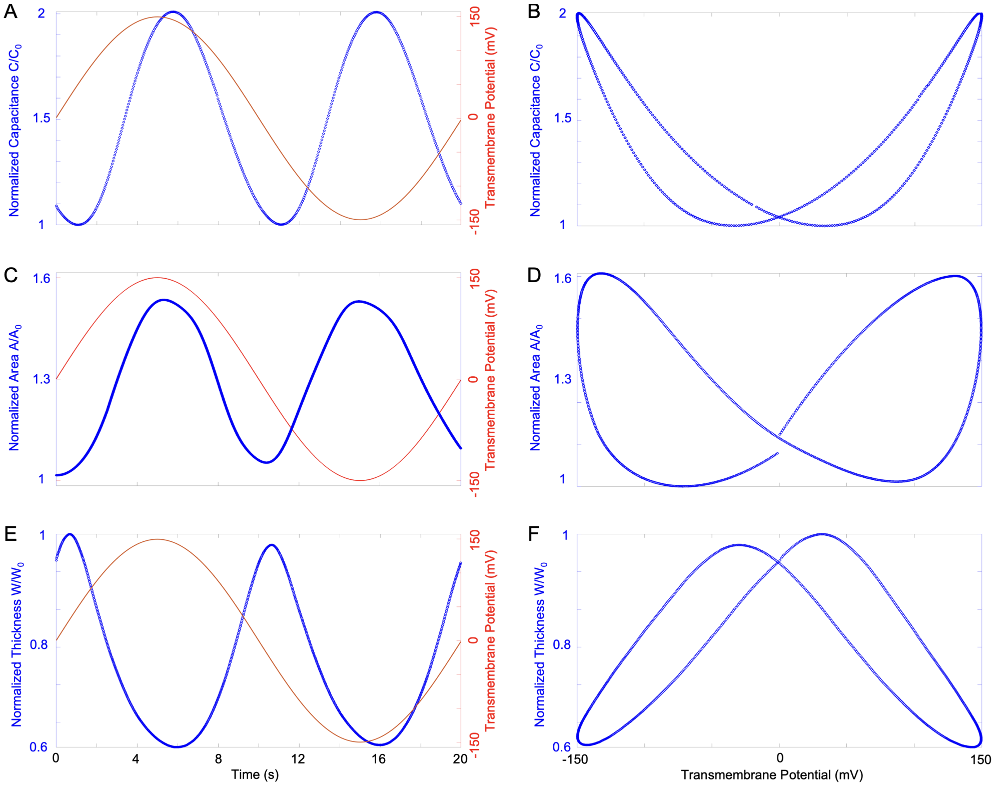


Figure 2.6: (a) Normalized dynamic capacitance as a function of a 150mV, 50mHz, sinusoidal transmembrane potential. (b) A corresponding $C - v$ curve, where a pinched hysteresis was observed near 0mV transmembrane potential. (c) Normalized dynamic bilayer area as a function of a 150mV, 50mHz, sinusoidal transmembrane potential. (d) A corresponding $A - v$ curve, where a pinched hysteresis was observed near 0mV transmembrane potential. (e) Normalized dynamic hydrophobic thickness as a function of a 150mV, 50mHz, sinusoidal transmembrane potential. (f) A corresponding $W - v$ curve, where a pinched hysteresis was observed near 0mV transmembrane potential.

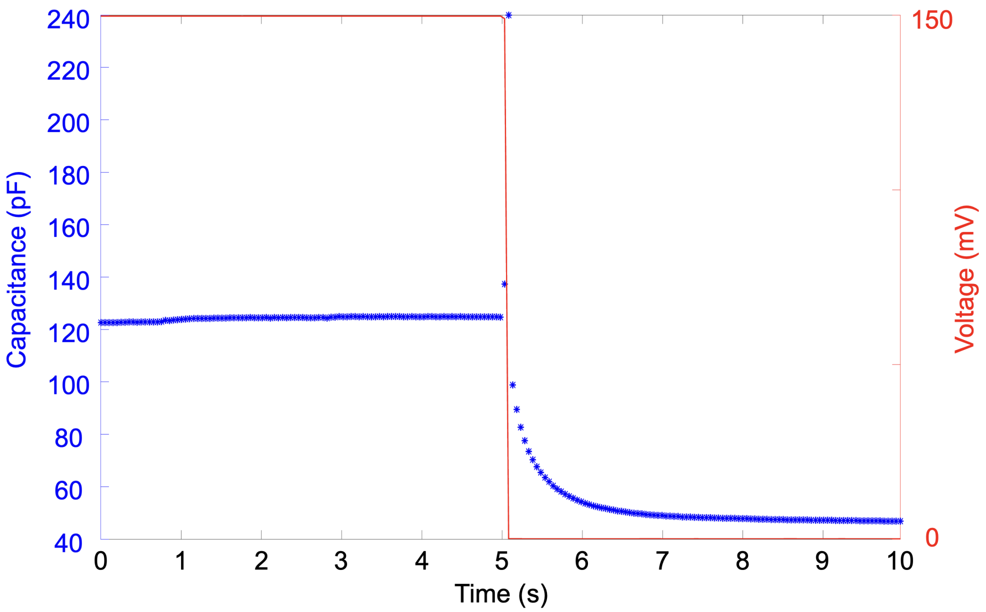


Figure 2.7: Measured restoration of initial capacitance upon voltage stimulus removal

lipid bilayer manifests geometrical changes due to electrowetting (EW) and electrocompression (EC), i.e., bilayer area increase and hydrophobic thickness decrease, respectively. EW is mainly caused by the bilayer tension reduction [131, 161] due to charge-induced electrostatic forces; meanwhile, the electrostatic force-driven entrapped oil expulsion is the main drive for EC [33, 107]. Gradients with respect to time in the minor axis radius, $R(t)$, of the bilayer area $A(t)$ and thickness, $W(t)$, are modeled using the coupled state equations [105] below:

$$\frac{dR(t)}{dt} = \frac{1}{\zeta_{ew}}((\frac{a\epsilon\epsilon_0}{2W(t)})v(t)^2 - k_{ew}(R(t) - R_0)) \tag{2.6.1}$$

$$\frac{dW(t)}{dt} = \frac{1}{\zeta_{ec}}((\frac{-a\epsilon\epsilon_0\pi R(t)^2}{2W(t)^2})v(t)^2 + k_{ec}(W_0 - W(t))) \tag{2.6.2}$$

where a is the eccentricity of an ellipse, ϵ is the equivalent dielectric constant for the hydrophobic tails and residual oil mixture, ϵ_0 is the permittivity of free space, R_0 (m) is the zero-volt, interfacial area minor axis radius, W_0 (m) is the zero-voltage hydrophobic thickness, ζ_{ew} and k_{ew} are the EW effective damping (Nsm^{-2}) and stiffness (Nm^{-2}) coefficients in the tangential direction, respectively, and ζ_{ec} and k_{ec} are the EC effective damping (Nsm^{-1})

and stiffness (Nm^{-1}) coefficients in the normal direction, respectively. Similar to a standard parallel-plate capacitor, the dynamic capacitance, $C(R(t), W(t))$, can be expressed as:

$$C(R(t), W(t)) = \frac{\varepsilon\varepsilon_0 A(t)}{W(t)} = \frac{\varepsilon\varepsilon_0(a\pi R(t)^2)}{W(t)} \tag{2.6.3}$$

The aforementioned voltage-induced area and thickness changes correspond to an analog, nonlinear 2-3 times increase in capacitance. In addition to the nonlinear dependence on absolute voltage, the corresponding capacitance variations exhibit short-term plasticity, particularly paired-pulse facilitation (PPF) [86]. we have found that the memcapacitor's monotonic increase in normalized capacitance to an initial train of four $200mV$ pulses, insinuating short-term plasticity, where the memcapacitance was computed at the end of each pulse. Following the first four pulses, the device was left unstimulated for 6.5 seconds, which restored the initial capacitance state, conveying memory loss in the device. In fact, it takes approximately ($\sim 2s$) for the device's capacitance to fully decay. It is also important to note that the device is slightly stochastic with native cycle-to-cycle variations. The cycle-to-cycle variation can be observed in our device as a subtle difference in magnitude between the first recorded capacitance and restored state capacitance. This relatively short-term memory loss in an unstimulated device implies short-term memory and device volatility [181].

3 RESERVOIR COMPUTING

3.1 Recurrent Neural Networks

Unlike feedforward networks, Recurrent Neural Networks (RNNs) allow data to flow in loops, making it possible to process sequences of data, such as time series or natural language text. The output of a neuron in one-time step becomes the input to the same neuron in the next time step, allowing information to persist and be processed. RNNs are commonly used for tasks such as language translation and speech recognition.

(RNNs) are a type of artificial neural network that are specifically designed to handle sequential data with temporal dependencies. Unlike feedforward neural networks, where the data flows in one direction from input to output, RNNs allow the data to flow in loops, making it possible to process sequences of data such as time series, natural language text, or speech signals.

The key idea behind RNNs is that the output of a neuron at one time step is fed back as input to the same neuron at the next time step, allowing information to persist and be processed over time. This is achieved by introducing a recurrent connection from each neuron back to itself, in addition to the normal feedforward connections from the inputs to the hidden layer. The hidden layer in an RNN is also organized into time steps, so that each time step has its own set of neurons, and the inputs from the previous time step are fed into the neurons of the current time step.

RNNs can be trained using various optimization algorithms, such as gradient descent, to minimize the error between the predicted and actual outputs. During the training process, the weights of the recurrent connections are adjusted to capture the temporal dependencies in the data, while the weights of the feedforward connections are adjusted to extract relevant features from the input data.

RNNs have been used in a wide range of tasks, such as language translation, speech recognition, image captioning, and sequential decision making. For example, in language translation, an RNN can be trained to translate a sentence from one language to another by processing the words one at a time and generating the translated words in sequence. In speech recognition, an RNN can be trained to transcribe speech signals into text by processing the audio signals one time step at a time and generating the transcribed text in sequence.

One of the challenges in training RNNs is the vanishing gradient problem, which occurs when the gradients of the weights become very small during backpropagation, making it difficult to update the weights effectively. This problem is particularly pronounced in deep RNNs, where the gradients have to be backpropagated through many time steps. Several variants of RNNs have been developed to address this issue, such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), which incorporate gating mechanisms to control the flow of information and prevent the vanishing gradient problem.

Types of RNNs: There are several variants of RNNs, each with their own strengths and weaknesses. Some of the most commonly used variants are: Vanilla RNNs are the simplest form of RNNs, where the hidden state is updated at each time step based on the input and the previous hidden state. However, they suffer from the vanishing gradient problem and are unsuitable for processing long sequences.

Long Short-Term Memory (LSTM) Networks: These RNNs incorporate gating mechanisms to control the flow of information and prevent the vanishing gradient problem. LSTMs are particularly well-suited for processing long sequences and have been used for language translation and speech recognition tasks.

Gated Recurrent Units (GRUs): These are another type of RNNs that incorporate gating mechanisms to control the flow of information. GRUs are computationally more efficient than LSTMs and have been used for various tasks such as language modeling and sentiment analysis.

Bidirectional RNNs: In some cases, processing the input sequence in both forward and backward directions may be useful to capture the dependencies in both directions. Bidirectional RNNs are a type of RNN that do exactly this by processing the input sequence in both directions and concatenating the hidden states to form the final output.

Stacking RNNs: Another way to improve the performance of RNNs is to stack multiple RNN layers on top of each other, to form a deep RNN. This allows the network to learn more complex representations of the data and can improve performance on tasks such as language modeling and speech recognition.

RNNs have been used in a wide range of applications, such as:

Natural language processing: RNNs have been used for tasks such as language translation, text classification, sentiment analysis, and language generation [184].

Speech processing: RNNs have been used for speech recognition, speech synthesis, and speaker identification tasks [67].

Computer vision: RNNs have been used for image captioning and video classification tasks [170].

Sequential decision making: RNNs have been used for tasks such as reinforcement learning, where the agent has to make sequential decisions based on the current state of the environment [190].

In conclusion, Recurrent Neural Networks are a powerful type of artificial neural network that are designed to handle sequential data with temporal dependencies. They have been successfully applied to a wide range of tasks and continue to be an active area of research and development. With the increasing availability of large amounts of sequential data, RNNs will likely play an increasingly important role in solving a wide range of problems in various fields.

3.2 Reservoir Computing

Reservoir Computing is a type of recurrent neural network that is designed for online and real-time processing of sequential data. It is based on the idea of using a fixed and untrained hidden layer, called the reservoir, to provide a rich and nonlinear representation of the input data, and then using a trained output layer to make predictions based on this representation.

The hidden layer, or the reservoir, is created using a recurrent network, such as an Echo State Network (ESN) or a Liquid State Machine (LSM), and is designed to capture the short-term dependencies in the data. The reservoir is fixed and not trained, and the only part of the network that is trained is the output layer, which is trained to make predictions based on the representation provided by the reservoir.

One of the key benefits of Reservoir Computing is its computational efficiency, as only the output layer needs to be trained, while the reservoir is fixed. This makes it well-suited for real-time and online processing of sequential data, where the computational resources are limited. Additionally, Reservoir Computing is less prone to overfitting than traditional recurrent neural networks, as the fixed reservoir provides a rich and nonlinear representation of the input data, which can be used by the output layer to make predictions without overfitting to the training data.

The performance of Reservoir Computing depends on several hyperparameters, such as the size of the reservoir, the connectivity of the reservoir, the type of activation function used, and the type of output layer. These hyperparameters need to be tuned to obtain the best performance on the specific problem. The input data is projected into the reservoir, and the projection matrix is typically trained to maximize the performance of the network.

Reservoir Computing has been used in a wide range of applications, such as time series prediction, speech recognition, and chaotic time series analysis. For example, in time series prediction, a Reservoir Computing network can be trained to predict the future values of a time series based on the past values. In speech recognition, a Reservoir Computing network can be trained to transcribe speech signals into text based on the audio signals.

Reservoir Computing is a powerful and efficient approach to processing sequential data, with several advantages over traditional recurrent neural networks. Its simplicity, flexibility, and robustness to overfitting make it well-suited for a wide range of applications, and it continues to be an active area of research and development. With the increasing availability of sequential data, Reservoir Computing will likely play an increasingly important role in solving a wide range of problems in various fields.

3.3 Comparison between RNNs and RC network

RNNs are a type of artificial neural network that are designed to handle sequential data with temporal dependencies. They allow the data to flow in loops, making it possible to process data sequences such as time series, natural language text, or speech signals. The hidden state in an RNN is updated at each time step based on the input and the previous hidden state, and the weights are adjusted during the training process to minimize the error between the predicted and actual outputs.

Reservoir Computing, on the other hand, is a type of recurrent neural network that is designed for online and real-time processing of sequential data. In Reservoir Computing, the hidden layer, called the reservoir, is fixed and not trained, and the only part of the network that is trained is the output layer. The idea behind this approach is to use the reservoir to provide a rich and nonlinear representation of the input data, and then use the output layer to make predictions based on this representation.

One of the key benefits of Reservoir Computing is that it is computationally efficient, as only the output layer needs to be trained. This makes it well-suited for real-time and online processing of sequential data, where computational resources are limited. Additionally, Reservoir Computing has been shown to be effective in various tasks, such as time series prediction, speech recognition, and chaotic time series analysis.

While Recurrent Neural Networks (RNNs) and Reservoir Computing are both approaches to processing sequential data, Reservoir Computing has several advantages over RNNs in certain scenarios. Here are some of the reasons why Reservoir Computing is considered better than RNNs in some cases:

Computational Efficiency: One of the key benefits of Reservoir Computing is that it is computationally efficient, as only the output layer needs to be trained. This makes it well-suited for real-time and online processing of sequential data, where the computational resources are limited. In contrast, RNNs require training of all the weights in the network, which can be computationally expensive, especially for deep RNNs with many layers.

Robustness to Overfitting: Another advantage of Reservoir Computing is that it is less prone to overfitting than RNNs. This is because the hidden layer, or the reservoir, is fixed and not trained, and the only part of the network that is trained is the output layer. This means that the reservoir provides a rich and nonlinear representation of the input data, which can be used by the output layer to make predictions without overfitting to the training data.

Simplicity: Reservoir Computing is also considered to be simpler than RNNs, as it only requires training of the output layer, while the reservoir is fixed. This makes it easier to implement and faster to train, especially for real-time and online processing of sequential data.

Flexibility: The fixed and untrained reservoir in Reservoir Computing provides a rich and nonlinear representation of the input data, which can be used by the output layer to make predictions. This allows for more flexibility in choosing the reservoir structure, and makes it possible to use Reservoir Computing for a wide range of tasks, such as time series prediction, speech recognition, and chaotic time series analysis.

It's important to note that while Reservoir Computing has several advantages over RNNs in certain scenarios, RNNs are a more general and flexible approach to processing sequential data, and are well-suited for a wide range of tasks. The choice between RNNs and Reservoir Computing depends on the specific problem and the computational resources available.

3.4 Basic Reservoir Computing Network

The basic three layers of a Reservoir Computing network are the input layer, the reservoir layer, and the output layer.

Input Layer: The input layer feeds the input data into the network. The input data is typically a sequence of vectors, such as time series data or speech signals. The input layer projects the input data into the reservoir, and the projection matrix is typically trained to maximize the performance of the network.

Reservoir Layer: The reservoir layer is the fixed and untrained hidden layer in the Reservoir Computing network. It is created using a recurrent network, such as an Echo State Network (ESN) or a Liquid State Machine (LSM), and is designed to capture the short-term dependencies in the data. The reservoir provides a rich and nonlinear representation of the input data, which can be used by the output layer to make predictions.

Output Layer: The output layer is the trained part of the Reservoir Computing network and is responsible for making predictions based on the representation provided by the reservoir. The output layer can be a simple linear layer, or a more complex nonlinear layer, such as a multi-layer perceptron. The output layer weights are adjusted during the training process to minimize the error between the predicted and actual outputs.

Each of these three layers plays an important role in the overall performance of the Reservoir Computing network. The input layer projects the input data into the reservoir, the reservoir layer provides a rich and nonlinear representation of the input data, and the output layer makes predictions based on this representation. By understanding the basic three layers of a Reservoir Computing network, one can design and train more effective Reservoir Computing networks for various problems.

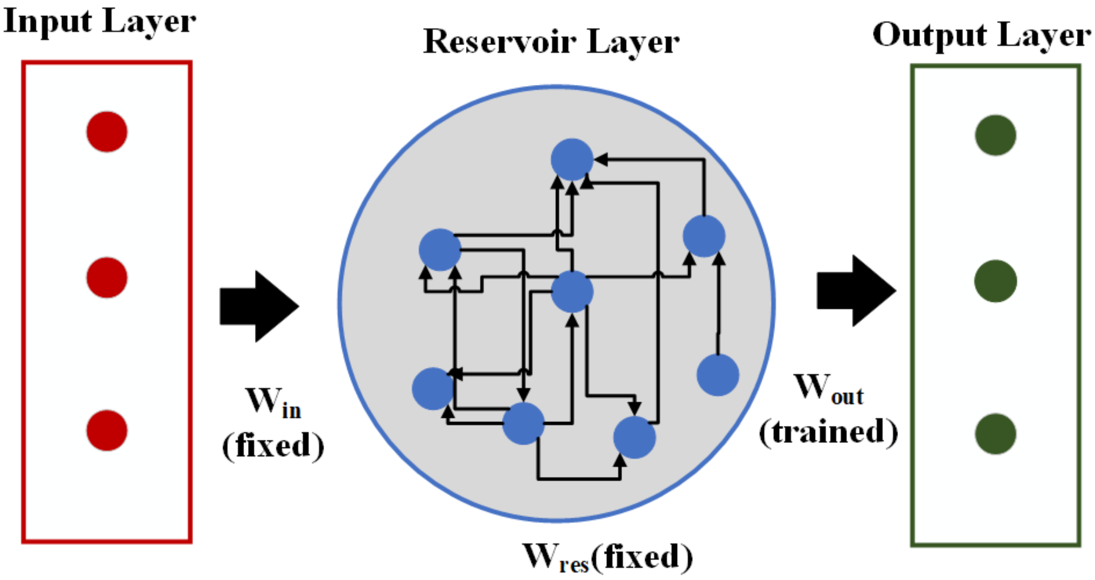


Figure 3.1: Conventional Reservoir Computing architecture.

Figure 3.1 shows a conventional reservoir computing system [160, 62, 166]. The input weights and reservoir weights remain fixed (W_{in}, W_{res}), and the output weights (W_{out}) are trained using the regression model.

The general input and output mathematical model for Reservoir Computing can be represented as:

$$x(t+1) = f(W_{in}u(t) + W_{res}x(t)) \tag{3.4.1}$$

$$y(t) = W_{out}x(t) \tag{3.4.2}$$

Here x(t) is the state of the reservoir at time step t, u(t) is the input signal at time step t, y(t) is the output signal at time step t, W_{in} is the input weight matrix, W_{res} is the recurrent weight matrix of the reservoir,W_{out} is the output weight matrix, and f is a non-linear activation function. The state of the reservoir, x(t), is a high-dimensional representation of the input signal that can be used for various tasks such as time series prediction, classification, and pattern recognition. The output signal, y(t), is obtained by training a linear readout layer on the state of the reservoir.

3.5 Masked Based RC network

Masked-based Reservoir Computing is a relatively new computational paradigm that combines the strengths of both traditional recurrent neural networks (RNNs) and Echo State Networks (ESNs). It is a machine learning model designed to process sequential data and has been used for various tasks such as time series prediction, natural language processing, and speech recognition.

In traditional RNNs, the model learns to process sequential data by updating its hidden state with each time step. The hidden state is updated based on both the previous hidden state and the current input. However, the training process in traditional RNNs can be challenging due to the vanishing or exploding gradient problem, preventing the model from learning long-term dependencies in the data.

On the other hand, ESNs have been designed to overcome the training difficulties of traditional RNNs by using a fixed, randomly generated recurrent weight matrix. The fixed weights in ESNs allow the model to store information about the long-term dependencies in the data while the readout layer is trained to perform the desired task. However, ESNs can struggle with the curse of dimensionality, as the number of neurons in the reservoir needs to be large enough to capture the necessary information from the input data.

Masked-based Reservoir Computing aims to overcome the limitations of both traditional RNNs and ESNs by using a fixed, randomly generated recurrent weight matrix that is masked in a specific way. The masking process in Masked-based Reservoir Computing controls the number of non-zero entries in the weight matrix, which helps reduce the number of neurons required in the reservoir. This, in turn, helps to reduce the curse of dimensionality and make the model more computationally efficient.

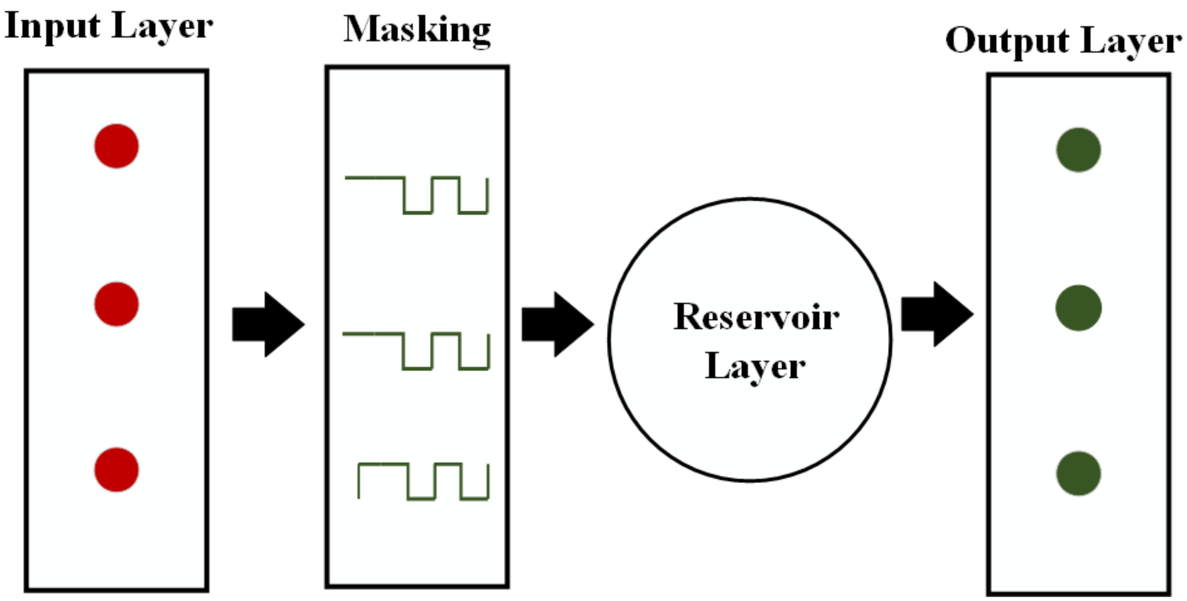


Figure 3.2: Masked-based Reservoir Computing architecture.

In Masked-based Reservoir Computing, the recurrent weight matrix is generated by randomly selecting a small number of neurons to be connected to each other. This creates a sparse matrix with a low density of connections, which helps reduce the model's computational cost. Additionally, the masked recurrent weight matrix allows the model to capture the necessary information from the input data without requiring many neurons in the reservoir.

Masked-based Reservoir Computing is a promising computational paradigm combining the strengths of traditional RNNs and ESNs to provide a more efficient and effective way to process sequential data. Its use of masked recurrent weight matrices helps to overcome the limitations of both traditional RNNs and ESNs, making it a valuable tool for various tasks such as time series prediction, natural language processing, and speech recognition. With its potential for improved performance and reduced computational cost, MRC is an exciting research area worth exploring further.

Figure 3.2 shows masked based reservoir computing system.

The masking technique (also known as time division multiplexing) serves a dual purpose: it sequences the input and maximizes the utilization of the system's dimensionality. In traditional network approaches, nodes in the reservoir can be directly accessed through direct

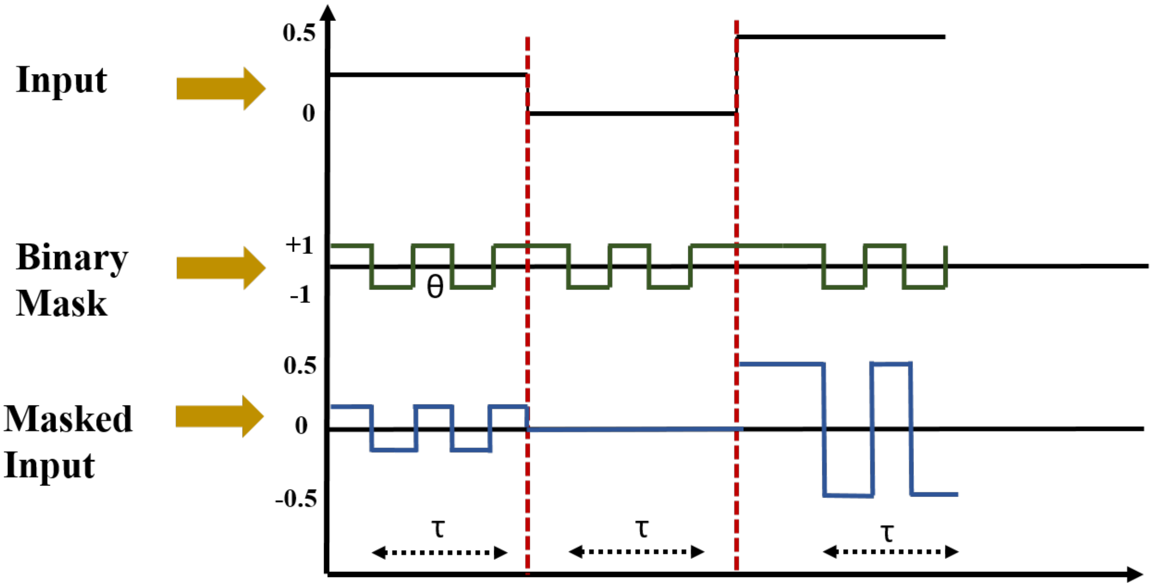


Figure 3.3: Input data process using the binary mask.

connections from the input layer to the reservoir layer. However, in the delayed feedback approach, the input signal is first subjected to nonlinear transformation at a nonlinear node before propagating through the delay line to the virtual nodes. The proper input scaling is imprinted on the input signal before its injection, and time multiplexing is used to provide the input signal to the corresponding virtual node. This results in a piecewise constant input series, with constant intervals corresponding to the separation between virtual nodes in the delay line [8].

The delayed coupling in the system provides an infinite-dimensional state space. However, the available dimensions are not utilized optimally without applying different scaling factors to each node. Similar inputs would result in similar outputs, limiting the span of the high-dimensional state space. The mask containing the scaling factors can be chosen randomly for many nodes, but this approach may not yield desirable results for a smaller set of nodes. Therefore, a reliable method for assigning mask values that maximize diversity in the reservoir states is highly desirable [8].

Figure 3.3 shows masked based reservoir computing system. The input signal (0 to 0.5) is sent to the masking layer. The binary mask (-1,1) processes the input with many nonlinear nodes, eventually incorporating the reservoir to process the data.

3.6 Memristive RC network

Memristive Reservoir Computing (MRC) is a type of Reservoir Computing network that uses memristive devices, such as memristors, as the building blocks of the reservoir layer. Memristors are nanoscale devices that can change their resistance based on the amount of current that has flowed through them, making them well-suited for use in Reservoir Computing networks.

The main advantage of using memristors in Reservoir Computing is that they provide a rich and nonlinear representation of the input data, which the output layer can use to make predictions. Memristors are also highly scalable and can be integrated into extensive networks, making it possible to create large and complex Reservoir Computing networks. Additionally, memristors are low-power and can be integrated into portable and wearable devices. MRC is well-suited for real-time and online sequential data processing in resource-constrained environments.

Another advantage of MRC is that the memristive devices can be used to store the state of the reservoir, making it possible to create stateful Reservoir Computing networks. This allows MRC networks to have a memory of past inputs, which can be used to make predictions based on the history of the input data.

Memristive Reservoir Computing (MRC) is a type of Reservoir Computing network that uses memristive devices as the building blocks of the reservoir layer. MRC has several advantages, such as scalability, low power, and statefulness, making it well-suited for real-time and online processing of sequential data in resource-constrained environments. Despite its limitations, MRC is an active area of research and development and has the potential to play an essential role in solving a wide range of problems in various fields.

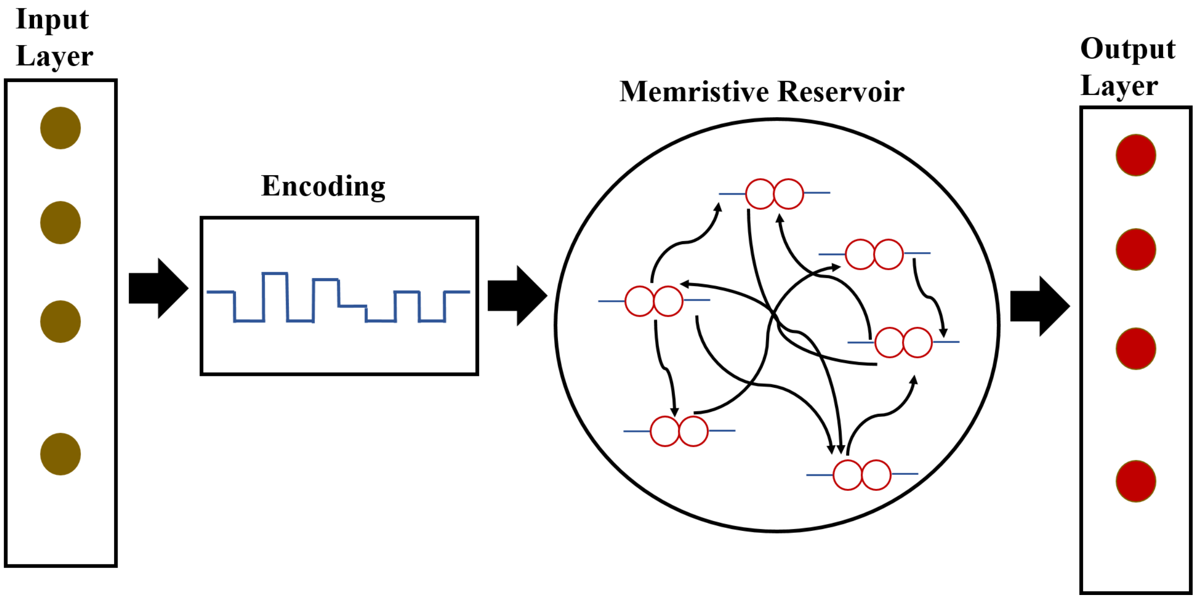


Figure 3.4: Memristive Reservoir Computing architecture.

Figure 3.4 shows the Memristive based reservoir computing system. The input signals are fed into the encoding layer. The input signals can be in voltage, current, or any other suitable form that can modulate the memristor's state. Since the memristor needs voltage for operation, the input signals are encoded within a specific voltage range and made ready for the reservoir layer. The system's core component is the memristive reservoir, a network of interconnected memristors. The memristors are connected in a random, sparse topology to create a complex dynamical system. This reservoir exhibits rich and diverse transient responses to input signals. It has a unique ability to adapt to input signals due to its inherent memory, enabling the system to learn and process information over time. The output layer collects the responses from the memristive reservoir and generates a desired output signal. This layer consists of output nodes connected to the reservoir nodes through readout weights. The output layer can be trained using various learning algorithms to optimize the readout weights, thereby improving the performance of the system. Since the memcapacitor and memristor network architecture is identical, I have explained the principle of only using biomolecular memristor in the reservoir [57, 58].

4 CHAOTIC TIME SERIES PREDICTION

4.1 Chaos

In a world that is inherently unpredictable and complex, the concept of chaos has long captivated the minds of scientists, mathematicians, and philosophers alike. Chaos, a seemingly disordered and random phenomenon, emerges in various disciplines, from the natural sciences to social systems. The enigmatic nature of chaos offers a unique opportunity to delve into the intricate patterns that underlie disorder and unveil the hidden harmony within. This thesis aims to comprehensively understand chaos theory, its applications, and its implications for future research and technology.

Chaos theory, a dynamical systems theory subfield, explores systems' behavior highly sensitive to initial conditions, leading to unpredictable and disorderly outcomes [111]. The interdisciplinary nature of chaos has contributed to its wide-ranging impact, from meteorology and ecology to finance and social sciences. Understanding the complexities of chaos has enhanced our comprehension of natural systems. Still, it has also enabled us to develop advanced technologies and models that can harness the power of chaos for the benefit of society.

The concept of a chaotic system holds a significant position in the realm of complex systems, as it challenges the deterministic notion of predictability and order. Chaotic systems [130, 128] are characterized by their inherent sensitivity to initial conditions, which leads to seemingly unpredictable and erratic behavior over time. Despite the apparent randomness, these systems exhibit an underlying deterministic nature governed by nonlinear equations that dictate their evolution. This paradoxical blend of order and chaos has captivated researchers across various disciplines, including physics, mathematics, and biology, driving them to explore its implications and potential applications. Unraveling the intricate dynamics of chaotic systems furthers our understanding of the mechanisms that govern natural phenomena and paves the way for harnessing chaos to solve complex problems and develop innovative technologies.

Chaotic systems exhibit defining characteristics that differentiate them from other dynamic systems. Understanding these characteristics [52, 162] is essential for studying chaos theory and analyzing complex, nonlinear phenomena. Some of the key characteristics of chaotic systems include:

1. Nonlinearity: Chaotic systems are inherently nonlinear, meaning that their behavior cannot be described by linear equations or models. Nonlinearity often leads to complex and unpredictable interactions between system components, which is a primary reason for the chaotic behavior observed in these systems.

2. Sensitivity to Initial Conditions: A hallmark of chaotic systems is their extreme sensitivity to initial conditions, often called the "butterfly effect"[152, 115]. Even the smallest variation in the initial state of a chaotic system can lead to drastically different outcomes, making it difficult to predict the system's behavior over time.

3. Deterministic yet Unpredictable: Despite being deterministic in nature, chaotic systems exhibit highly unpredictable behavior. Although the equations governing their dynamics are deterministic, the extreme sensitivity to initial conditions makes it nearly impossible to accurately predict the system's long-term behavior.

4. Strange Attractors: Many chaotic systems exhibit a phenomenon known as strange attractors. These are points in the system's state space toward which the system's trajectory converges over time. Strange attractors have a fractal structure, and their intricate patterns provide insights into the complex dynamics of chaotic systems.

5. Fractal Geometry: Chaotic systems often exhibit self-similar patterns and structures that can be described using fractal geometry. Fractals are mathematical constructs characterized by their non-integer dimensions and scale-invariant properties, which help quantify the complexity of chaotic systems and their strange attractors.

6. Mixing and Ergodicity: Chaotic systems display mixing behavior, meaning that trajectories in the system's state space become increasingly intertwined. This leads to ergodicity, which implies that the system's long-term behavior is determined solely by the underlying dynamics and not by the initial conditions.

7. Boundedness: Although chaotic systems exhibit unpredictable behavior, their dynamics are typically confined within a bounded region of the state space. This means that chaotic systems do not diverge to infinity but instead remain within a certain range of values, often characterized by the strange attractor.

8. Lyapunov Exponents: These are numerical measures used to quantify the sensitivity of a chaotic system to initial conditions. Positive Lyapunov exponents indicate the presence of chaos, as they signify that nearby trajectories in the system's state space diverge exponentially over time.

Understanding these characteristics is crucial for studying chaotic systems and their applications across various fields, including physics, engineering, biology, and economics.

Chaotic systems have numerous applications [173, 171, 164] across various fields, offering valuable insights into complex, nonlinear phenomena. Some of the key applications of chaotic systems include:

1. Weather and Climate Modeling: The earth's atmosphere is highly complex and nonlinear, making it difficult to accurately predict weather patterns and climate changes. Chaos theory has been instrumental in understanding the inherent unpredictability of atmospheric dynamics, leading to improvements in weather forecasting and climate models.

2. Ecology and Population Dynamics: Chaotic systems can be used to model the dynamics of interacting populations in ecosystems. These models can help predict fluctuations in population sizes and the potential for sudden shifts in population dynamics due to changes in environmental factors.

3. Neuroscience and Brain Dynamics: The human brain is a complex network of interconnected neurons that exhibit nonlinear behavior. Chaotic systems have been used to model brain dynamics, providing insights into the mechanisms underlying various cognitive functions and brain disorders.

4. Engineering and Control Systems: Chaotic behavior can be found in numerous engineering systems, such as electronic circuits, fluid dynamics, and mechanical systems. Understanding chaotic dynamics helps engineers design more robust and efficient control systems and enhances the performance and stability of these systems.

5. Finance and Economics: The financial markets and economic systems often exhibit chaotic behavior, characterized by unpredictable fluctuations and sudden changes. Chaos theory has been applied to model the dynamics of financial markets, leading to better risk management strategies and improved decision-making processes.

6. Secure Communications: Chaotic systems can be employed to design secure communication systems, as they can generate complex, random-like signals that are difficult to intercept or decode. The unpredictability of chaotic systems makes them suitable for creating encryption schemes and secure communication protocols.

7. Biological Systems: Various biological systems, such as gene regulatory networks, cellular signaling pathways, and cardiac dynamics, exhibit chaotic behavior. Analyzing these systems using chaos theory can provide insights into their underlying mechanisms and help develop novel therapeutic strategies for diseases.

These applications demonstrate the wide-ranging impact of chaotic systems. They provide valuable insights into the dynamics of complex systems and help develop novel solutions to various real-world problems.

4.2 Chaotic Maps

Chaotic maps are a class of discrete-time dynamical systems that exhibit chaotic behavior. These mathematical models are characterized by a simple transformation rule or mapping function that governs the system's evolution from one state to the next. Chaotic maps often possess key characteristics of chaotic systems, such as nonlinearity, sensitivity to initial conditions, and unpredictability. Examples of well-known chaotic maps include the logistic, Hénon, and Arnold's cat maps. These maps have been extensively studied in the context of chaos theory and have provided valuable insights into the fundamental principles of chaotic dynamics. Moreover, chaotic maps have applications in various fields, such as secure communications, image encryption, and pseudorandom number generation. By analyzing the behavior of chaotic maps, researchers can gain a deeper understanding of the mechanisms that drive chaos and complexity in discrete and continuous-time dynamical systems.

Chaotic maps can be categorized into different types based on various criteria, such as the number of variables involved, the mathematical structure of the mapping function, or the specific properties they exhibit. Here are some common types of chaotic maps:

One-dimensional Chaotic Maps: These maps involve a single variable and are often used as introductory models for understanding chaotic behavior. Examples include: a. Logistic Map: A simple one-dimensional map defined by the equation,

$$x_{n+1} = r * x_n * (1 - x_n) \tag{4.2.1}$$

where x_n represents the current state, x_{n+1} the next state, and r is a parameter. Despite its simplicity, the logistic map can exhibit complex behavior for specific values of r.

b. Tent Map: Another one-dimensional map characterized by the equation,

$$x_{n+1} = r * min(x_n, 1 - x_n) \tag{4.2.2}$$

where x_n is the current state, x_{n+1} the next state, and r is a parameter. The tent map derives its name from the tent-like shape of its graph.

Multidimensional Chaotic Maps: These maps involve multiple variables, often modeled as vectors in higher-dimensional state spaces. Examples include: a. Hénon Map: A two-dimensional chaotic map defined by the equations

$$x_{n+1} = 1 - a * x_n^2 + y_n \tag{4.2.3}$$

$$y_{n+1} = b * x_n \tag{4.2.4}$$

where a and b are parameters. The Hénon map is known for its strange attractor, which has a fractal structure.

b. Arnold's Cat Map: A two-dimensional, area-preserving chaotic map representing a toral automorphism. It is defined by the matrix transformation

$$(x_{n+1}, y_{n+1}) = (x_n + y_n, x_n + 2 * y_n) mod 1 \tag{4.2.5}$$

where (x_n, y_n) is the current state and (x_{n+1}, y_{n+1}) is the next state. The map is named after Vladimir Arnold, who used it to illustrate the properties of dynamical systems.

Continuous Maps: Although chaotic maps are typically discrete-time systems, there are continuous counterparts that can exhibit chaotic behavior, such as: a. Circle Map: A continuous map defined on the unit circle, given by the equation

$$\theta_{n+1} = \theta_n + \Omega - (K/2\pi) * sin(2\pi * \theta_n)(mod 1) \tag{4.2.6}$$

Here, θ_n is the current angle, θ_{n+1} is the next angle, and Ω and K are parameters. The circle map is used to study phase locking and mode-locking in coupled oscillators.

Chaotic Maps with Special Properties: Some chaotic maps exhibit specific properties that make them particularly interesting for certain applications:

a. Baker's Map: A piecewise-linear chaotic map that preserves the area and volume of the state space. It is defined on the unit square and involves stretching, folding, and reassembling the square in each iteration. Baker's map is often used to demonstrate ergodicity and mixing in chaotic systems.

These examples illustrate the diversity of chaotic maps and their potential applications in various fields, including secure communications, image encryption, and complex system modeling. A list of literature has been reviewed in the following paragraph:

In this article [87], Lorenz, E. N. introduced the Lorenz system, a system of ordinary differential equations that exhibit chaotic behavior. The Lorenz system is a significant chaotic map that has been extensively studied and applied in various fields. Li et al. [83] have presented a theorem stating that if a continuous one-dimensional map has a period-three orbit, it must also have orbits of every other period and exhibit chaotic behavior. The book [27] provides an accessible introduction to chaotic maps and dynamical systems. It covers the fundamentals of one-dimensional dynamics and the concept of chaos in mathematical systems. May's [98] paper discusses how simple mathematical models, including chaotic maps, can exhibit complex dynamics. It highlights the importance of studying chaotic behavior in diverse fields, including ecology, economics, and biology. This comprehensive textbook [113] covers the theory and applications of chaotic maps and dynamical systems. It discusses different types of chaotic maps, their properties, and their relevance to various scientific disciplines. Pecora et al. [124] investigate the synchronization of chaotic systems, including chaotic maps, and demonstrate how they can be used for secure communication and other applications. Brown et al. use a compact analog discrete-time chaotic circuit [150] for exploring stochasticity in neuromorphic computing [15]. This book [136] explores the concept of synchronization in chaotic maps and other nonlinear systems, presenting various methods and applications in diverse fields. This book [164] offers an overview of recent advances in chaotic systems and their applications, focusing on chaotic maps, synchronization, control, and secure communications. Energy and area efficient analog map, leveraging the intrinsic nonlinearity of transistors, has attracted a lot of attention [65, 120, 139, 69, 121, 122] since it offers an alternate paradigm for chaotic system implementation in stark contrast to conventional digital replication of mathematical maps. Gong et al. [42] introduces a new four-dimensional chaotic map and demonstrates its application in image encryption, highlighting the practical uses of chaotic maps in secure communication. This science book [41] offers a comprehensive and accessible introduction to chaos theory, discussing chaotic maps and other examples of chaotic systems in various scientific fields. This book [157] provides a detailed introduction to nonlinear dynamics and chaos, including the analysis of chaotic maps and other chaotic systems, with applications in diverse disciplines. Baptista [10] explores the use of chaotic maps in cryptography, demonstrating the potential for secure communication using chaotic systems. Compact analog maps have been used for mitigation of side-channel attack [92, 91, 46]. Chaos-based reconfigurable logic [150, 138, 141] has been explored in numerous implementations and has potential logic obfuscation and PUF (Physical Unclonable Function) [149]. Wolf et al. [177] introduce a method for estimating the Lyapunov exponents of chaotic maps and other dynamical systems, a key indicator of chaos, using time-series data. Casdagli et al. [17] discuss techniques for reconstructing the state space of chaotic maps and other dynamical systems from noisy data, a critical step in analyzing real-world chaotic phenomena. Compact analog 1D and 2D maps have been explored as promising candidates for low-cost RNG (Random Number Generator) design [123, 119, 148, 118] specially geared towards resource-constrained IoT (Internet of Things) applications. Robert L Devaney [66] provides an in-depth introduction to nonlinear time series analysis, focusing on chaotic maps and other chaotic systems, including techniques for detecting chaos and estimating system parameters. This book presents [167] a thorough treatment of Lyapunov exponents, a key concept in studying chaotic maps and other dynamical systems, including their computation, properties, and applications. Rulkov et al.

[137] investigate the concept of generalized synchronization in chaotic maps and other nonlinear systems, exploring the relationship between synchronization and directed dynamical influence.

4.3 Lyapunov exponent

The Lyapunov exponent (LE) [49] is a key concept in studying dynamic systems and chaos theory. It quantitatively measures the rate at which nearby trajectories in phase space diverge or converge. Hence, it can be used to assess the predictability and sensitivity to the initial conditions of a given system. In the context of chaotic systems, the Lyapunov exponent helps us understand the nature and degree of instability within the system.

The Lyapunov exponent (denoted by λ) quantifies the rate of separation or convergence of two infinitesimally close trajectories in the phase space of a dynamical system. A positive Lyapunov exponent indicates that the trajectories diverge exponentially with time, suggesting chaos and sensitivity to initial conditions. Conversely, a negative Lyapunov exponent indicates that the trajectories converge, suggesting stability and predictability in the system's behavior.

Multiple Lyapunov exponents can be associated with different directions in the phase space for a given system. The largest Lyapunov exponent (LLE) is of particular interest, as it determines the overall predictability of the system. If the LLE is positive, the system is generally considered chaotic. Calculating the Lyapunov exponent involves linearizing the system's equations of motion and analyzing the perturbations' behavior. The exponent is obtained by taking the average logarithm of the divergence rate of nearby trajectories over time.

Let us consider a simple example of a one-dimensional discrete-time dynamical system, the logistic map, to illustrate the concept of the Lyapunov exponent. The logistic map is reformatted and expressed in following

$$x_{n+1} = 4ax_n(1 - x_n) \tag{4.3.1}$$

where a is a bifurcation parameter ranging from 0 to 1. x_n is the system's state at time step n, $x_n + 1$ is the state at the next time step n+1, and r is a parameter controlling the system's behavior. The value of x_n lies between 0 and 1, representing the population of a species as a fraction of the maximum possible population.

The logistic map exhibits various behaviors, including periodicity, quasi-periodicity, and chaos, depending on the value of r. The system becomes chaotic for certain values of r, meaning that it displays sensitivity to initial conditions and seemingly random behavior.

To compute the Lyapunov exponent for the logistic map, we first linearize the equation by considering a small perturbation $\delta(x_n)$ in the state x_n:

$$\delta(x_{n+1}) = (r - 2 * r * x_n) * \delta(x_n) = f(x)\delta(x_n) \tag{4.3.2}$$

Now, we track the evolution of the perturbation $\delta(x_n)$ over time. We initialize two nearby points in the phase space, x_n and $x_n + \delta(x_n)$, and observe how their difference (the perturbation) evolves as the system iterates. We then calculate the Lyapunov exponent as follows [139]:

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} ln|f(x_i)| \tag{4.3.3}$$

where the sum runs from i=0 to n-1, and x_i represents the system's state at each iteration.

A positive Lyapunov exponent indicates that the perturbation $\delta(x_n)$ grows exponentially over time, meaning that nearby points in the phase space diverge and the system is chaotic. Conversely, a negative exponent implies that the perturbation decreases and the system is stable.

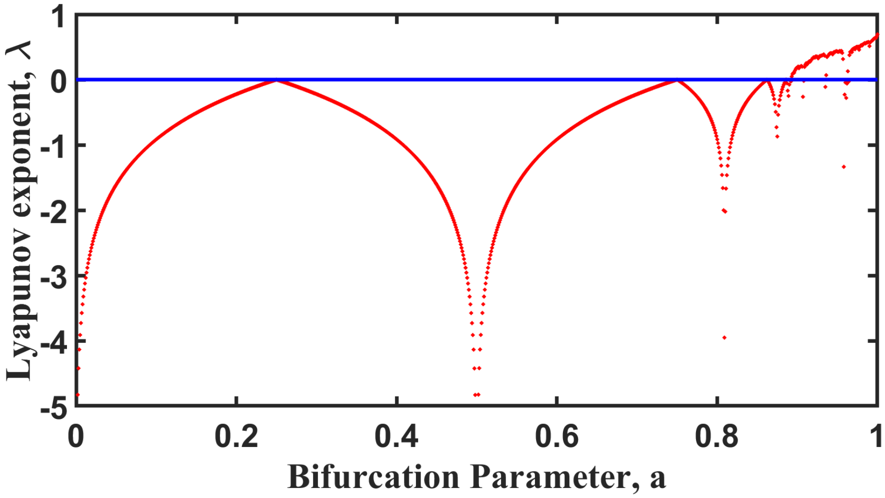


Figure 4.1: Lyapunov exponent of Logistic map.

For the logistic map, the Lyapunov exponent depends on a value. When a is between 0 and 1, the Lyapunov exponent is negative, and the system converges to a fixed point. The Lyapunov exponent becomes positive for a greater than approximately 0.89, indicating chaotic behavior.

Figure 4.1 shows the LE of the logistic map. The LE is positive for the bifurcation parameter between 0 to approximately 0.85, i.e., the region is a periodic region. On the other hand, for bifurcation parameters greater than 0.85, the LE is positive, which determines the chaotic region.

To conclude, the Lyapunov exponent offers a quantitative measure of a system's sensitivity to initial conditions, as illustrated by the logistic map example. By analyzing the Lyapunov exponent, we can gain insights into dynamical systems' stability, periodicity, and chaotic nature.

4.4 Bifurcation Diagram

A bifurcation diagram [139] is a graphical representation used to illustrate the long-term behavior of a dynamical system as a function of a control parameter. In the context of chaotic systems, bifurcation diagrams are particularly useful for understanding the onset of chaos and the transitions between different dynamical behaviors. This section will explain bifurcation diagrams, focusing on the features, construction, interpretation, and importance of studying chaotic systems. Key features of bifurcation diagrams include:

1. Fixed points: Represented as horizontal lines, fixed points are points in the phase space where the system remains stationary.

2. Periodic orbits: Represented by discrete points or line segments, periodic orbits are closed trajectories in the phase space where the system returns to its initial state after a certain period.

3. Bifurcation points: These are points where the system undergoes a qualitative change in behavior, such as a change in the number of fixed points or the onset of chaos.

4. Chaotic regions: These are areas in the diagram where the system exhibits chaotic behavior, often indicated by a dense distribution of points.

Interpreting a bifurcation diagram involves analyzing the various fixed points, periodic orbits, bifurcation points, and chaotic regions to understand the system's behavior as the control parameter changes. The diagram can reveal:

1. The onset of chaos: This is typically marked by a transition from periodic orbits to a chaotic region, which often occurs through a period-doubling bifurcation route.

2. Windows of periodicity: These are regions within a chaotic region where the system exhibits periodic behavior.

3. Hysteresis and multistability: Bifurcation diagrams can reveal situations where multiple stable states coexist for a single control parameter value.

Bifurcation diagrams are essential tools for studying chaotic systems for several reasons:

They visually represent the system's behavior, making it easier to identify transitions and different dynamical regimes. They allow researchers to identify the onset of chaos and the mechanisms responsible for the transition to chaos. They can reveal the existence

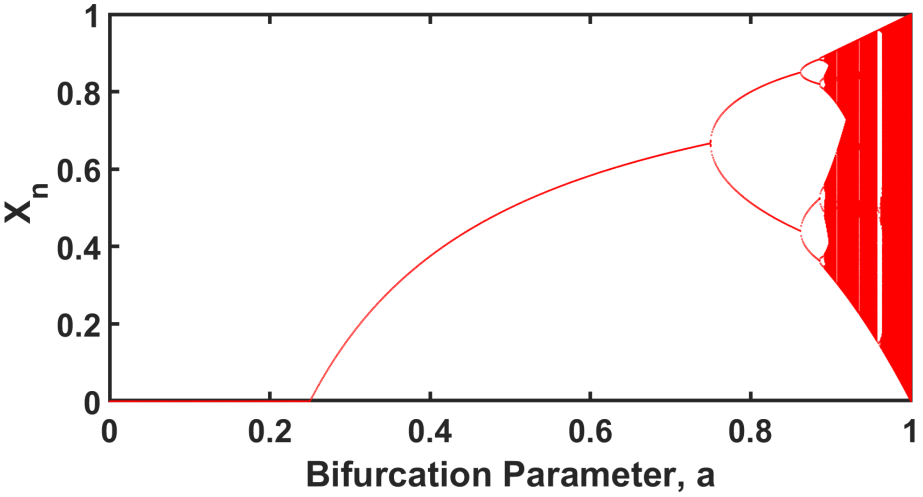


Figure 4.2: Bifurcation Diagram of Logistic map.

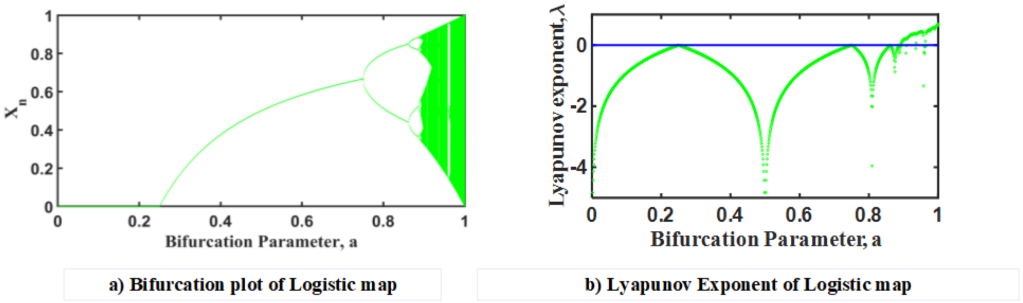
of multistable states, helping to understand the sensitivity of chaotic systems to initial conditions and parameter changes.

Figure 4.2 shows a bifurcation diagram of the logistic map. The dense region focuses on the chaotic region with positive LE.

To conclude, bifurcation diagrams are powerful tools for understanding the complex behavior of chaotic systems. By analyzing the different features and transitions in a bifurcation diagram, researchers can gain valuable insights into the long-term behavior of a chaotic system and the underlying mechanisms responsible for the onset of chaos. Including a bifurcation diagram and its interpretation in a thesis will provide a solid foundation for discussing the system's dynamical behavior under investigation.

4.5 Logistic Map

The Logistic map has various applications [194] and is a one-dimensional chaotic map. When the initial input value falls within the range of [0,1], this map generates an output sequence within the same range. The formula [48] for the Logistic map is expressed mathematically as follows:



a) Bifurcation plot of Logistic map

b) Lyapunov Exponent of Logistic map

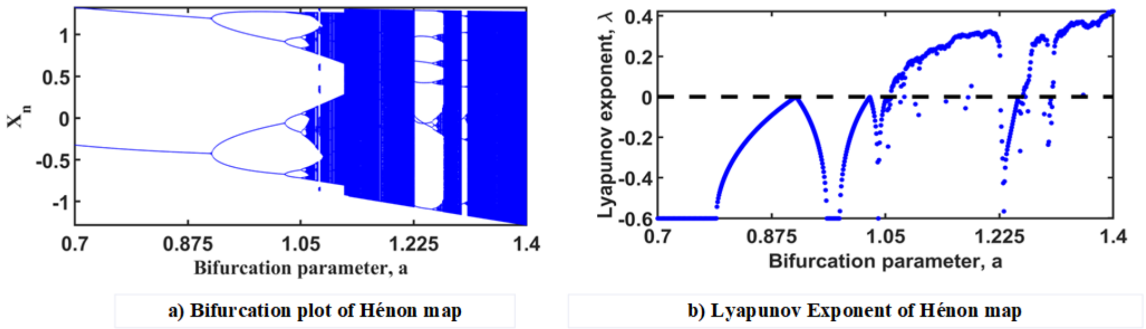
Figure 4.3: Logistic map characteristics analysis.

$$x_{n+1} = 4ax_n(1 - x_n) \tag{4.5.1}$$

In this context, the bifurcation parameter a varies from 0 to 1. The bifurcation diagram is displayed in Fig. 4.3 [140], along with the Lyapunov exponent (LE) plot for the logistic map [123]. Bifurcation diagrams involve creating and plotting multiple steady-state sequence sets with respect to the controlled parameter (a). The dark green area in Fig. 4.3(a) signifies the chaotic region corresponding to specific control parameter values [139]. In a chaotic sequence, a minor alteration in the initial condition causes two adjacent trajectories to diverge exponentially. The Lyapunov exponent, represented by λ, is a widely utilized metric to measure the sensitivity of the chaotic circuit to initial conditions. Periodic regions are indicated by negative LE values, while positive values denote chaotic regions [122]. The bifurcation diagram and positive LE values reveal that the logistic map exhibits chaos when the controlled parameter value lies between 0.89 and 1. For this study, we choose $a = 1$.

4.6 Hénon Map

Hénon map is a discrete-time 2D chaotic map that shows excellent chaotic behavior. It transforms a point (x_n, y_n) on the plan into a new point (x_{n+1}, y_{n+1}). The mathematical equation of the Hénon map is defined as follows:



a) Bifurcation plot of Hénon map

b) Lyapunov Exponent of Hénon map

Figure 4.4: Hénon Map characteristics analysis.

$$x_{n+1} = y_n - a(x_n)^2 \tag{4.6.1}$$

$$y_{n+1} = bx_n \tag{4.6.2}$$

Fig. 4.4 shows bifurcation and LE plots of the Hénon map. Steady-state values of x_n are shown in the bifurcation plot of fig. 4.4(a), whereas the maximum LE of the 2-D system is shown in Fig. 4.4(b) for $b = 0.3$. In this work, we select $a = 1.4$ and $b = 0.3$ [193].

4.7 Chaotic Time Series prediction using memristive RC system

This section showcased the prediction of 1D Logistic and 2D Hénon chaotic maps using a memristive reservoir computing (RC) system. By employing a controllable mask process (section 3.5), we enhanced the reservoir states within the network. The mask parameters allow us to fine-tune both the states' richness and the feedback's strength [7]. We have explored masking and nonmasking techniques in our analysis [58]. We have used four different types of memristors to build the reservoir. A brief description of those memristors has been given in table 4.1.

In this study, I generate a sequence of 2001 data points for both maps. The RC system for the training section predicts the first 1 to 1000 data points to predict the subsequent 2 to 1001 data points (with a one-step lag). This means that each data point is employed for the next step of data prediction. After the training section, I obtain readout weights that

Table 4.1: Introduction to different type of memristors.

Device	**Memristor Type**	**Device Materials**	**Operate Volt.(V)**
Memristor 1 [193]	Solid State Device	Ti/TiO_x / TaO_y / Pt	2.5
Memristor 2 [102]	Solid State Device	WO_x	1.8
Memristor 3 [195]	Solid State Device	SiO_2	3.5
Memristor 4 [107]	Biomolecular Device	Alamethicin	0.15

are then used to test the following 1000 sequences. To construct a network of 100 reservoir states, we select 25 memristors and a mask length of 4 for all memristors. Conversely, we employ 100 memristors for the network without masking.

For comparison purposes, NRMSE (Normalized Root Mean Square Error) has been used to measure the goodness of fit between the actual and the predicted signal [133], [53], which is described as:

$$NRMSE = \sqrt{\frac{mean\left\{(y(t)_{predicted} - y(t)_{actual})^2\right\}}{variance\left\{y(t)_{actual}\right\}}} \tag{4.7.1}$$

where $y(t)_{actual}$ and $y(t)_{predicted}$ are the original and predicted signal of the chaotic map, respectively. Table **??** shows the prediction results for all different RC architectures. The NRMSE values are calculated based on the 1000 data points of the actual and predicted signal for both maps.

Figure 4.5,4.6,4.7,4.8 shows the predicted results obtained by the RC system for 1D Logistic map and 2D Hénon map for 50-100 time steps. The NRMSE value is higher for the Hénon map as it is a 2D map with more complex chaotic dynamics than the 1D logistic map.

The results show that the RC system utilizing the mask process demonstrates a strong ability to predict chaotic time series. This can be attributed to the controllable mask process, which generates rich reservoir states by introducing added nonlinearity. Additionally, the masking process contributes to the input's sequentialization and optimizes the system's dimensionality usage [8]. Furthermore, the memristor possesses the capability to remember

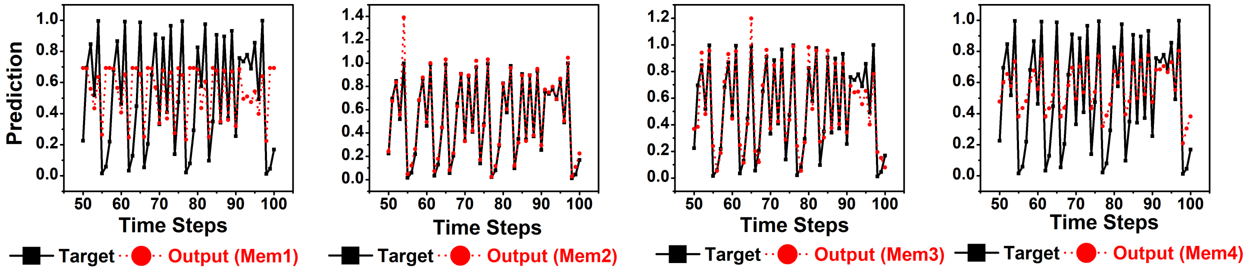


Figure 4.5: The predicted results obtained by RC system for 1D Logistic map (Without Mask).

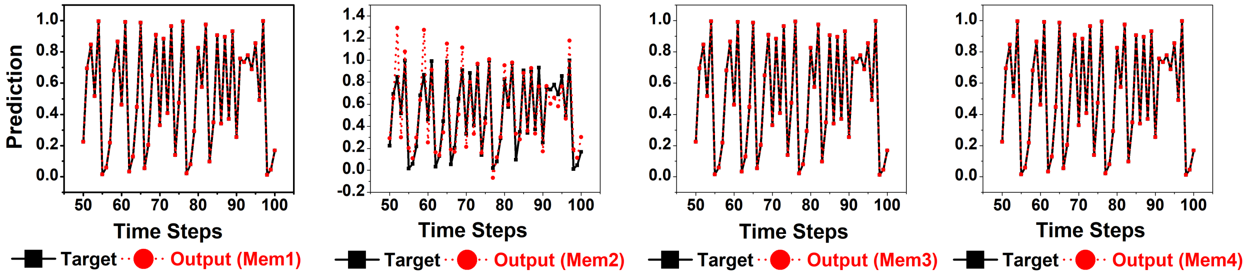


Figure 4.6: The predicted results obtained by RC system for 1D Logistic map (With Mask)

previous and current states. As a result, combining these techniques enhances non-linearity, which aids in predicting chaotic time series.

Furthermore, the efficiency of the memristor-based RC system varies with the mask length and memristors. Fig.4.9 4.10 show the performance of RC network variation with the mask length and the number of memristors. From Fig.4.9 4.10, it is clear that the performance of the memristor-based RC system varies with mask length and the number of memristors. In addition, as an example, we have explored more masked-based RC networks (mask length 5 and 7, number of memristors 30 and 40 for logistic map and Hénon map, respectively) with biomolecular memristor devices [107], where we varied operating voltage, pulse width time with duty cycle.

Fig. 4.11 4.12 shows a 3D scatter plot for 1D logistic map and 2D Hénon map prediction. We have found an optimum combination that gives the lowest NRMSE value of 6.21×10^{-6} for the logistic map and 5.31×10^{-3} for the Hénon map. Fig. 4.13 shows the final optimized results for both maps using a masked-based biomolecular memristive device.

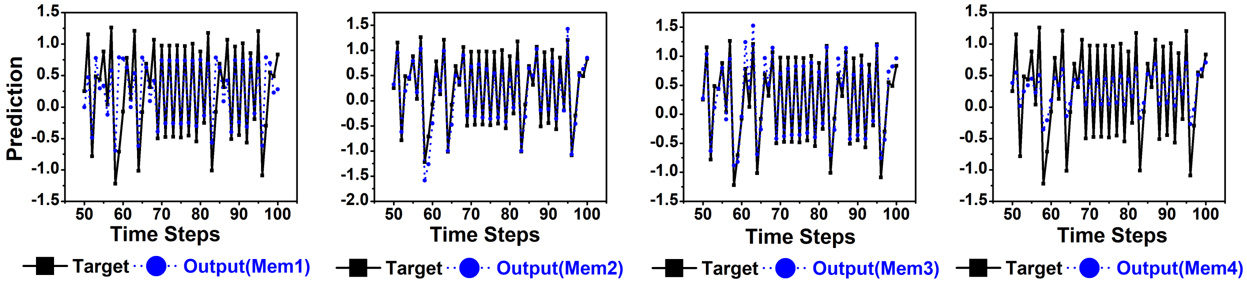


Figure 4.7: The predicted results obtained by RC system for 2D Hénon map (Without Mask)

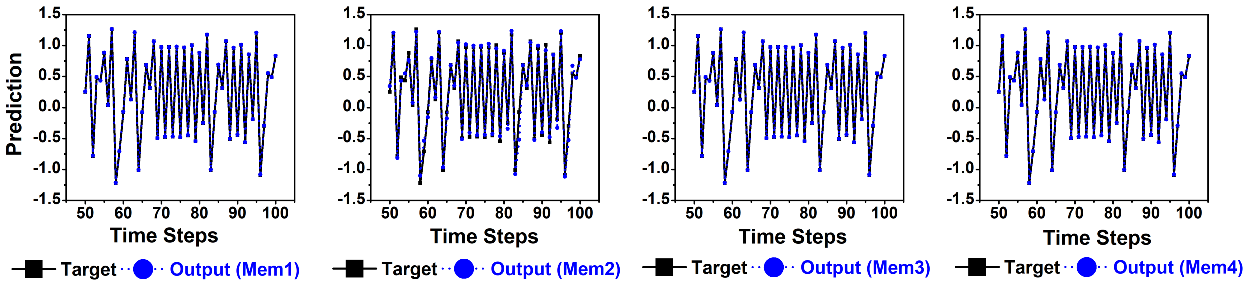


Figure 4.8: The predicted results obtained by RC system for 2D Hénon map (With Mask)

It is clear from the figure that the prediction matches very closely with the target output. Moreover, our future endeavor will be to explore different types of complex maps by changing different properties of the memristors.

As chaotic systems are very sensitive to initial conditions and aperiodic in nature, it is hard to predict the system for more delayed time steps. On [58] has shown that the performance worsens with the increasing delayed time steps. Even though the performance worsens with increased delay, the masked-based memristive RC system performs better than the non-mask system in most cases.

4.8 Cascaded Chaotic Maps

Cascaded chaotic maps are a mathematical concept used in studying chaos theory, a branch of applied mathematics that deals with nonlinear dynamic systems. These cascaded chaotic maps are formed by the composition of multiple individual chaotic maps applied in a sequential manner. The primary purpose of cascaded chaotic maps is to create more

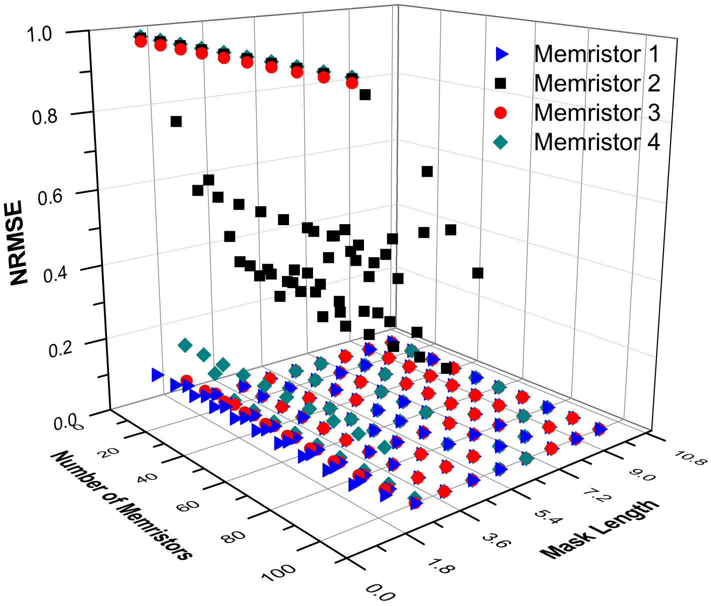


Figure 4.9: The NRMSE changes with the reservoir size in different masked memristive RC systems for predicting Logistic map

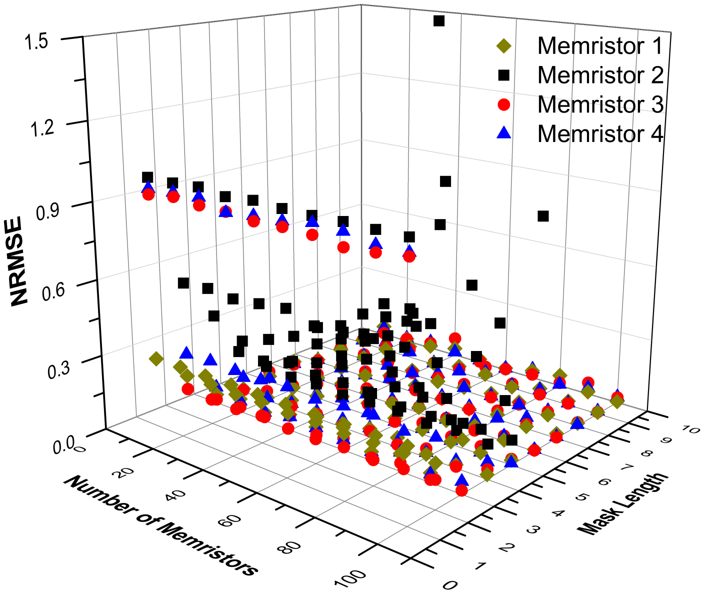


Figure 4.10: The NRMSE changes with the reservoir size in different masked memristive RC systems for predicting the Hénon map.

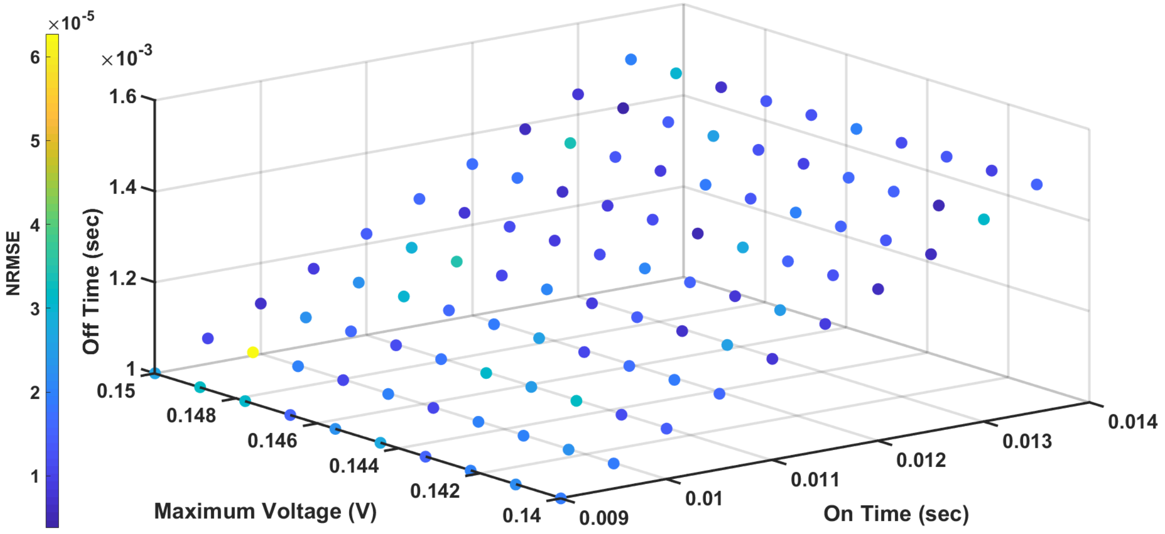


Figure 4.11: 1D Logistic map further exploration using biomolecular memristor-based RC system. The prediction error varies with the pulse width's voltage on and off time.

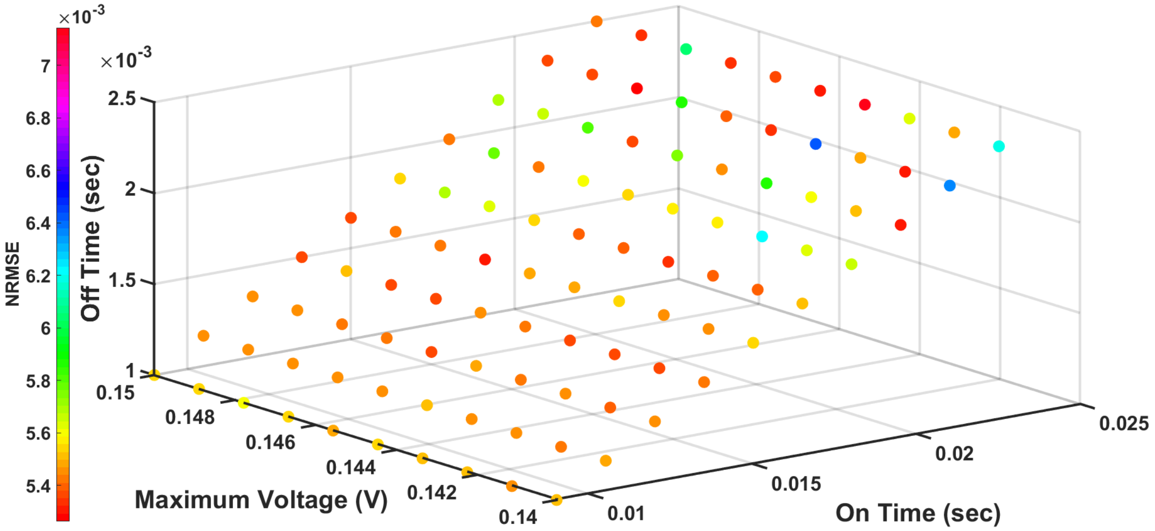
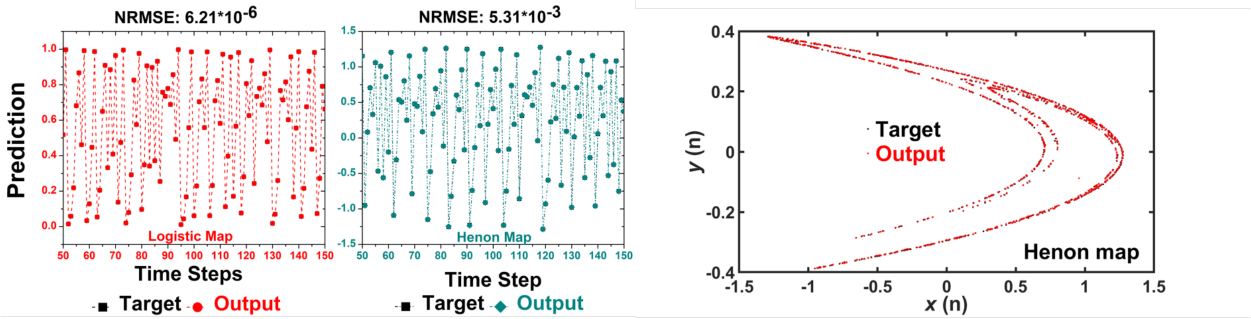


Figure 4.12: 2D Hénon map further exploration using biomolecular memristor-based RC system. The prediction error varies with the pulse width's voltage on and off time.



a) Logistic map and Hénon map prediction with improved parameters

b) Hénon map 2D trajectory

Figure 4.13: (a) 1D Logistic map and 2D Hénon map prediction with improved parameters using biomolecular memristor-based RC system. (b)2D display of predicted results of Hénon map.

complex and intricate chaotic behavior, which can be employed in various fields, such as cryptography, secure communication, and engineering.

A chaotic map is a discrete-time dynamical system that exhibits chaotic behavior characterized by high sensitivity to initial conditions and unpredictable long-term behavior. It is typically represented by a nonlinear function applied iteratively to an initial state. Some well-known examples of chaotic maps are the logistics, Hénon, and Arnold's cat maps.

In a cascaded chaotic map, the output of one chaotic map is fed as input to another chaotic map, creating a more complex transformation. This process can be repeated multiple times, forming a cascade of chaotic maps. The resulting cascaded chaotic map can be represented as:

$$x_{n+1} = f_k(...f_2(f_1(x_n))) \tag{4.8.1}$$

where x_n is the system's state at time n, and $f_i, i = 1, 2, ..., k$ are the individual chaotic maps in the cascade.

The main advantage of cascaded chaotic maps is that they generate richer and more diverse chaotic behavior than individual maps. By carefully selecting and combining different chaotic maps, researchers can create customized systems that exhibit desired properties, such as improved security and robustness.

Cascaded chaotic maps have been applied in various domains, including:

a) Cryptography: Due to their high sensitivity to initial conditions and unpredictable behavior, cascaded chaotic maps can be used to design secure encryption algorithms and pseudorandom number generators.

b) Secure communication: Cascaded chaotic maps can be employed in designing schemes for the secure transmission of information by exploiting their complex behavior to create robust and hard-to-predict communication signals.

c) Image processing and watermarking: The unpredictable nature of cascaded chaotic maps can be utilized for image encryption, steganography, and digital watermarking, providing a means for secure storage and transmission of images.

d) Engineering and control systems: The rich dynamics of cascaded chaotic maps can be harnessed in designing advanced control systems, such as adaptive controllers and chaotic synchronization systems.

Cascaded chaotic maps are an essential tool in studying chaos theory, providing a means to generate more complex and intricate chaotic behavior. Their application extends across multiple domains, including cryptography, secure communication, and engineering, with the potential to contribute significantly to the advancement of these fields.

Cascaded chaotic maps have been extensively researched due to their potential applications in various fields, including secure communication, image encryption, and optimization algorithms. The literature review highlights key contributions to understanding and developing cascaded chaotic map-based systems:

Chen et al. [82] investigates the dynamical degradation of digital piecewise linear chaotic maps and lays the groundwork for understanding the behavior of cascaded chaotic maps. Alvarez et al. [4] discusses the necessary cryptographic requirements for chaos-based cryptosystems, providing essential insights into developing cascaded chaotic map-based applications. Pareek et al. [117] explore the use of chaotic logistic maps for image encryption, setting the stage for further research into cascaded chaotic maps in this area. Li et al. [81]

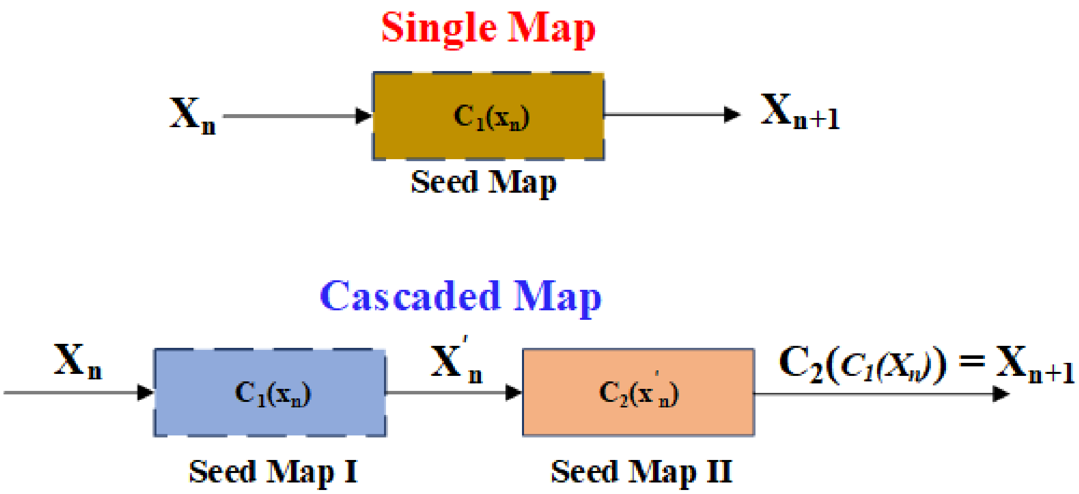


Figure 4.14: Schematic of cascading scheme

analyze the security of a chaos-noise-based secure communication scheme, providing insights into potential vulnerabilities in cascaded chaotic map-based communication systems. Lu et al. [81] introduce a new generalized Lorenz-like chaotic system, which can be a potential building block for cascaded chaotic maps and related applications. Lu et al. [13] propose a new image encryption scheme based on substitution-permutation networks and chaos, which can inform future research on cascaded chaotic map-based encryption techniques.

Figure 4.14 shows the seed and cascaded maps. For 1D and 2D seed maps, each seed map is connected to another to build the cascaded map.

4.9 Hierarchical RC architecture

To improve prediction performance, we have explored hierarchical reservoir architecture inspired by the article.[103] Several works have been proposed[37, 93, 89] to increase the richness of the reservoir. This work used wide reservoirs (independent sub-reservoirs connected parallel) and deep reservoirs (sub-reservoirs stacked in series). Figure 4.15 gives a schematic of wide RC architecture.

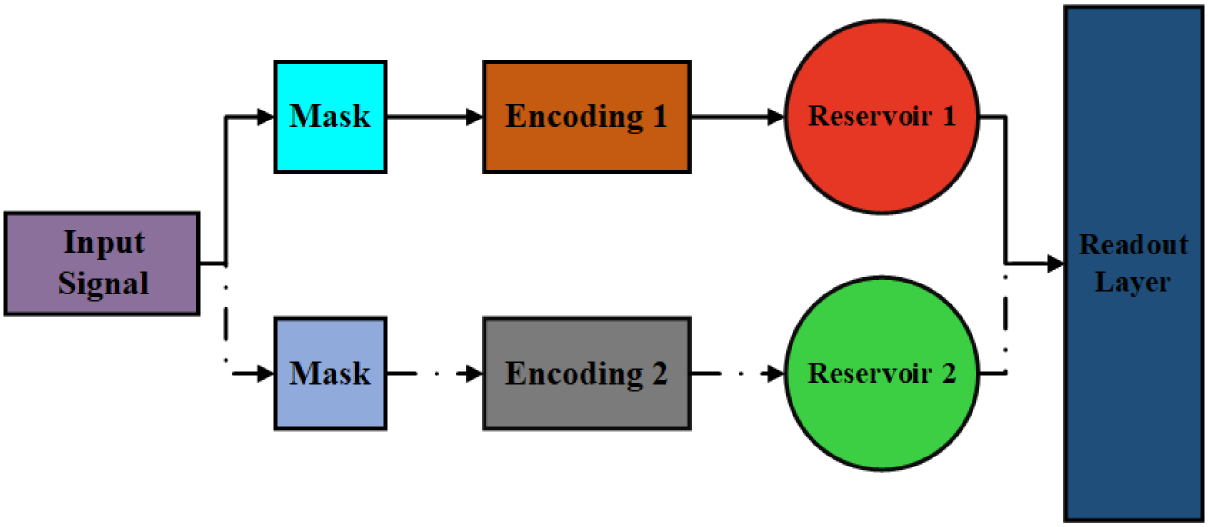


Figure 4.15: The schematic of the wide RC system, where two reservoirs are parallelly connected to the readout layer.

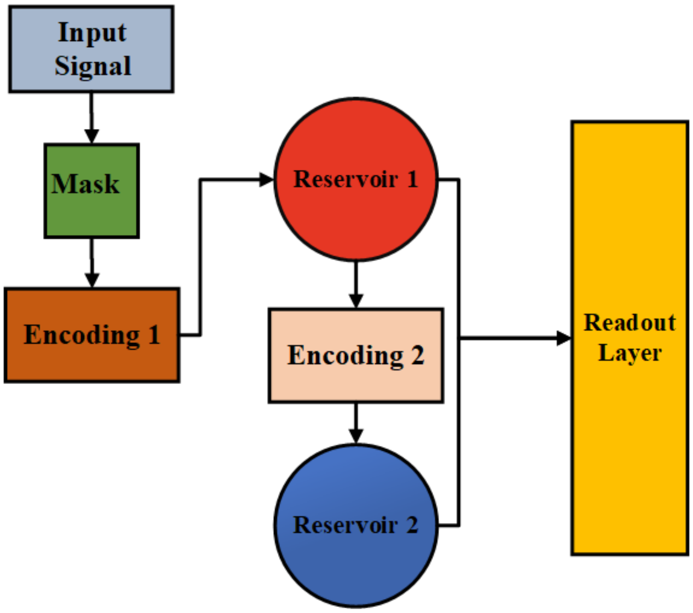


Figure 4.16: The schematic of the Deep RC system, where two reservoirs are series connected and reservoir 2 depends on the output of reservoir 1.

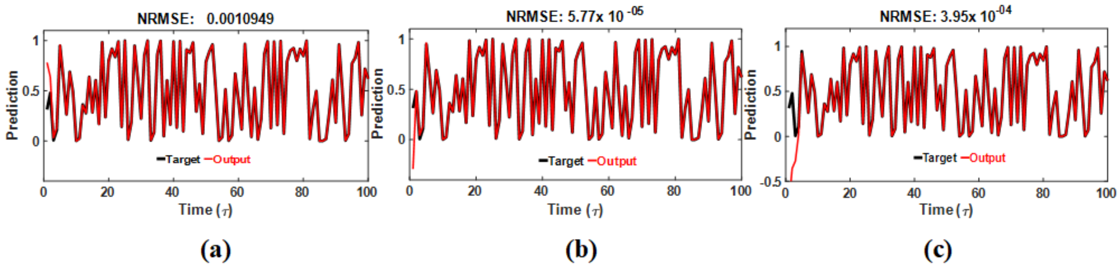


Figure 4.17: The target vs. output signal results obtained by biomolecular memristive RC network for cascaded 1D Logistic map. (a) Shallow Reservoir (b) Wide Reservoir (c) Deep Reservoir

In this architecture, different voltage encoding has been done for two parallel connections while keeping the mask parameters the same. In the readout layer, weights from both reservoirs are concatenated before performing the linear regression.

Figure 4.16 gives a schematic of deep RC architecture. Here the data has passed through the reservoir1 after the first voltage encoding. The second voltage encoding has been done based on the output of the first reservoir, and both reservoirs are connected with the readout layer. One point to note is that we have used half the number of memristors in each reservoir in the hierarchical reservoir to make a fair comparison between all architectures.

4.10 Cascaded Chaotic maps prediction using Hierarchical memristive RC architecture

We have conducted training and testing for 1D seed and cascaded logistic map and 2D seed and cascaded Hénon map using three RC architectures with two types of memristors [193, 107]. For the sake of comparison purposes, NRMSE (Normalized Root Mean Square Error) has been used in our analysis to measure the goodness of fit between the actual target signal and the predicted output signal.

Figure 4.17 to Figure 4.20 shows predicted results obtained by three RC architectures with two different types of memristors. From the figures, it is clear that the hierarchical reservoir shows better performance compared to the shallow reservoir architecture. It's because a

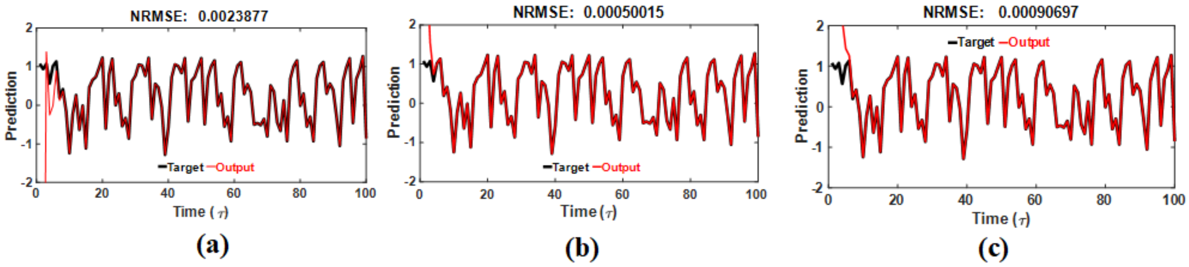


Figure 4.18: The target vs. output signal results obtained by biomolecular memristive RC network for cascaded 2D Hénon map. (a) Shallow Reservoir (b) Wide Reservoir (c) Deep Reservoir

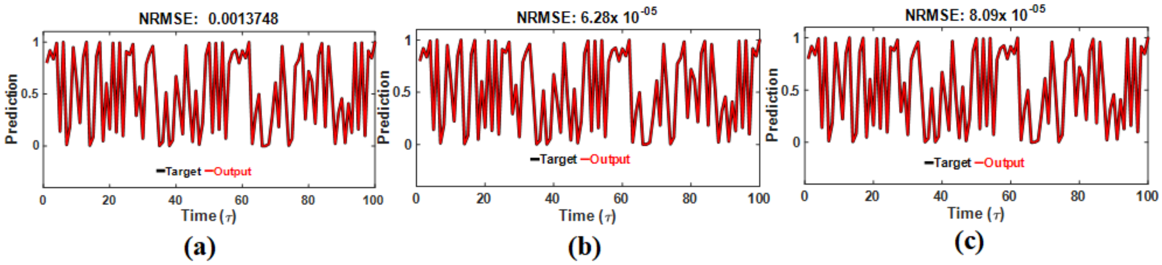


Figure 4.19: The target vs. output signal results obtained by solid-state memristive RC network for cascaded 1D Logistic map. (a) Shallow Reservoir (b) Wide Reservoir (c) Deep Reservoir.

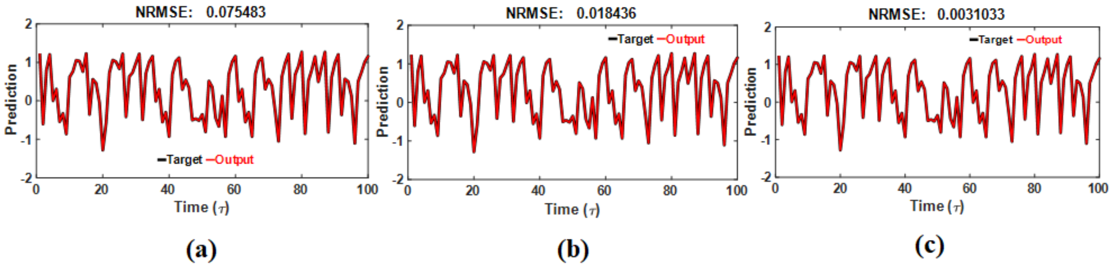
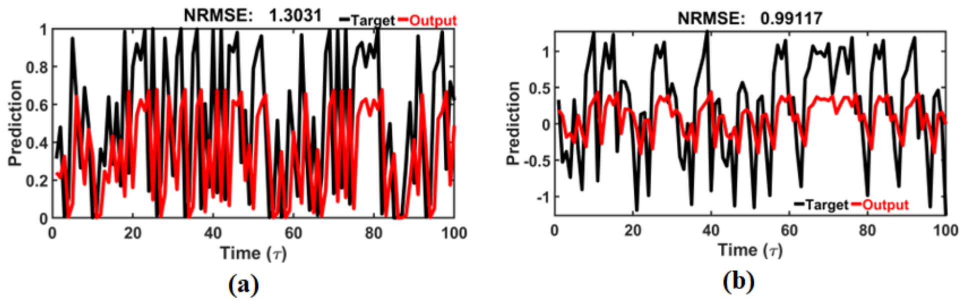
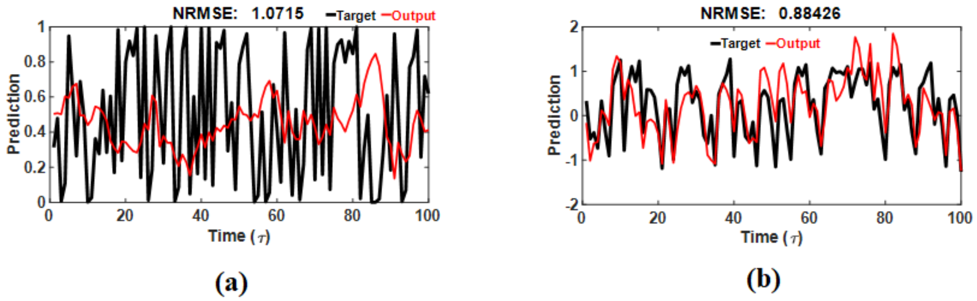


Figure 4.20: The target vs. output signal results obtained by solid-state memristive RC network for 2D cascaded Hénon map. (a) Shallow Reservoir (b) Wide Reservoir (c) Deep Reservoir.



(a) (b)

Figure 4.21: The target vs. output signal results obtained by the linear network for cascaded 1D Logistic map and cascaded 2D Hénon map.



(a) (b)

Figure 4.22: The target vs. output signal results obtained by Echo State Network for cascaded 1D Logistic map and cascaded 2D Hénon map.

hierarchical reservoir captures more diverse temporal dynamics than the conventional reservoir [103].

To investigate the nonlinear transformation required for the task, I have explored our work with a linear network to check the network's capability to predict chaotic time series. Here we have multiplied our original inputs with the following randomly scaling factors:

$$X_{linear} = rand(0,1) \tag{4.10.1}$$

In the case of a linear network, no nonlinear transform is provided by the reservoir. Figure 4.21 shows the prediction of the 1D cascaded logistic map and the 2D cascaded Hénon map using a linear network. From the figures, it is clear that nonlinear transformation and memory property is necessary, provided by the memristive RC network, to solve the chaotic time series problem. Moreover, we have also conducted the chaotic time series prediction

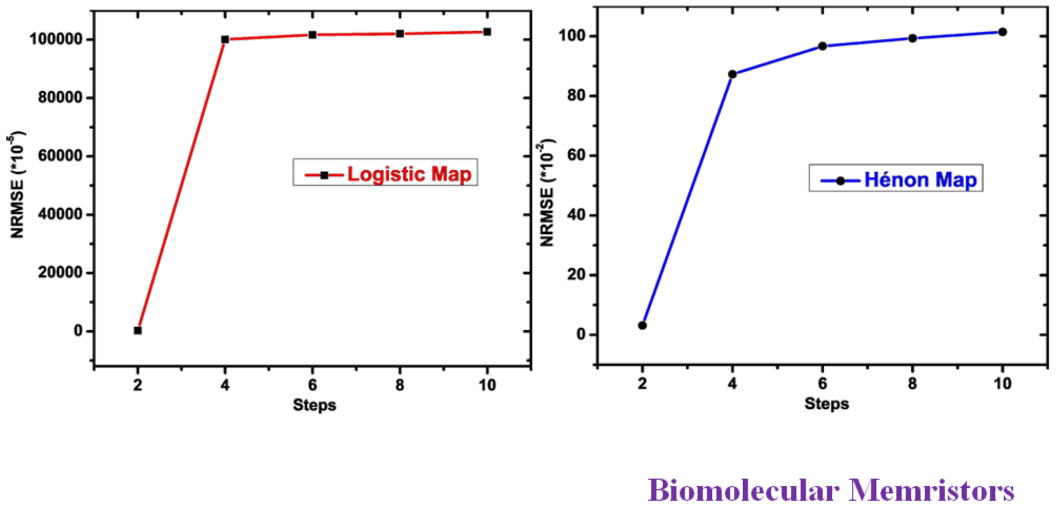


Figure 4.23: Limitations on Chaotic time series predictions.

with the Echo State Network (ESN)[133, 25]. Here we have taken the same number of reservoir nodes as the conventional physical reservoir. Figure 4.22 shows the prediction of the 1D cascaded logistic map and the 2D cascaded Hénon map using a linear network.

4.11 Limitations

The illustration found in Figure 4.23 expounds on the boundaries of the research conducted. It demonstrates one-step delayed prediction for two types of maps: the Logistic and Hénon maps. However, an intriguing phenomenon is observed when the time step delay is extended beyond a single step. It results in a considerable elevation in the NMSE (Normalized Mean Square Error) value, which can be mainly attributed to the amplified error rate. This substantial error rate is primarily due to the chaotic system's inherently aperiodic nature and extreme sensitivity.

Delving into the specifics, this research presents the NMSE margin escalation as the time steps increment from 2 to 10. This progression is exclusively depicted for the 1D Logistic map and the 2D Hénon map in the context of a biomolecular memristor-based RC (Reservoir Computing) system. The amplification of the error rate is of particular interest, as it poses a formidable challenge to the predictive ability of the system. It's worth noting that the data and insights presented here are constrained to the biomolecular memristor-based RC system.

For additional perspective, other devices are subjected to the same analysis. However, the outcomes and discussions related to these devices are beyond the scope of this document. Readers interested in a broader understanding of the NMSE margin increases across different devices are referred to the detailed report in [58]. This referenced work provides further insights and contributes to a more holistic comprehension of the performance and limitations of various systems in this domain.

5 STATIC DATASET CLASSIFICATION

5.1 IRIS dataset Classification

The IRIS dataset is a widely recognized, versatile, and easy-to-use benchmark dataset in the field of machine learning and data science. Introduced by the British statistician and biologist Ronald A. Fisher in 1936, the dataset comprises 150 instances with four key features: sepal length, sepal width, petal length, and petal width, all measured in centimeters. These features correspond to three distinct species of the iris flower: Iris setosa, Iris versicolor, and Iris virginica, each represented by 50 instances in the dataset [63]. As a result, the IRIS dataset serves as an excellent starting point for beginners to learn classification algorithms, data visualization techniques, and exploratory data analysis. The dataset's simplicity and well-documented history have cemented its position as a cornerstone for the initial steps of machine learning education and experimentation.

Figure 5.1 illustrates the various species within the iris dataset, comprising 150 data points. I allocated 60% of the data for training and the remaining 40% for testing. The analysis was conducted using biomolecular memristors and memcapacitors to construct the reservoir. Initially, the attributes were transformed into voltage pulse trains through the voltage encoding block. The conversion of voltage for biomolecular memristors ranged from 100 mV to 170 mV with a 15ms pulse period. In addition, for biomolecular memcapacitors, the voltage conversion spanned from 100 mV to 200 mV, featuring a 2.5s pulse period. Four weights were obtained simultaneously from the reservoir for every data sample and subsequently fed through a 4x3 readout layer. This readout layer was trained using logistic regression techniques. The process flow of the iris dataset utilizing a memcapacitive-based

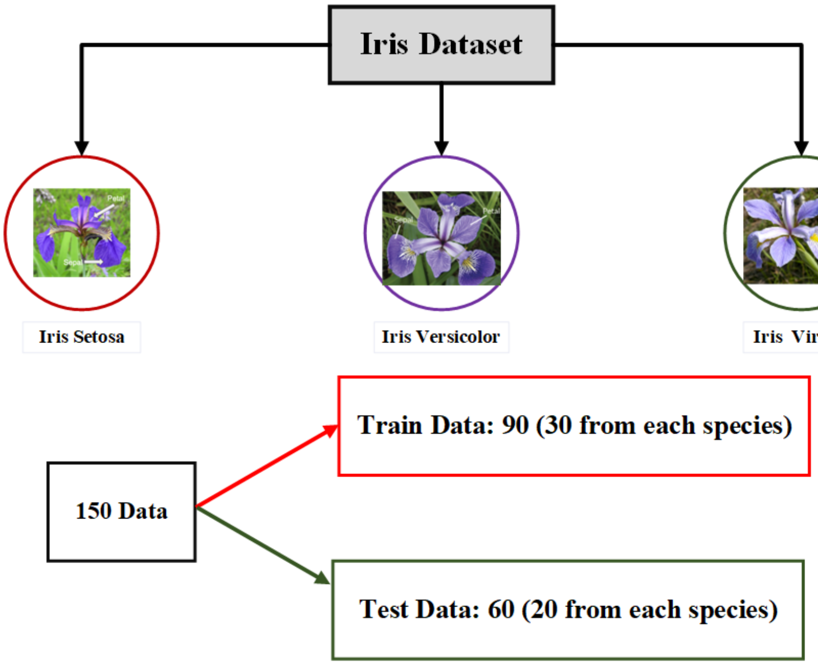
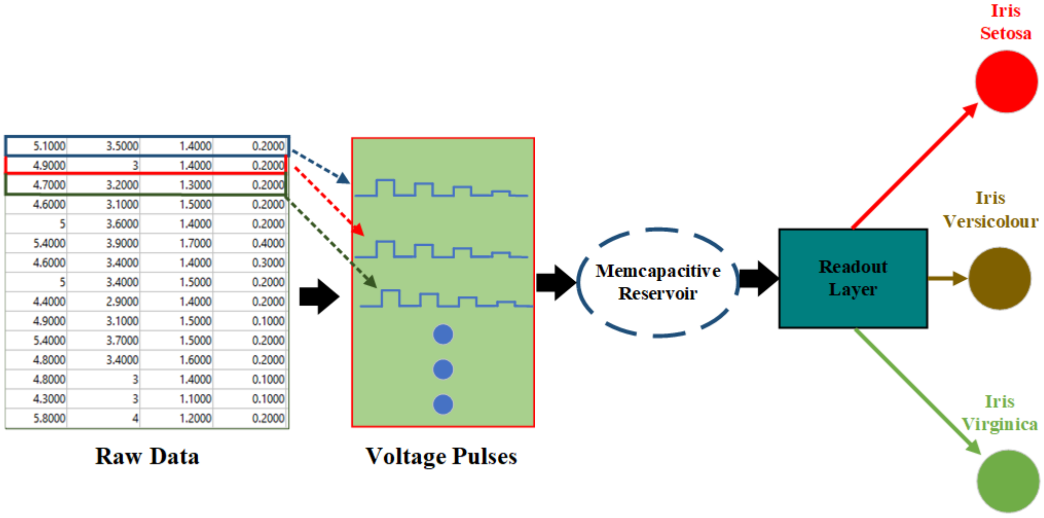


Figure 5.1: Iris dataset; train and test data separation.



5.1000	3.5000	1.4000	0.2000
4.9000	3	1.4000	0.2000
4.7000	3.2000	1.3000	0.2000
4.6000	3.1000	1.5000	0.2000
5	3.6000	1.4000	0.2000
5.4000	3.9000	1.7000	0.4000
4.6000	3.4000	1.4000	0.3000
5	3.4000	1.5000	0.2000
4.4000	2.9000	1.4000	0.2000
4.9000	3.1000	1.5000	0.1000
5.4000	3.7000	1.5000	0.2000
4.8000	3.4000	1.6000	0.2000
4.8000	3	1.4000	0.1000
4.3000	3	1.1000	0.1000
5.8000	4	1.2000	0.2000

Figure 5.2: The process flow of Iris Dataset classification using a memcapacitor-based RC system.

Table 5.1: Comparitive analysis between biomolecular memristors and memcapacitors for classifying Iris dataset.

Devices	**Vol. Range, (mV)**	**Pulse Width**	**Test Accuracy,%**
Bio. Memristor	100 to 170	15 ms	96.67
Bio. Memcapacitor	100 to 200	2.5 s	98.33

RC system is depicted in Figure 5.2. The testing accuracy for biomolecular memristors and memcapacitors reservoirs are 96.67% and 98.33%, respectively 5.1.

5.2 MNIST Image Classification

MNIST Image Classification is a fundamental task in the field of computer vision and machine learning, designed to recognize handwritten digits (0 to 9) using the Modified National Institute of Standards and Technology (MNIST) dataset. The dataset comprises a collection of 70,000 grayscale images, each 28x28 pixels in size, which are split into 60,000 training and 10,000 testing samples. As a benchmark problem, MNIST image classification has served as a stepping stone for researchers and practitioners, enabling them to test, validate, and fine-tune a variety of algorithms, such as deep learning models like convolutional neural networks (CNNs), as well as classical machine learning approaches like support vector machines (SVMs) and k-nearest neighbors (k-NN). The widespread adoption and success of the MNIST dataset has fueled advances in pattern recognition, laying the foundation for more complex real-world applications in areas such as object detection, facial recognition, and autonomous driving.

LeCun et al. [77] introduced the LeNet-5 architecture, an early example of a convolutional neural network (CNN), which achieved a 0.95% error rate on the MNIST dataset. The authors presented the backpropagation algorithm as an effective means for training the network and provided insights into how gradient-based learning could be applied to document recognition tasks. Cortes and Vapnik [24] proposed the Support Vector Machine (SVM) as a powerful classification method. The paper demonstrated the effectiveness of SVMs in separating data into distinct classes, including its application to the MNIST dataset. SVMs have since become a popular approach for tackling image classification problems. Simard et al. [153] presented best practices for applying CNNs to visual document analysis, focusing on the MNIST dataset. The authors introduced elastic distortion as a data augmentation technique, which significantly improved the performance of CNNs on the dataset, achieving a 0.7% error rate. This paper by Hinton and Salakhutdinov [54] demonstrated the efficacy of deep learning techniques in dimensionality reduction. The authors used a deep autoencoder to learn a compact representation of the MNIST dataset and showed that this approach could be effectively combined with other classification methods, such as SVMs or k-nearest neighbors, to improve overall performance. Cireşan et al. [22] proposed multi-column deep neural networks (MCDNNs) for image classification, which combined multiple CNNs to produce an ensemble model. The authors demonstrated that MCDNNs achieved state-of-the-art results on the MNIST dataset, with an error rate as low as 0.23%. Wan et al. [168] introduced DropConnect, a novel regularization technique for neural networks, extending the idea of dropout. The authors applied DropConnect to the MNIST dataset and observed improved performance, suggesting that it could be useful for preventing overfitting in deep learning models. Goodfellow et al. are [43] work is relevant to MNIST image classification as it demonstrated how deep convolutional neural networks could be used for recognizing multi-digit numbers. The authors leveraged their experience with the MNIST dataset to develop a model that performed well on the more complex SVHN dataset. Srivastava et al. [155] proposed dropout, a widely-used regularization technique for neural networks. The authors demonstrated the effectiveness of dropout by applying it to various datasets, including MNIST. The technique proved valuable for mitigating overfitting in deep learning models, particularly in image classification tasks. Ioffe and Szegedy introduced [61] batch normalization, a technique for improving deep neural networks' training speed and performance. The authors demonstrated the effectiveness of batch normalization on several datasets, including MNIST, and showed that it could significantly accelerate training while improving classification accuracy. Li et al. [78] presented an in-situ learning approach for

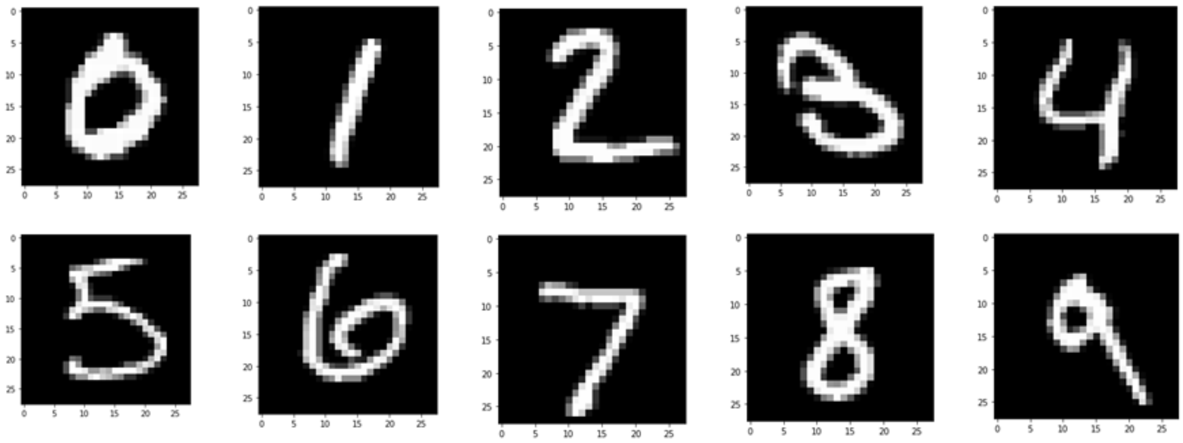


Figure 5.3: Schematic of MNIST digits. Each digit is 28*28 pixels.

multilayer memristor neural networks. The authors demonstrated their method's effectiveness on the MNIST dataset and highlighted the potential for memristor-based hardware implementations in deep learning tasks. Du et al. [30] explored the implementation of synaptic functions using oxide memristors, simulating various synaptic behaviors. Although not directly addressing MNIST image classification, this work provided insights into how memristor-based reservoir computing could be developed in the future. Pedretti et al. [125] presented a memristive neural network based on spike-timing-dependent plasticity (STDP), showcasing the network's ability for online learning and tracking. While not explicitly applied to the MNIST dataset, this research demonstrates the potential of memristor-based neural networks for pattern recognition tasks.

Figure 5.3 shows the schematic of MNIST images from 0 to 9. Initially, these grayscale images (28x28 pixels) undergo a conversion to binary format, simplifying the data and facilitating further processing. To enhance computational efficiency, redundant borders surrounding the digits are eliminated, effectively reducing the dimensions of each image to 22x20 binary pixels, with 22 rows and 20 pixels per row.

Following this preprocessing, each row of 20 pixels is segmented into four distinct sections, each containing 5 pixels. To effectively encode these sections, we generate four pulse trains for every image row, with each pulse train corresponding to one of the four

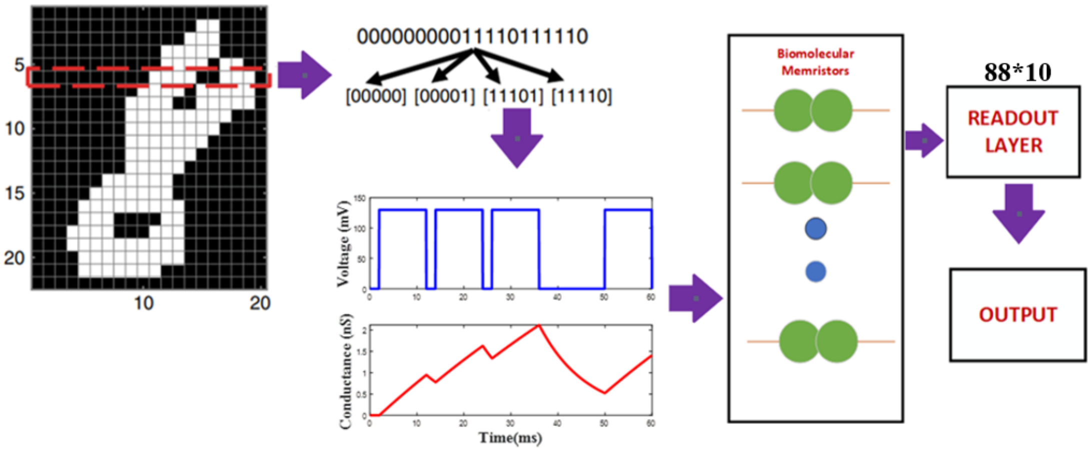


Figure 5.4: Training process of MNIST dataset using biomolecular memristive based RC system.

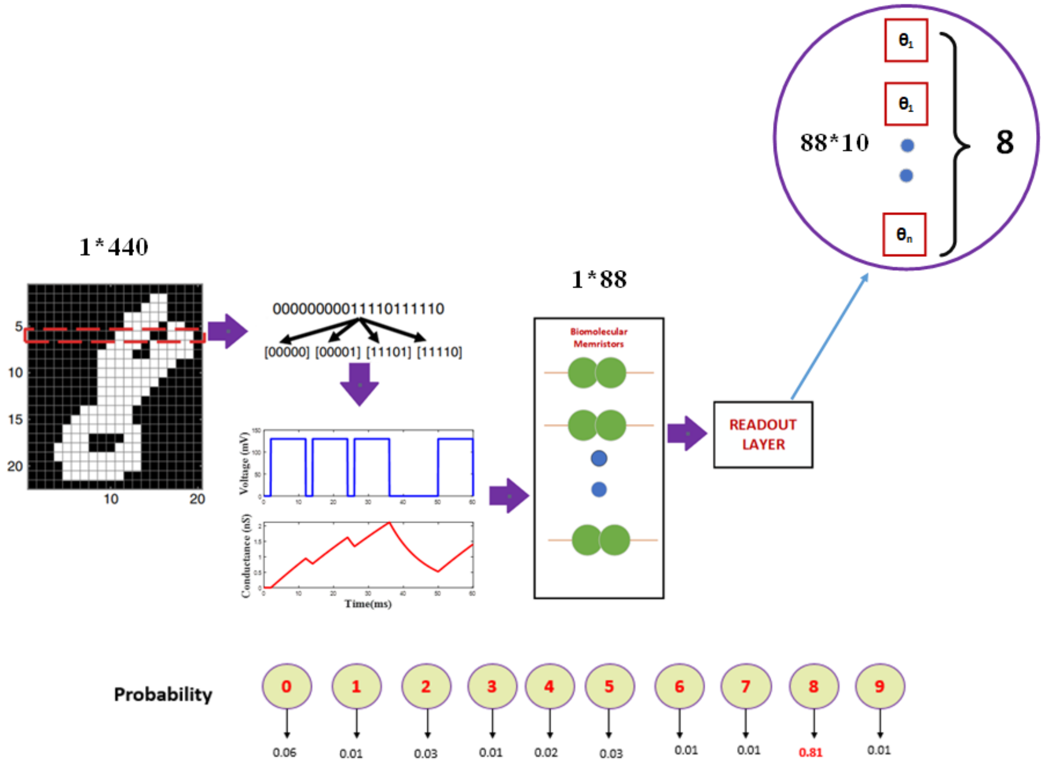


Figure 5.5: Testing process of MNIST dataset using biomolecular memristive based RC system.

sections. Within each pulse train, five pulses directly represent the five binary pixels in the corresponding section.

These pulse trains are subsequently input into a memristor-based reservoir comprising 88 memristors, where each memristor is associated with one of the four sections for each of the 22 rows. It is crucial to ensure that the time frame (i.e., the period of the pulses) is not excessively large relative to the time constant of the dominant state variable. This ensures that the increased conductance caused by each pulse does not undergo significant decay before the arrival of the subsequent pulse. As a result of this design, the final conductance of each memristor is influenced by the unique sequence of pulses in its input, effectively leveraging the inherent memory properties of memristors.

Figure 5.4 illustrates the comprehensive procedure for handwritten digit classification employing a memristor-based reservoir computing (RC) system (The MNIST image of digit eight has been taken from [29]. Prior to inputting the image data into the reservoir, a preprocessing stage is conducted to prepare the data for efficient processing. The preprocessed image data is divided into 5-pixel sections, which are then converted into input pulse streams and supplied to the reservoir at various rates. These rates are determined based on the pulse's on-time and off-time durations.

Once the data undergoes processing within the reservoir layer, it is subsequently directed to the readout network. Logistic regression is applied to analyze and classify the processed data in this stage. In our primary investigation, we utilized pulse streams with identical timeframe widths for application to the corresponding memristors. As a result, we obtained 88 synaptic weights, which are defined as the conductance end values subsequent to each time frame. An 88x10 readout layer is employed to perform the classification task, which leverages the synaptic weights to effectively distinguish between the various handwritten digits.

The digit with the highest probability is subsequently displayed as the final output. As illustrated in Figure 5.5, the probability for a specific image is higher compared to other

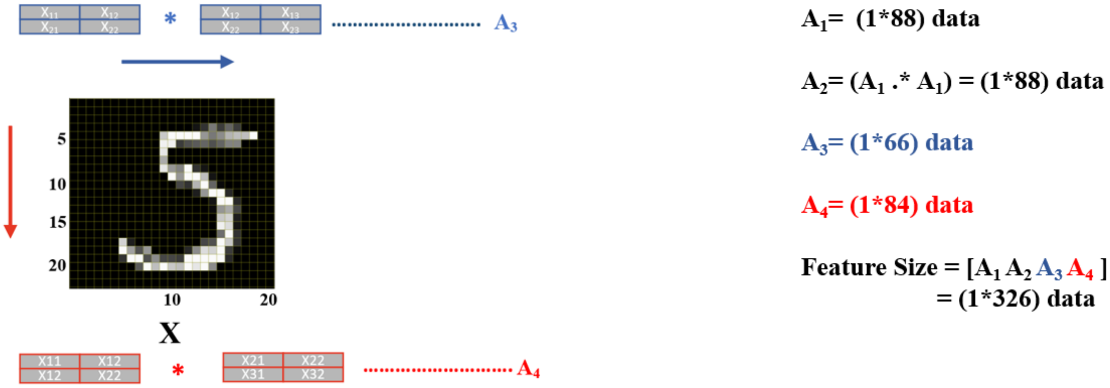


Figure 5.6: Feature modification of the image by row and column scanning.

cases, and thus, it is selected for display in the output. This demonstrates the effectiveness of the memristor-based reservoir computing system in accurately classifying handwritten digit images from the MNIST dataset.

Further Improvement:

In an effort to enhance the prediction accuracy, I expanded the feature size through raw data modification. As illustrated in Figure 5.6, the image undergoes feature alteration. Initially, I utilized one data set (88 features) and generated a second set by doubling the first features. I employed a column-by-column left-to-right scanning approach for the third set, resulting in an additional 66 features. Subsequently, I scanned the image row by row, from top to bottom, yielding 84 more features. Consequently, I incorporated a total of 326 features from a single dataset for classification purposes.

Results:

Table 5.2 presents the MNIST image classification results, detailing the system's performance for each case. In these cases, a test dataset comprising 10,000 images was employed for classification. As evidenced by the table, employing feature size in the context of a biomolecular memristor-based reservoir computing system contributes to improved data classification compared to other cases. This observation highlights the importance of feature

Table 5.2: MNIST Test accuracy for different models.

Model Name	**Test Accuracy,%**
Decision Tree	91.99
Support Vector Machine	92.97
Naïve Bayes	56.5
K-nearest neighbors algorithm	94.24
Logistic Regnression	90.89
Biomolecular Memristor (Simulation)	88.66
Biomolecular Memristor (Experiment)	88.02
Biomolecular Memristor (Simulation,extended features)	**94.46**
Solid State Memristor (Simulation) [29]	88.2

size selection in enhancing the performance of neuromorphic computing systems for image classification tasks.

6 TEMPORAL DATA PREDICTION AND CLASSIFICATION

6.1 Solving a second-order nonlinear dynamic task

Nonlinear dynamical systems are mathematical models that describe the complex behavior of natural and engineered systems characterized by time-dependent interactions among their constituent elements. Studying these nonlinear systems has proven invaluable in elucidating various phenomena in various disciplines, such as fluid dynamics, engineering, population biology, climatic systems, and neural networks. Governing equations of second-order nonlinear dynamical systems incorporate second-order time derivatives. Some prominent examples of second-order nonlinear dynamics within these domains include electrical system inverters and spring and damping properties in mechanical systems, among others. This section has selected a second-order dynamic nonlinear transfer function documented in [29]. The transfer function is shown below:

$$y(t) = 0.4y(t-1) + 0.4y(t-1)y(t-2) + 0.6u^3(t) + 0.1 \tag{6.1.1}$$

The output of the signal $y(t)$ depends on the present input $u(t)$ as well as on the previous two inputs $y(t-1)$ and $y(t-2)$(time lag of two-time steps) as shown in Eq. (6.1.1). This study aims to train a biomolecular memcapacitor-based RC system to map the hidden nonlinear transfer function, thereby enabling the generation of accurate output from the input after training without prior knowledge of the underlying mathematical relationship between input and output.

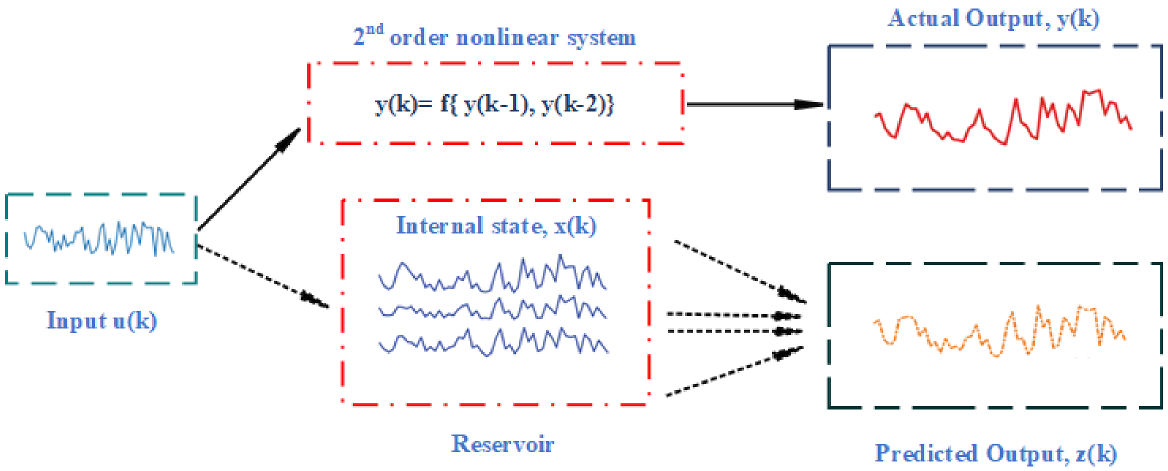


Figure 6.1: Process flow of solving second order nonlinear dynamic task.

This study aims to train a biomolecular memristor-based RC system to identify the concealed nonlinear transfer function, enabling accurate output derivation from the input after training without knowing the original mathematical correlation between input and output. Figure 6.1 demonstrate solving second-order nonlinear dynamic task. We employed a random input signal sequence within a 0 to 0.5 range and transformed it into voltage amplitudes. A random sequence of 300 timeframes is used to train the memristor-based RC system. The reservoir comprises 90 memristive states, divided into ten groups with nine devices each. Input voltages are provided through pulse streams with ten distinct timeframe widths (1 ms, 3 ms, 5 ms, 7 ms, 9 ms, 11 ms, 13 ms, 15 ms, 17 ms, and 19 ms) respectively, applied to the ten groups during testing. We discovered that having nine devices per group with slightly varying properties enhances reservoir performance due to inherent device variations, which aid in making the reservoir output more distinguishable. A similar performance improvement was observed with inputs having 10 different timeframe widths. In this instance, the readout layer is a 90 × 1 feedforward layer, which converts the reservoir state into a single output. A straightforward linear regression training algorithm based on batch gradient descent is employed to train the readout layer weights.

Initially, the input signal is transformed to a voltage range between 120 mV and 160 mV. These altered voltage signals serve as the amplitude for the pulse sequence applied

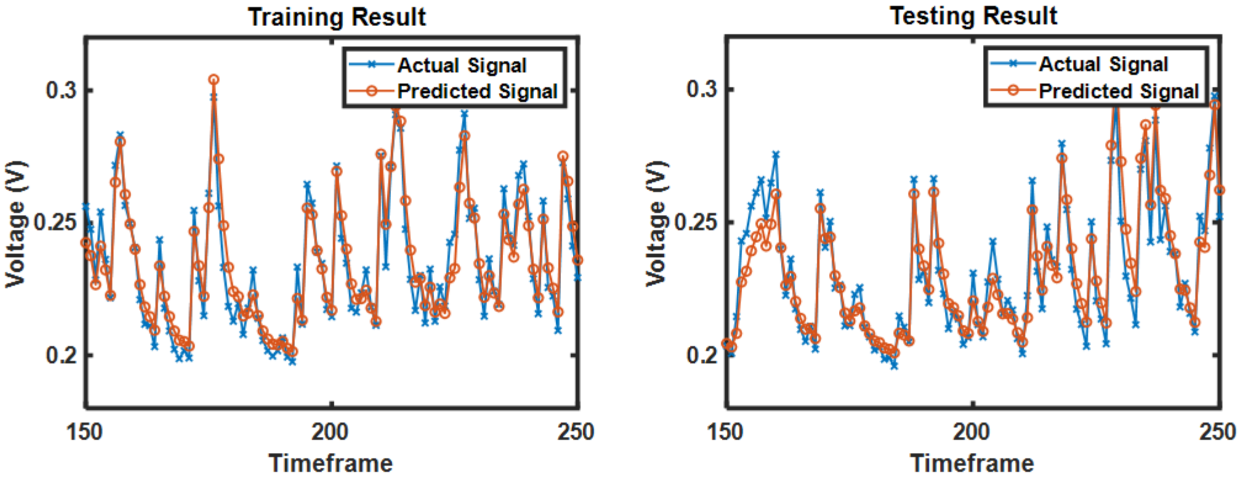


Figure 6.2: Solving a second-order nonlinear dynamic task. Actual signal vs predicted signal for training and testing data.

to the memristor. A 90% duty cycle is employed for each time frame. A linear regression model featuring least-squares learners is utilized in the readout layer. The signal error value is calculated using the Normalized Mean Squared Error (NMSE) method 6.1.2, [29].

$$NMSE = \frac{\sum_i (z_i(t) - y_i(t))^2}{\sum_i {y_i}^2(t)} \tag{6.1.2}$$

Where, $z(t)$ is the predicted signal and $y(t)$ is the actual signal. As the actual signal normalizes the result, the error is unitless. The NMSE values for training and testing are 0.0010 and 0.0011, respectively. Figure 6.2 compares the actual signal and the predicted signal outcomes for both training and testing datasets, using 90 memristive reservoir states.

Now, we present a comparative analysis between the memristor-based reservoir computing (RC) network and the traditional linear network, highlighting the influence of the memristor device's inherent nonlinear physics. We substitute the memristor reservoir layer with a linear hidden layer, generating 90 random signals derived from the original input, in accordance with the subsequent equation:

$$x(k) = rand(1, 100) * u(k) \tag{6.1.3}$$

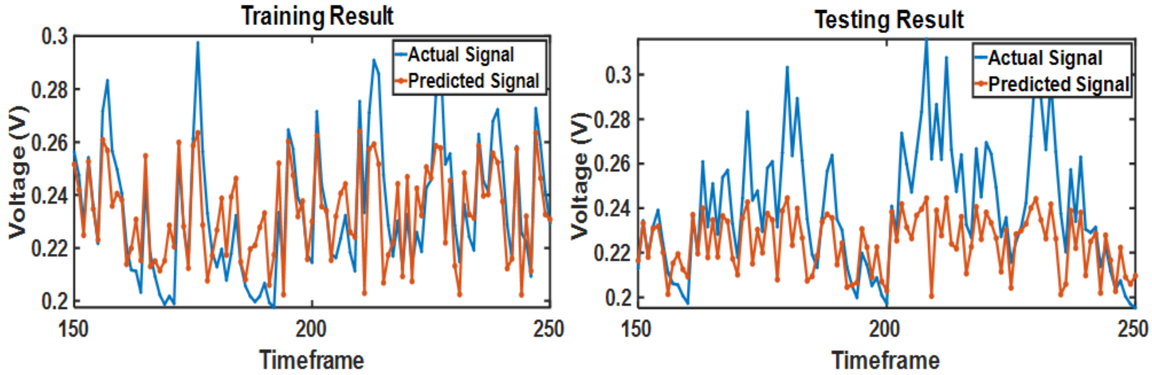


Figure 6.3: Simulating the prediction of a second-order nonlinear dynamic task using the conventional linear network. Actual signals vs. predicted signals for both training and testing datasets.

Table 6.1: Comparison work of solving second order nonlinear dynamic task problem.

Devices/Network	Train (NMSE)	Test (NMSE)	Reservoir States
Biomolecular Memristor	0.001	0.0011	90
Biomolecular Memcapacitor	0.00052	0.00063	50
Linear Network	0.0040	0.0043	90

where $x(k)$ is the scaled output from the linear resistor instead of the memristor, and $u(k)$ is the input vector. In this particular case, no nonlinear transformation is done by the reservoir.

Figure 6.3 illustrates the performance plot of a conventional linear network when tackling a second-order nonlinear dynamic task. The signal fitting is suboptimal, with elevated error levels, particularly in the testing dataset. The NMSE values for training and testing datasets, calculated using Equation 6.1.2, are 0.0051 and 0.0071, respectively. This comparison underscores the significance of the memcapacitor device's inherent nonlinear physics for effectively performing nonlinear transformations. A comprehensive overview of the calculated NMSE values in simulation is reported in Table 6.1. In our attempt to address this issue utilizing an experimental platform with a biomolecular memcapacitor-based RC system, we obtained NMSE values of 0.00057 during the training phase and 0.00078 during the testing phase.

6.2 Autonomous Prediction of Mackey Glass time series

Nowadays, predicting time series in chaotic systems is a difficult task due to the presence of positive Lyapunov exponents in these systems. This leads to an exponential increase in the separation of nearby trajectories, causing even slight inaccuracies in the initial state to result in a considerable deviation between the forecast and the actual signal after just a few time steps. In this experiment, we employed a biomolecular memristor-based RC system to predict the well-known Mackey-Glass time series [102], which follows a specific equation;

$$\frac{dx}{dt} = \beta \frac{x(t-\tau)}{1+(x(t-\tau))^n} - \gamma x(t) \tag{6.2.1}$$

In our experiment, we set n= 10, γ= 0.1, β= 0.2, τ= 18 in Eq. (6.2.1) to obtain chaotic dynamics. Though the chaotic system exhibits a deterministic structure, its highly sensitive nature to minor disturbances in the initial condition makes the prediction task extremely difficult. The objective is to train the memristor-based RC system to represent the concealed nonlinear transfer function. This enables the accurate derivation of output from input after training without knowledge of the original relationship between them. Initially, we generated a signal for 650 timeframes using a specific initial condition and normalized the signal values to a range between -0.5 and 0.5.

Figure6.4 illustrates the training phase data process employed to address the problem. The initial fifty data points are utilized to predict the subsequent value, and this pattern continues. As a result, the first 500 time steps are dedicated to training. Once the signal is converted into an appropriate voltage range, voltage pulses are applied to a memristor, and its conductance after each timeframe serves as a reservoir weight. This approach presumes that each data point relies on the preceding 50 weights. The 20 memristors used in this experiment exhibit slight property variations inspired by the unavoidable device-to-device discrepancies found in real-world implementations, which enhance the system's nonlinearity.

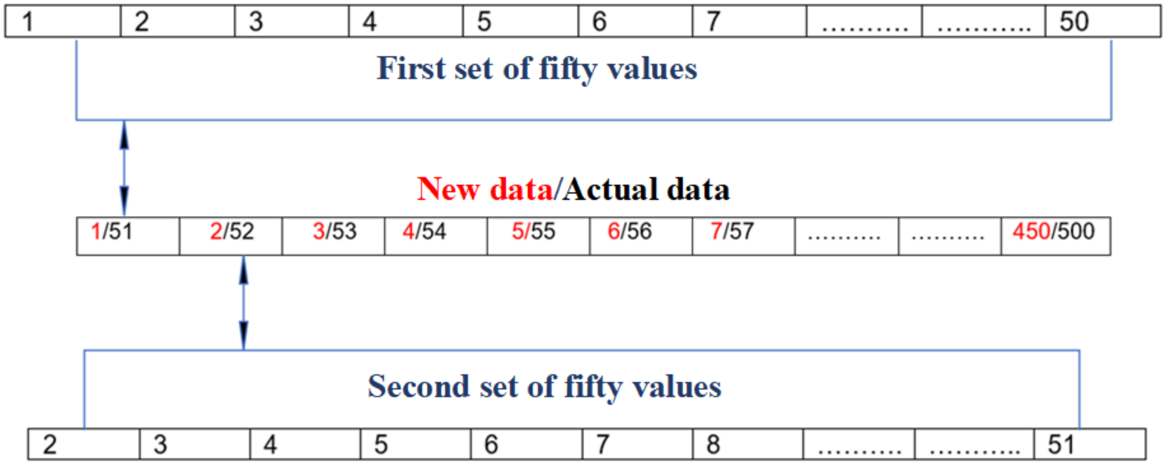


Figure 6.4: Schematic of data process for autonomous Mackey Glass time series problem prediction.

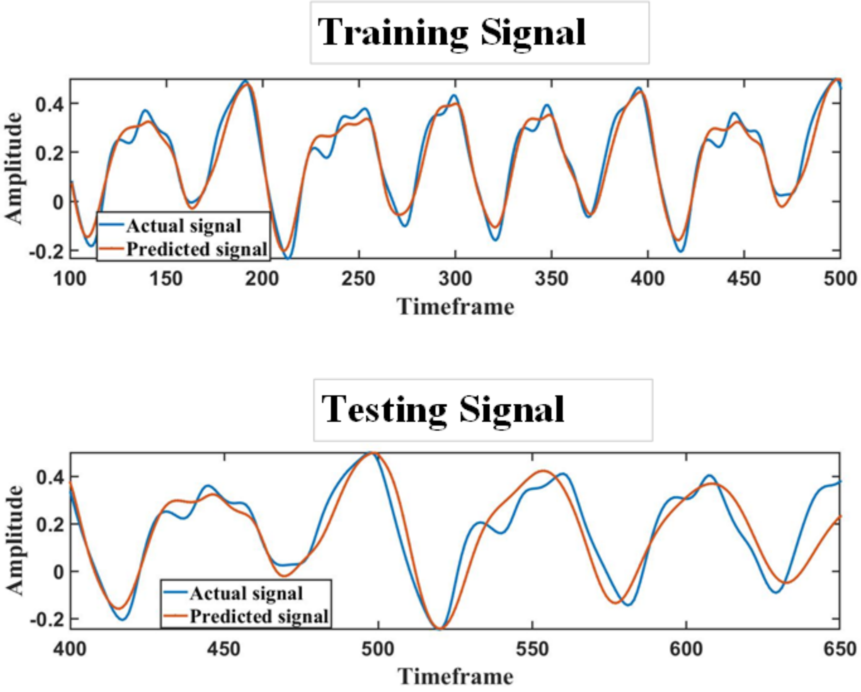


Figure 6.5: Actual signal vs. predicted signal from memristive RC system for Mackey Glass time series autonomous prediction both the training and testing sets.

In effect, 1000 weights are available for predicting each successive data point, accomplished by training the readout layer using linear regression. Testing occurs from the 501^{th} to the 650^{th} time step. The model autonomously predicts data for future time steps based on training weights.

In the Mackey-Glass time series forecasting, the signal is transformed into a voltage ranging from 110 mV to 130 mV. The time frame is 15 ms with an 80% duty cycle. This simulation employs 20 distinct memristors. Fig. 6.5 reveals that the actual and predicted signals nearly align perfectly for the training set, encompassing the first 500 time frames. Fig. 6.5 also displays the testing outcomes for autonomous prediction between the 501^{th} and 650^{th} time frames. Our model can accurately predict the signal for up to 30-40 steps beyond the 500-time steps. However, due to the exponential growth of minor errors, the autonomous prediction diverges from the ground truth.

6.3 EEG Signal Classification

Electroencephalogram (EEG) signals are a non-invasive method of measuring the electrical activity generated by the brain. These signals are detected by placing small electrodes on the scalp that capture the brain's electrical oscillations. EEG signals have been widely used in clinical and research settings to understand the complex dynamics of brain function, diagnose neurological disorders, and study cognitive processes. The advantages of using EEG signals include high temporal resolution, which allows researchers to capture fast-changing brain processes. Additionally, EEG equipment is relatively affordable and portable, making it a widely accessible tool for brain research. However, the spatial resolution of EEG is limited due to the nature of scalp-recorded signals, which can be affected by the skull and scalp tissues. EEG signals have various applications, including diagnosing epilepsy, sleep disorders, and coma. They are also utilized in brain-computer interfaces (BCIs) to enable communication and control for individuals with severe motor disabilities. Furthermore, EEG signals have been employed to study neural correlates of cognition, memory, and attention, providing valuable insights into the workings of the human brain. EEG signal classification based on a machine learning system has been described as a literature review in the following;

Lotte et al. [88] paper overviews classification algorithms used in EEG-based brain-computer interfaces. The authors discuss the advantages and drawbacks of various techniques, including linear discriminant analysis, support vector machines, and neural networks. They also provide guidelines for selecting and evaluating classifiers for specific BCI applications. Bashashati et al. [12] paper focuses on the signal processing algorithms used in EEG-based brain-computer interfaces. The authors discuss various preprocessing, feature extraction, and classification techniques, highlighting their suitability for different BCI applications. They also provide insights into future research directions in this field. Murugappan et al. [104] propose a method for classifying human emotions using EEG signals and discrete wavelet transform. They employ machine learning algorithms to classify the extracted features, including k-Nearest Neighbors, linear discriminant analysis, and support vector machines. The results demonstrate the potential of using machine learning techniques for emotion recognition based on EEG signals. The survey paper [3] provides a comprehensive overview of seizure detection and prediction algorithms based on EEG signals. The authors discuss various techniques, including time-domain, frequency-domain, and time-frequency-domain methods. They also review different machine learning algorithms employed in seizure classification, such as support vector machines, artificial neural networks, and fuzzy logic systems. Cecotti et al. [18] propose using convolutional neural networks (CNNs) to detect P300 components in EEG signals, commonly used in brain-computer interfaces. The results show that CNNs can achieve high classification accuracy and outperform traditional machine-learning techniques in this application. Tzallas et al. [163] present a novel approach to epileptic seizure detection using time-frequency analysis of EEG signals. They employ wavelet transform and machine learning techniques, such as artificial neural networks and support vector machines, to classify seizure and non-seizure EEG epochs. The results demonstrate the effectiveness of the proposed method for seizure detection. Lawhern

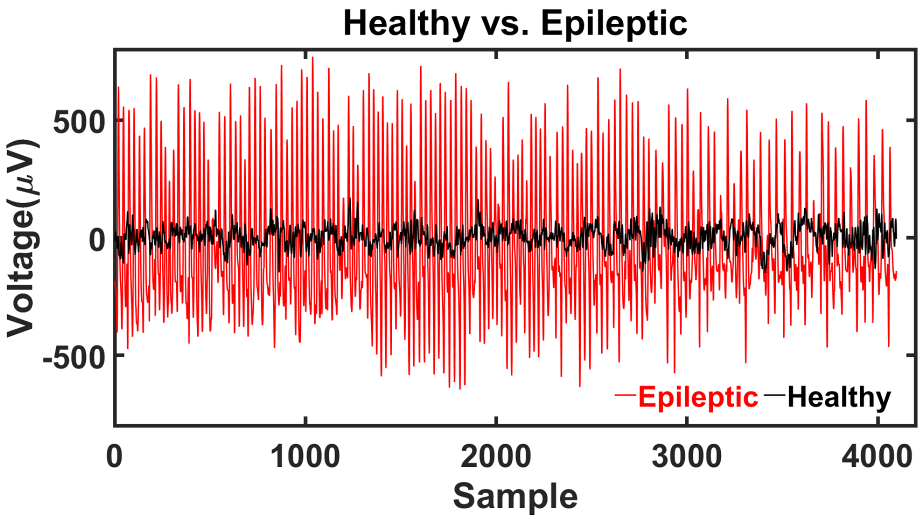


Figure 6.6: Example time series plots of two EEG signals: Healthy and Epileptic.

et al. [76] introduce EEGNet, a compact convolutional neural network (CNN) architecture designed specifically for EEG-based brain-computer interfaces. The authors demonstrate that EEGNet achieves competitive performance with other state-of-the-art classifiers while maintaining a smaller model size and lower computational complexity. The results highlight the potential of using tailored deep-learning architectures for EEG signal classification. Huang et al. [59] present the extreme learning machine (ELM) algorithm, a type of single-layer feedforward neural network with a fast learning speed. The authors demonstrate the effectiveness of ELM in various applications, including EEG signal classification. The ELM algorithm has since been widely adopted in EEG data analysis for its efficiency and accuracy.

The work aims to demonstrate a memristor-based RC system to classify epileptic and healthy EEG signals. The proposed method is general and can be implemented using any volatile memristor with fading memory and nonlinearity. I have shown reservoir implementation using only biomolecular memristor and later compared it with other works.

EEG Dataset: EEG serves to record brain activation signals over time. Valuable information about the brain can be gleaned from these signals, as noted by [132]. In this study, we utilized a dataset from the University of Bonn, Germany [6]. The dataset comprises five classes of EEG signals: F, N, S, Z, and O, each containing 100 EEG signals lasting 23.6

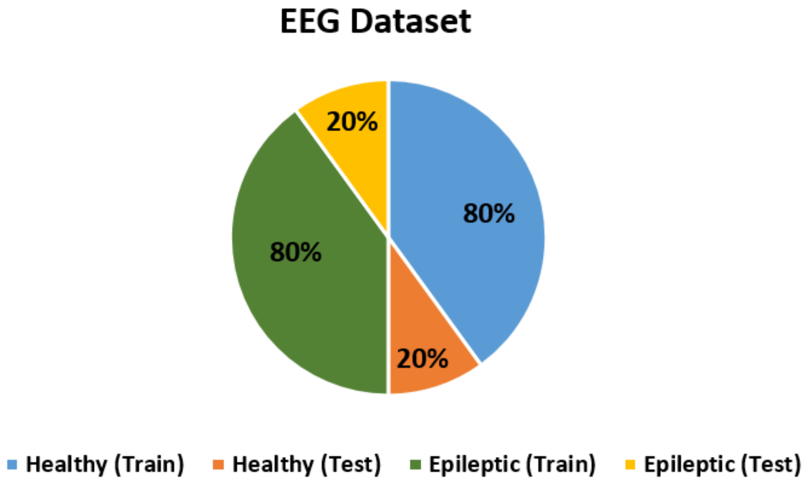


Figure 6.7: Proportion of data used in our analysis.

seconds and sampled at a frequency of 173.67 Hz. Seizure activity is captured in class S EEG signals, while healthy patient data is represented in class Z EEG signals [110]. My work focuses solely on classifying class Z and S EEG signals. Figure 6.6 illustrates examples of healthy (class Z) and epileptic (class S) EEG signals, and 6.7 demonstrates the distribution of data used for training and testing in our analysis.

In a memristor-based reservoir computing (RC) system, there are three distinct layers: the input layer, the reservoir layer, and the output layer,[57] [39]. The input layer introduces the input signals to the encoding layer, which transforms these signals into voltage amplitude pulses. These pulses are then transmitted to the memristive reservoir. The reservoir's dynamics are altered based on the input of pulse trains, while the output layer houses a readout function where linear or logistic regressions are executed [57]. Unlike traditional recurrent networks, only the weights from the readout layer need to be trained in the RC system, significantly reducing training costs.

In order to enhance the performance of the RC system, we have incorporated an additional layer (the feature modification layer) following the reservoir layer, which reduces feature dimensions. The benefit of decreasing feature size lies in memory conservation and increased system performance accuracy.

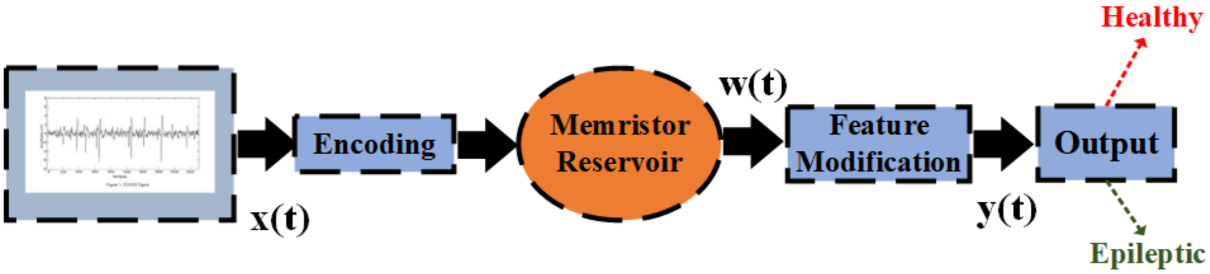


Figure 6.8: Proposed RC framework to reduce feature size in the network.

Figure 6.8 shows the proposed RC framework. At first, the raw data is preprocessed, taking absolute value and then clipping at a maximum of 500 μV. Then the data is linearly encoded into a suitable voltage range depending on the particular memristor's functionality. These amplitude-encoded values are the on-voltages of a pulse train sent to the memristive reservoir. Memristors use these voltage inputs to change their conductance non-linearly; also, the conductance depends on the present and the past inputs. Thereby, memristor conductance can be treated reservoir state. The distinct characteristic of the volatile memristive element compared to other passive elements is that its resistance changes nonlinearly with time depending on the sequence of past applied voltages [20] [176]. We have used both solid-state memristors [193] and biomolecular memristors [50] separately to build the reservoir model for simulation. The solid-state memristors operate at 2.5 V, whereas the biomolecular memristors operate between 10 to 200 mV.

The feature modification layers take weights $w(t)$ from the reservoir and produce a reduced set of features for the readout layer. Here, we have shown two techniques. The first is the virtual node technique, where instead of taking the conductance at each input time step, we take the conductance at the n^{th} time step. The second technique is a new one we propose in this work called 'the integration method.' Instead of taking just the n^{th} step conductance, we sum all the conductances between 1 and n^{th} step and treat that as a single feature to be used in the readout layer. In the feature modification layer, we summed up the first 60 weights to get the first feature, then calculated the sum of the following 60 weights to get the second feature, and so on. Thereby, we have reduced the feature size, which reduces

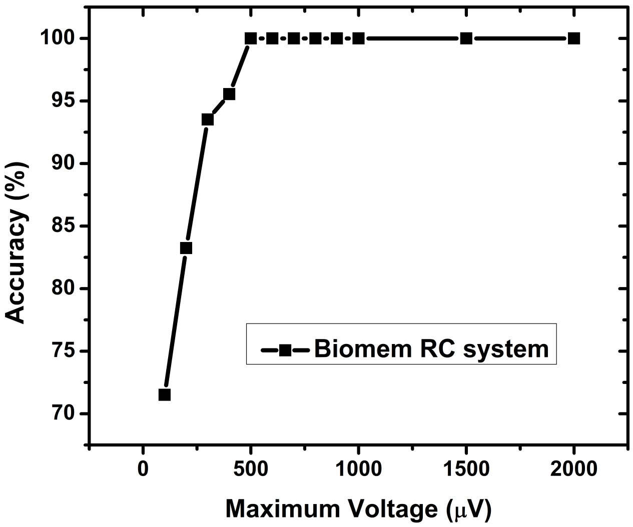


Figure 6.9: Maximum voltage clipping of raw EEG signals.

the readout layer size and consequently optimizes network complexity. In addition, each specific time point does not have any intrinsic meaning in this EEG signal. Hence, treating each time step response as a feature can be misleading when it comes to classifying signals with a phased difference. Summing over a significant number of time steps mitigates this issue.

Logistic regression is a supervised machine learning algorithm to classify the input signals [134]. In the proposed RC network, the logistic regression is performed after the feature modification to train the network and use the trained weights in the testing phase for classification.

Figure 6.9 shows the reason behind choosing the maximum clipping voltage level of 500 μV. We have achieved maximum accuracy with up to 500 μV raw data. We have used two techniques for both solid-state and biomolecular memristor-based RC systems in this work. Figure 6.10 shows another reason for choosing 60-time steps in our analysis. The virtual node is the point where we collect the data, and in this analysis, we have chosen 60-time steps. The selection of the virtual nodes also impacts the accuracy. The number of virtual nodes increases the feature size more than the dataset size, which creates poor accuracy. On the other hand, the large number of virtual nodes drastically reduces feature

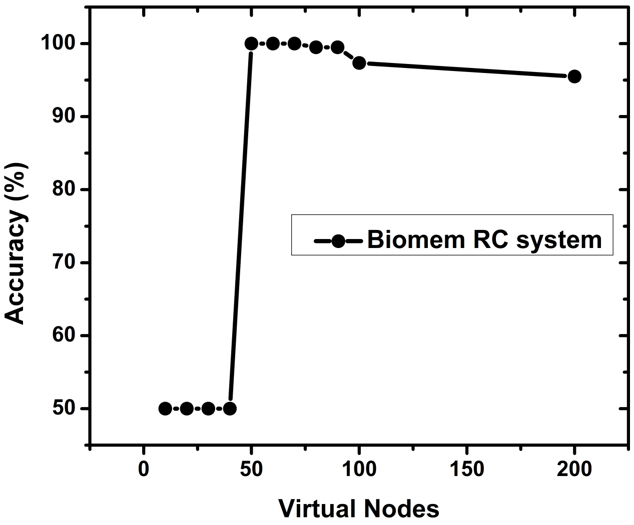


Figure 6.10: Selection of Virtual Nodes for EEG classification.

size and accuracy. In addition, in the integration method, the network sustains better properties than traditional memristive RC networks as the 60 weights are summed up to create one feature. In contrast, the traditional memristive RC network only gets the 60^{th} position weights having less past information. A linear network consisting of resistors has been used in a separate work to understand the significance of the memory property of the memristor to solve the EEG classification task.

A list of EEG data classification results and our work has been reported in table 6.2. From the table, it is clear that using the feature modification layer, memristor-based RC system can classify EEG signals with 100% accuracy and reduce the feature size of the readout layer. In addition, I have conducted an experiment for memcapacitor-based RC system to classify EEG signal classification and get 100% test accuracy in simulation and experiment.

6.4 Spoken digit classification

In this section, we conduct a speech recognition experiment utilizing a memristor-based reservoir computing (RC) system. The dataset employed is the NIST TI45 [39],

Table 6.2: Comparison of performance of the proposed method with other methods in healthy vs. epileptic EEG signal classification problem.

Authors	Description	Accuracy,%
Nigam et al.[109]	Non-linear preprocessing filter, artificial neural network	97.2
Polat et al.[129]	Fast Fourier transform, decision tree	98.72
Subasi [159]	Discrete wavelet transform, mixture of expert model	95
Guo et al. [45]	Discrete wavelet transform -relative wavelet energy	95.2
Oweis et al.[114]	Hilbert–Huang transform, weighted frequencies	94
Wang et al. [169]	Wavelet transform, Shannon entropy, k-NN	99-100
Kai et al. [35]	Hilbert–Huang transform, SVM	99.12
Bajaj et al. [9]	Time–frequency image-based features, least squares SVM	99.5
Guo et al. [44]	Preprocessed by Genetic Algorithm, Feature: curve length, standard deviation, KNN classifier	99.20
Siuly et al. [84]	Preprocessed by Clustering, Feature: 9 temporal features, LS-SVM classifier	99.90
Samiee et al. [142]	Preprocessed by Rational DSTFT, Feature: 5 time-frequency features, MLP classifier	99.80
Rincon et al. [96]	Preprocessed by wavelet transform, Feature: bag of words, SVM classifier	99.85
Reynolds et al. [132]	Perceptron model	54.05
Reynolds et al. [132]	Multi-Layer Perceptron model	48.25
Reynolds et al. [132]	Convolutional Deep Learning network model	99.00
Reynolds et al. [132]	Long Short-Term Memory model	45.00
Reynolds et al. [132]	Evolutionary Optimization of Neuromorphic System	99.00
Reynolds et al. [132]	Reservoir Computing model	98.00
This Work	**Linear Network**	**56.25**
This Work	**Solid state memristor-based RC network with virtual node**	**99.75**
This Work	**Solid state memristor-based RC network with integration method**	**100**
This Work	**Biomolecular memristor-based RC network with virtual node**	**93.12**
This Work	**Biomolecular memristor-based RC network with integration method**	**100**

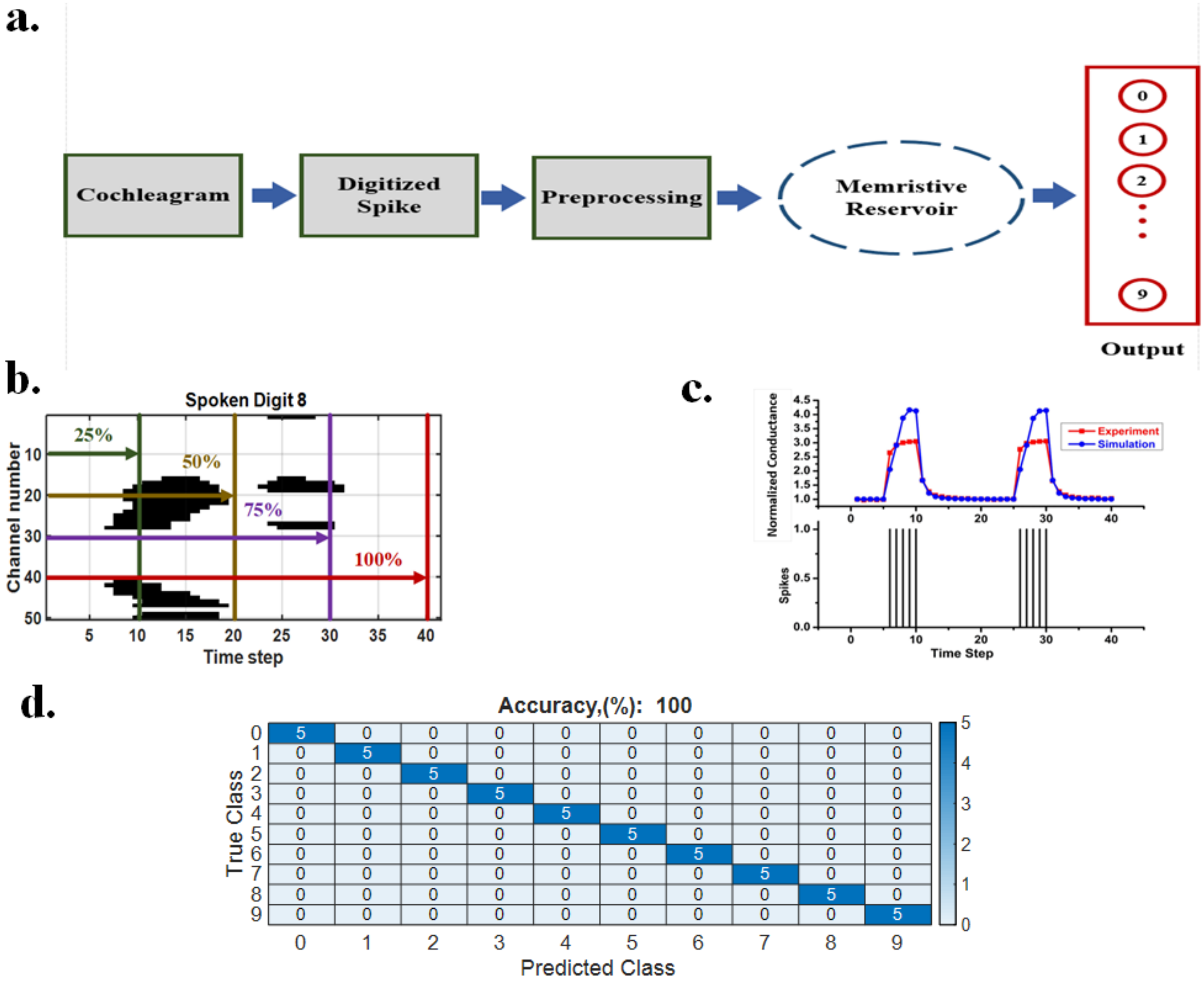


Figure 6.11: Spoken Digit Classification. a) Process flow for Spoken Digit Classification. b) Digitized spike for spoken digit ”8” explored for 25%,50%,75%,100%. c) memristor conductance change for the digitized spike. d) simulation results for 40-time steps of the cochleagram.

consisting of isolated spoken digit waveforms (0-9 in English). Initially, the dataset undergoes preprocessing via Lyon's passive ear model, converting the sound waveforms into 50-dimensional vectors (representing frequency channels) with a maximum of 40-time steps. The dataset we used, NIST TI46, comprises binary cochleograms of isolated spoken digit waveforms, 0-9 in spoken English. The dataset was provided to us as a courtesy by the authors of Moon*etal.*[102]. Each data point in the cochleagram signifies the firing probability of a hair cell sensitive to a specific frequency (channel) at a particular time.

Figure 6.11a presents an illustration of the process flow for tackling the EEG classification issue. A dataset consisting of 450 training samples and 50 test samples is employed to do this. Initially, the cochleagram data is transformed into a digitized spike representation, where white areas are designated as 0, while black areas containing voice data are denoted as 1. Subsequently, the digitized data is converted into a voltage pulse train, which is then transmitted to the memristive reservoir.

The data is processed in a high-dimensional space within the memristive reservoir before being sent to the output layer for training purposes. A straightforward logistic regression is conducted to determine the weights during the training phase, which will later be used in the testing phase. Figure 6.11b demonstrates the observations made at various time steps to ascertain if the device can detect the digit before processing the entire data segment.

Figure 6.11c depicts the relationship between the memristive conductance changes and the spike data. It becomes evident that the conductance increases when spike data (or pulse train) is present and decreases for non-spike positions. Lastly, Figure 6.11d presents the results obtained for a 40-time step dataset, where a 100% recognition rate accuracy was achieved. This extended description of the EEG classification process offers a more comprehensive understanding for dissertation purposes.

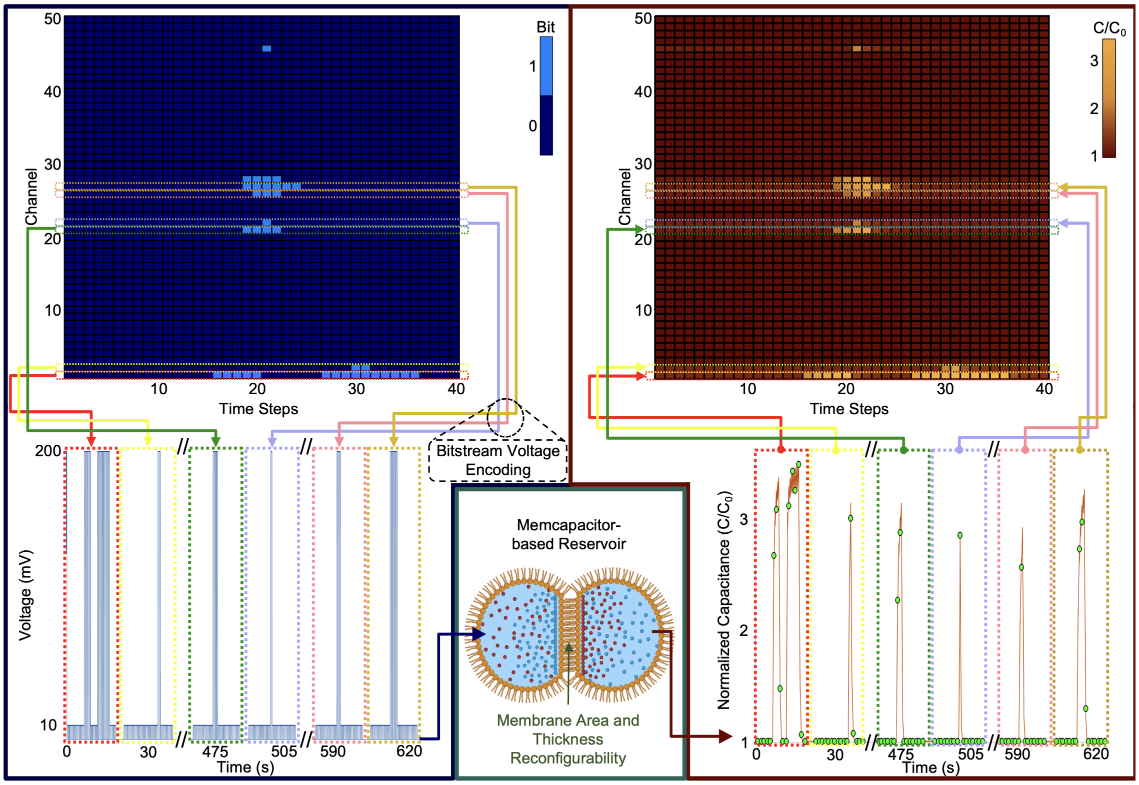

Bit
1
0
50
40
30
20
10
Channel
10
20
30
40
Time Steps
C/C$_0$
3
2
1
Bitstream Voltage Encoding
200
10
Voltage (mV)
0
30
475
505
590
620
Time (s)
Memcapacitor-based Reservoir
Membrane Area and Thickness Reconfigurability
Normalized Capacitance (C/C$_0$)

Figure 6.12: The encoding process flow of the Spoken Digit classification problem. The binary 2-D cochleogram (top-left corner) represents neural spike trains in different human cochlear channels. Each input was converted into voltage pulses (bottom-left corner), where 10 mV and 200 mV correspond to a '0' (resting neuron) bit and a '1' (firing neuron) bit, respectively. The input voltage train was fed to the memcapacitor-based reservoir, where the normalized capacitance response is recorded (bottom-right corner). The dynamic normalized capacitance was then mapped to a 2-D matrix (top-right corner) and, for every channel or row, one virtual node is selected for every 5 timesteps as depicted by the green circles on the capacitance plot (bottom-right corner)

The figure 6.12 under discussion depicts the encoding process flow involved in the Spoken Digit classification problem, a task aimed at correctly identifying and categorizing spoken numbers.

The process begins with a binary 2-D cochleogram (top-left corner). This graphical representation captures the auditory information as 'neural spike trains' along different human cochlear channels, essentially representing how the human ear processes the spoken digit. Each point in the cochleogram signifies a specific firing neuron in a particular channel at a certain point in time. This encapsulates the complexity of human auditory data in a simple binary format.

This cochleogram data is then converted into voltage pulses (bottom-left corner). In this system, the binary '0' and '1' data from the cochleogram corresponds to 10 mV and 200 mV, respectively. The '0' bit signifies a resting neuron, whereas a '1' bit symbolizes a firing neuron. This transformation provides an electrical representation of the auditory data, essentially serving as an input voltage train.

The next step involves feeding this input voltage train to a memcapacitor-based reservoir. A memcapacitor is a type of memory device that has the ability to remember its history through the voltage and charge that has passed through it. The reservoir acts as a high-dimensional space where the time-series data (input voltage train) is mixed and transformed nonlinearly, allowing the system to detect and exploit patterns in the input.

The resultant normalized capacitance response from the memcapacitor is recorded (bottom-right corner). Capacitance, in this context, refers to the capacity of the memcapacitor to store electrical charge. This recorded response displays the unique and dynamic capacitance characteristics of the auditory data and serves as a distinctive fingerprint for each spoken digit.

Finally, the dynamic normalized capacitance is mapped to a 2-D matrix (top-right corner). For each channel (or row), a virtual node is selected for every 5 timesteps as depicted by the green circles on the capacitance plot (bottom-right corner). This step allows

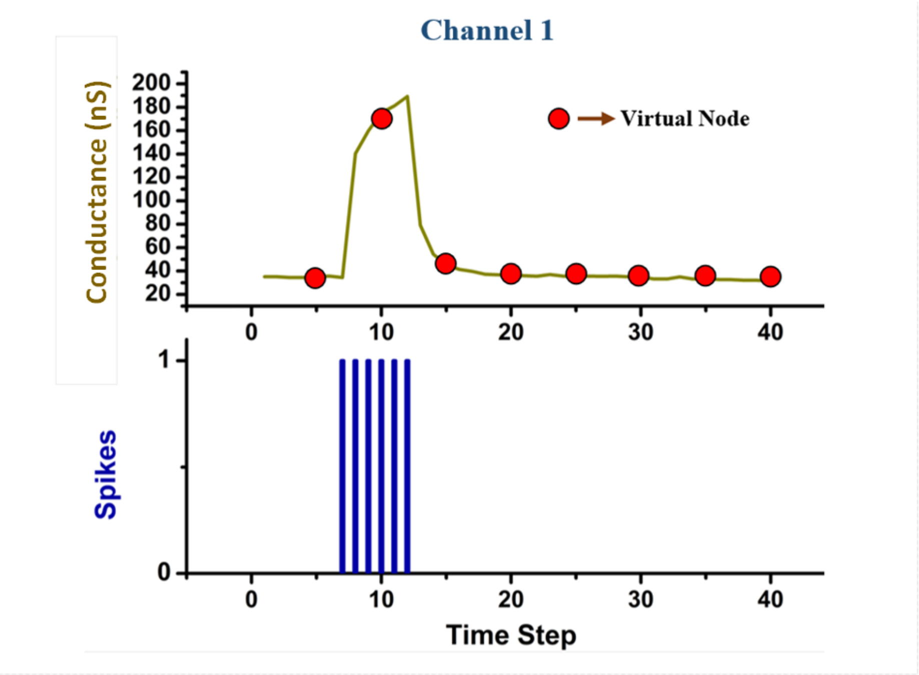


Figure 6.13: Temporal response of memcapacitor to the spike trains in channel 1 for utterance '0'.

Table 6.3: Comparison of Spoken digit classification work

Devices/Network	**Operating voltage**	**Recognition rate, %**
Biomolecular Memristor	170 mV	99.75
Biomolecular Memcapacitor	150 mV	99.6
Solid state Memristor [102]	1.8	99.2

the system to condense and interpret the data in a simpler form, enabling it to effectively classify the spoken digits. Selecting a virtual node every 5 timesteps, it ensures that the system gets a diverse and representative sampling of the unique auditory data for each digit, thus boosting the accuracy and reliability of the classification process.

In this case, the device's memcapacitance is sampled at intervals of 5-time steps, following the pattern n, 2n, 3n, 4n, ... 8n, with n equating to 5. The red dots in Figure 6.13 signify virtual nodes. These virtual nodes, as referenced in [102], represent the states of the memcapacitor where memcapacitance values are stored. For each channel, eight virtual nodes have been utilized. Ultimately, the states of these virtual nodes are relayed to the readout layer. A list of comparison work for spoken digit classification for 40-time steps

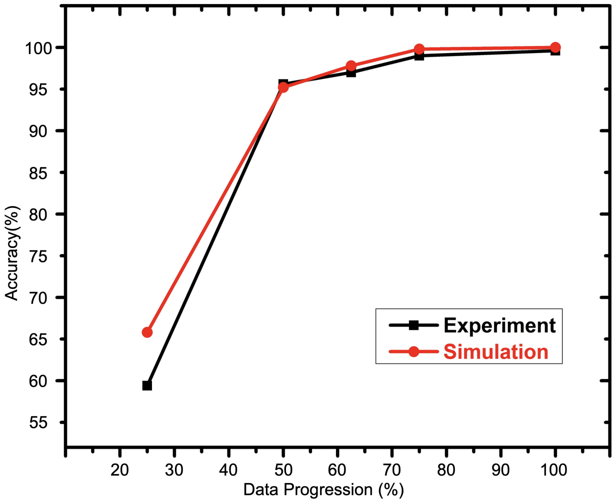


Figure 6.14: Recognition rate of spoken digit classification problem using memcapacitor based RC system.

has been presented in table 6.3. The outcomes for spoken digit classification are displayed in both simulation and experimental settings for biomolecular memcapacitor-based RC systems in figure 6.14. The performance is evaluated at various input completion percentages, specifically for 25%, 50%, 62.5%, 75%, and 100% (i.e., $10, 20, 25, 30, 40$ time steps) of the test dataset.

7 CONCLUSION and FUTURE WORK

Several promising avenues for future research have been identified as a result of this dissertation. Firstly, the development of novel bio-mem devices with enhanced performance and functionality should be explored, focusing on the integration of various biomolecules and optimization of their switching mechanisms. This would contribute to the creation of more sophisticated and reliable bio-mem devices, which could enable further breakthroughs in the field.

Secondly, the scalability of bio-mem devices in larger RC networks should be investigated, in order to determine their suitability for real-world applications that demand even greater computational power. This would involve the development of novel architectures and algorithms to efficiently incorporate bio-mem devices into large-scale systems.

Additionally, further exploration of alternative learning algorithms for RC systems should be conducted to improve performance and enable more complex tasks to be solved. It would be worthwhile to investigate the use of unsupervised and reinforcement learning algorithms in conjunction with bio-mem devices, as well as the development of novel hybrid approaches that combine the strengths of various learning paradigms.

Finally, as the field of neuromorphic computing continues to evolve, it would be beneficial to explore the integration of bio-mem devices with other emerging technologies, such as optical computing and quantum computing. This interdisciplinary approach could lead to the development of unprecedented computing capabilities, pushing the boundaries of what is currently achievable in the field of computing.

In conclusion, this dissertation has demonstrated the potential of bio-inspired memory devices as a promising alternative to traditional solid-state electronics for tackling the limitations imposed by the slowing down of Moore's law, the breakdown of Dennard Scaling, and the von Neumann bottleneck. By leveraging the unique properties of biomolecular devices, such as non-linear current-voltage characteristics and inherent memory, we have successfully integrated them into Reservoir Computing (RC) systems to solve classification and temporal problems.

Our results show that bio-mem devices not only offer a more energy-efficient and biologically plausible computing solution but also exhibit remarkable performance in various tasks, including speech recognition, time-series prediction, and image classification. This work paves the way for the development of more efficient and powerful computing systems that can handle complex tasks more effectively while also reducing power consumption and mitigating the limitations faced by traditional computing paradigms.

BIBLIOGRAPHY

[1] J Ajayan, D Nirmal, Binola K Jebalin, and S Sreejith, *Advances in neuromorphic devices for the hardware implementation of neuromorphic computing systems for future artificial intelligence applications: A critical review*, Microelectronics Journal (2022), 105634.

[2] Fabien Alibart, Elham Zamanidoost, and Dmitri B Strukov, *Pattern classification by memristive crossbar circuits using ex situ and in situ training*, Nature communications **4** (2013), no. 1, 2072.

[3] Turkey N Alotaiby, Saleh A Alshebeili, Tariq Alshawi, Ishtiaq Ahmad, and Fathi E Abd El-Samie, *Eeg seizure detection and prediction algorithms: a survey*, EURASIP Journal on Advances in Signal Processing **2014** (2014), 1–21.

[4] Gonzalo Alvarez and Shujun Li, *Some basic cryptographic requirements for chaos-based cryptosystems*, International journal of bifurcation and chaos **16** (2006), no. 08, 2129–2151.

[5] Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan, *Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions*, Journal of big Data **8** (2021), 1–74.

[6] Ralph G Andrzejak, Klaus Lehnertz, Florian Mormann, Christoph Rieke, Peter David, and Christian E Elger, *Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state*, Physical Review E **64** (2001), no. 6, 061907.

[7] Lennert Appeltant, Miguel Cornelles Soriano, Guy Van der Sande, Jan Danckaert, Serge Massar, Joni Dambre, Benjamin Schrauwen, Claudio R Mirasso, and Ingo Fischer, *Information processing using a single dynamical node as complex system*, Nature communications **2** (2011), no. 1, 1–6.

[8] Lennert Appeltant, Guy Van der Sande, Jan Danckaert, and Ingo Fischer, *Constructing optimized binary masks for reservoir computing with delay systems*, Scientific reports **4** (2014), no. 1, 3629.

[9] Varun Bajaj, Khushnandan Rai, Anil Kumar, and Dheeraj Sharma, *Time-frequency image based features for classification of epileptic seizures from eeg signals*, Biomedical Physics & Engineering Express **3** (2017), no. 1, 015012.

[10] MS Baptista, *Cryptography with chaos*, Physics letters A **240** (1998), no. 1-2, 50–54.

[11] Colin Basham, Megan Pitz, Joseph Najem, Stephen Sarles, and Md Sakib Hasan, *Memcapacitive devices in neuromorphic circuits via polymeric biomimetic membranes*, Smart Materials, Adaptive Structures and Intelligent Systems, vol. 59131, American Society of Mechanical Engineers, 2019, p. V001T06A009.

[12] Ali Bashashati, Mehrdad Fatourechi, Rabab K Ward, and Gary E Birch, *A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals*, Journal of Neural engineering **4** (2007), no. 2, R32.

[13] Akram Belazi, Ahmed A Abd El-Latif, and Safya Belghith, *A novel image encryption scheme based on substitution-permutation network and chaos*, Signal Processing **128** (2016), 155–170.

[14] Dalibor Biolek, Zdeněk Kolka, Viera Biolková, Zdeněk Biolek, Milka Potrebić, and Dejan Tošić, *Modeling and simulation of large memristive networks*, International Journal of Circuit Theory and Applications **46** (2018), no. 1, 50–65.

[15] Samuel D Brown, Gangotree Chakma, Md Musabbir Adnan, Md Sakib Hasan, and Garrett S Rose, *Stochasticity in neuromorphic computing: Evaluating randomness for improved performance*, 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), IEEE, 2019, pp. 454–457.

[16] Geoffrey W Burr, Robert M Shelby, Abu Sebastian, Sangbum Kim, Seyoung Kim, Severin Sidler, Kumar Virwani, Masatoshi Ishii, Pritish Narayanan, Alessandro Fumarola, et al., *Neuromorphic computing using non-volatile memory*, Advances in Physics: X **2** (2017), no. 1, 89–124.

[17] Martin Casdagli, Stephen Eubank, J Doyne Farmer, and John Gibson, *State space reconstruction in the presence of noise*, Physica D: Nonlinear Phenomena **51** (1991), no. 1-3, 52–98.

[18] Hubert Cecotti and Axel Graser, *Convolutional neural networks for p300 detection with application to brain-computer interfaces*, IEEE transactions on pattern analysis and machine intelligence **33** (2010), no. 3, 433–445.

[19] Sanghyeon Choi, Jehyeon Yang, and Gunuk Wang, *Emerging memristive artificial synapses and neurons for energy-efficient neuromorphic computing*, Advanced Materials **32** (2020), no. 51, 2004659.

[20] Leon Chua, *Memristor-the missing circuit element*, IEEE Transactions on circuit theory **18** (1971), no. 5, 507–519.

[21] ———, *Everything you wish to know about memristors but are afraid to ask*, Handbook of Memristor Networks (2019), 89–157.

[22] Dan C Cireşan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber, *Mitosis detection in breast cancer histology images with deep neural networks*, Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013: 16th International Conference, Nagoya, Japan, September 22-26, 2013, Proceedings, Part II 16, Springer, 2013, pp. 411–418.

[23] Charles Collier, Joseph Najem, Stan Williams, Graham Taylor, Catherine Schuman, Alex Belianinov, Benjamin Doughty, Ryan Weiss, Md Sakib Hasan, Garrett Rose,

et al., *Memory and learning in biomolecular soft materials*, Bulletin of the American Physical Society **65** (2020).

[24] Corinna Cortes and Vladimir Vapnik, *Support-vector networks*, Machine learning **20** (1995), 273–297.

[25] Hongyan Cui, Xiang Liu, and Lixiang Li, *The architecture of dynamic reservoir in the echo state network*, Chaos: An Interdisciplinary Journal of Nonlinear Science **22** (2012), no. 3, 033127.

[26] Lei Deng, Guoqi Li, Ning Deng, Dong Wang, Ziyang Zhang, Wei He, Huanglong Li, Jing Pei, and Luping Shi, *Complex learning in bio-plausible memristive networks*, Scientific reports **5** (2015), no. 1, 1–10.

[27] Robert L Devaney, *An introduction to chaotic dynamical systems*, Bull. Am. Math. Soc **16** (1987), 313–315.

[28] Massimiliano Di Ventra, Yuriy V Pershin, and Leon O Chua, *Circuit elements with memory: memristors, memcapacitors, and meminductors*, Proceedings of the IEEE **97** (2009), no. 10, 1717–1724.

[29] Chao Du, Fuxi Cai, Mohammed A Zidan, Wen Ma, Seung Hwan Lee, and Wei D Lu, *Reservoir computing using dynamic memristors for temporal information processing*, Nature communications **8** (2017), no. 1, 2204.

[30] Chao Du, Wen Ma, Ting Chang, Patrick Sheridan, and Wei D Lu, *Biorealistic implementation of synaptic functions with oxide memristors through internal ionic dynamics*, Advanced Functional Materials **25** (2015), no. 27, 4290–4299.

[31] Forhan Bin Emdad, Shuyuan Mary Ho, Benhur Ravuri, and Shezin Hussain, *Towards a unified utilitarian ethics framework for healthcare artificial intelligence*, (2023).

[32] Forhan Bin Emdad, Shubo Tian, Esha Nandy, Karim Hanna, and Zhe He, *Towards interpretable multimodal predictive models for early mortality prediction of hemorrhagic stroke patients*, AMIA Summits on Translational Science Proceedings **2023** (2023), 128.

[33] EA Evans and S Simon, *Mechanics of electrocompression of lipid bilayer membranes.*, Biophysical Journal **15** (1975), no. 8, 850.

[34] Dario Floreano, Peter Dürr, and Claudio Mattiussi, *Neuroevolution: from architectures to learning*, Evolutionary intelligence **1** (2008), 47–62.

[35] Kai Fu, Jianfeng Qu, Yi Chai, and Yong Dong, *Classification of seizure based on the time-frequency image of eeg signals using hht and svm*, Biomedical Signal Processing and Control **13** (2014), 15–22.

[36] Steve Furber, *Large-scale neuromorphic computing systems*, Journal of neural engineering **13** (2016), no. 5, 051001.

[37] Claudio Gallicchio, Alessio Micheli, and Luca Pedrelli, *Deep reservoir computing: A critical experimental analysis*, Neurocomputing **268** (2017), 87–99.

[38] Crescenzio Gallo, *Artificial neural networks tutorial*, Encyclopedia of Information Science and Technology, Third Edition, IGI Global, 2015, pp. 6369–6378.

[39] Daniel J Gauthier, Erik Bollt, Aaron Griffith, and Wendson AS Barbosa, *Next generation reservoir computing*, Nature communications **12** (2021), no. 1, 1–8.

[40] Samanwoy Ghosh-Dastidar and Hojjat Adeli, *Spiking neural networks*, International journal of neural systems **19** (2009), no. 04, 295–308.

[41] James Gleick and M Berry, *Chaos-making a new science*, Nature **330** (1987), 293.

[42] Li-Hua Gong, Hui-Xin Luo, Rou-Qing Wu, and Nan-Run Zhou, *New 4d chaotic system with hidden attractors and self-excited attractors and its application in image encryption based on rng*, Physica A: Statistical Mechanics and its Applications **591** (2022), 126793.

[43] Ian J Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet, *Multi-digit number recognition from street view imagery using deep convolutional neural networks*, arXiv preprint arXiv:1312.6082 (2013).

[44] Ling Guo, Daniel Rivero, Julián Dorado, Cristian R Munteanu, and Alejandro Pazos, *Automatic feature extraction using genetic programming: An application to epileptic eeg classification*, Expert Systems with Applications **38** (2011), no. 8, 10425–10436.

[45] Ling Guo, Daniel Rivero, Jose A Seoane, and Alejandro Pazos, *Classification of eeg signals using relative wavelet energy and artificial neural networks*, Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009, pp. 177–184.

[46] Md Sakib Hasan, Md Badruddoja Majumder, Aysha S Shanta, Mesbah Uddin, and Garrett S Rose, *A chaos-based complex micro-instruction set for mitigating instruction reverse engineering*, Journal of Hardware and Systems Security **4** (2020), 69–85.

[47] Md Sakib Hasan, Joseph S Najem, Ryan Weiss, Catherine D Schuman, Alex Belianinov, C Patrick Collier, Stephen A Sarles, and Garrett S Rose, *Response of a memristive biomembrane and demonstration of potential use in online learning*, 2018 IEEE 13th Nanotechnology Materials and Devices Conference (NMDC), IEEE, 2018, pp. 1–4.

[48] Md Sakib Hasan, Partha Sarathi Paul, Maisha Sadia, and Md Razuan Hossain, *Design of a weighted average chaotic system for robust chaotic operation*, 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), IEEE, 2021, pp. 954–957.

[49] ———, *Integrated circuit design of an improved discrete chaotic map by averaging multiple seed maps*, SoutheastCon 2021, IEEE, 2021, pp. 1–6.

[50] Md Sakib Hasan, Catherine D Schuman, Joseph S Najem, Ryan Weiss, Nicholas D Skuda, Alex Belianinov, C Patrick Collier, Stephen A Sarles, and Garrett S Rose, *Biomimetic, soft-material synapse for neuromorphic computing: from device to network*, 2018 IEEE 13th Dallas Circuits and Systems Conference (DCAS), IEEE, 2018, pp. 1–6.

[51] Jennifer Hasler and Bo Marr, *Finding a roadmap to achieve large neuromorphic hardware systems*, Frontiers in neuroscience **7** (2013), 118.

[52] Boris Hasselblatt and Anatole Katok, *A first course in dynamics: with a panorama of recent developments*, Cambridge University Press, 2003.

[53] DJ Herzfeld and SA Beardsley, *Improved multi-unit decoding at the brain–machine interface using population temporal linear filtering*, Journal of Neural Engineering **7** (2010), no. 4, 046012.

[54] Geoffrey E Hinton and Ruslan R Salakhutdinov, *Reducing the dimensionality of data with neural networks*, science **313** (2006), no. 5786, 504–507.

[55] Mark Horowitz, *1.1 computing's energy problem (and what we can do about it)*, 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), IEEE, 2014, pp. 10–14.

[56] Md Razuan Hossain, Ahmed Salah Mohamed, Nicholas Xavier Armendarez, Joseph S Najem, and Md Sakib Hasan, *Energy-efficient memcapacitive physical reservoir computing system for temporal data processing*, arXiv preprint arXiv:2305.12025 (2023).

[57] Md Razuan Hossain, Joseph S Najem, Tauhidur Rahman, and Md Sakib Hasan, *Reservoir computing system using biomolecular memristor*, 2021 IEEE 21st International Conference on Nanotechnology (NANO), IEEE, 2021, pp. 116–119.

[58] Md Razuan Hossain, Partha Sarathi Paul, Maisha Sadia, Anurag Dhungel, Joseph S Najem, and Md Sakib Hasan, *Memristor based reservoir network for chaotic time series prediction*, 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), IEEE, 2022, pp. 1–6.

[59] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew, *Extreme learning machine: theory and applications*, Neurocomputing **70** (2006), no. 1-3, 489–501.

[60] Giacomo Indiveri and Shih-Chii Liu, *Memory and information processing in neuromorphic systems*, Proceedings of the IEEE **103** (2015), no. 8, 1379–1397.

[61] Sergey Ioffe and Christian Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, International conference on machine learning, pmlr, 2015, pp. 448–456.

[62] Herbert Jaeger and Harald Haas, *Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication*, science **304** (2004), no. 5667, 78–80.

[63] YeonJoo Jeong, Jihang Lee, John Moon, Jong Hoon Shin, and Wei D Lu, *K-means data clustering with memristor networks*, Nano letters **18** (2018), no. 7, 4447–4453.

[64] Sung Hyun Jo, Ting Chang, Idongesit Ebong, Bhavitavya B Bhadviya, Pinaki Mazumder, and Wei Lu, *Nanoscale memristor device as synapse in neuromorphic systems*, Nano letters **10** (2010), no. 4, 1297–1301.

[65] VD Juncu, M Rafiei-Naeini, and P Dudek, *Integrated circuit implementation of a compact discrete-time chaos generator*, Analog Integrated Circuits and Signal Processing **46** (2006), 275–280.

[66] Holger Kantz and Thomas Schreiber, *Nonlinear time series analysis*, vol. 7, Cambridge university press, 2004.

[67] Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al., *A comparative study on transformer vs rnn in speech applications*, 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), IEEE, 2019, pp. 449–456.

[68] Abdul Karim Khan and Byoung Hun Lee, *Monolayer mos2 metal insulator transition based memcapacitor modeling with extension to a ternary device*, AIP Advances **6** (2016), no. 9, 095022.

[69] Behnam Kia, Kenneth Mobley, and William L Ditto, *An integrated circuit design for a dynamics-based reconfigurable logic block*, IEEE Transactions on Circuits and Systems II: Express Briefs **64** (2017), no. 6, 715–719.

[70] Kuk-Hwan Kim, Sung Hyun Jo, Siddharth Gaba, and Wei Lu, *Nanoscale resistive memory with intrinsic diode characteristics and long endurance*, Applied Physics Letters **96** (2010), no. 5, 053106.

[71] Youngjin Kim, Minsung Kim, Ji Hyeon Hwang, Tae Whan Kim, Sang-Soo Lee, and Woojin Jeon, *Sustainable resistance switching performance from composite-type reram device based on carbon nanotube@ titania core–shell wires*, Scientific Reports **10** (2020), no. 1, 18830.

[72] Teuvo Kohonen, *Self-organizing maps*, vol. 30, Springer Science & Business Media, 2012.

[73] Subhadeep Koner, Joseph S Najem, Md Sakib Hasan, and Stephen A Sarles, *Memristive plasticity in artificial electrical synapses via geometrically reconfigurable, gramicidin-doped biomembranes*, Nanoscale **11** (2019), no. 40, 18640–18652.

[74] Duygu Kuzum, Rakesh GD Jeyasingh, Byoungil Lee, and H-S Philip Wong, *Nanoelectronic programmable synapses based on phase change materials for brain-inspired computing*, Nano letters **12** (2012), no. 5, 2179–2186.

[75] Daewoong Kwon and In-Young Chung, *Capacitive neural network using charge-stored memory cells for pattern recognition applications*, IEEE Electron device letters **41** (2020), no. 3, 493–496.

[76] Vernon J Lawhern, Amelia J Solon, Nicholas R Waytowich, Stephen M Gordon, Chou P Hung, and Brent J Lance, *Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces*, Journal of neural engineering **15** (2018), no. 5, 056013.

[77] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE **86** (1998), no. 11, 2278–2324.

[78] Can Li, Daniel Belkin, Yunning Li, Peng Yan, Miao Hu, Ning Ge, Hao Jiang, Eric Montgomery, Peng Lin, Zhongrui Wang, et al., *Efficient and self-adaptive in-situ learning in multilayer memristor neural networks*, Nature communications **9** (2018), no. 1, 2385.

[79] Can Li, Miao Hu, Yunning Li, Hao Jiang, Ning Ge, Eric Montgomery, Jiaming Zhang, Wenhao Song, Noraica Dávila, Catherine E Graves, et al., *Analogue signal and image processing with large memristor crossbars*, Nature electronics **1** (2018), no. 1, 52–59.

[80] Longyuan Li, Junchi Yan, Haiyang Wang, and Yaohui Jin, *Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder*, IEEE transactions on neural networks and learning systems **32** (2020), no. 3, 1177–1191.

[81] Shujun Li, Gonzalo Álvarez, Guanrong Chen, and Xuanqin Mou, *Breaking a chaos-noise-based secure communication scheme*, Chaos: An Interdisciplinary Journal of Nonlinear Science **15** (2005), no. 1, 013703.

[82] Shujun Li, Guanrong Chen, and Xuanqin Mou, *On the dynamical degradation of digital piecewise linear chaotic maps*, International journal of Bifurcation and Chaos **15** (2005), no. 10, 3119–3151.

[83] Tien-Yien Li and James A Yorke, *Period three implies chaos*, The theory of chaotic attractors (2004), 77–84.

[84] Yan Li, Peng Paul Wen, et al., *Clustering technique-based least square support vector machine for eeg signal classification*, Computer methods and programs in biomedicine **104** (2011), no. 3, 358–372.

[85] Zhiyuan Li, Wei Tang, Beining Zhang, Rui Yang, and Xiangshui Miao, *Emerging memristive neurons for neuromorphic computing and sensing*, Science and Technology of Advanced Materials **24** (2023), no. 1, 2188878.

[86] Juan Carlos López, *A fresh look at paired-pulse facilitation*, Nature Reviews Neuroscience **2** (2001), no. 5, 307–307.

[87] Edward N Lorenz, *Deterministic nonperiodic flow*, Journal of atmospheric sciences **20** (1963), no. 2, 130–141.

[88] Fabien Lotte, Marco Congedo, Anatole Lécuyer, Fabrice Lamarche, and Bruno Arnaldi, *A review of classification algorithms for eeg-based brain–computer interfaces*, Journal of neural engineering **4** (2007), no. 2, R1.

[89] Qianli Ma, Lifeng Shen, and Garrison W Cottrell, *Deepr-esn: A deep projection-encoding echo-state network*, Information Sciences **511** (2020), 152–171.

[90] Wolfgang Maass, *Liquid state machines: motivation, theory, and applications*, Computability in context: computation and logic in the real world (2011), 275–296.

[91] Badruddoja Majumder, Sakib Hasan, Mesbah Uddin, and Garrett S Rose, *Chaos computing for mitigating side channel attack*, 2018 IEEE international symposium on hardware oriented security and trust (HOST), IEEE, 2018, pp. 143–146.

[92] Md Badruddoja Majumder, Md Sakib Hasan, Aysha Shanta, Mesbah Uddin, and Garrett Rose, *Design for eliminating operation specific power signatures from digital logic*, Proceedings of the 2019 on Great Lakes Symposium on VLSI, 2019, pp. 111–116.

[93] Zeeshan Khawar Malik, Amir Hussain, and Qingming Jonathan Wu, *Multilayered echo state machine: A novel architecture and algorithm*, IEEE Transactions on cybernetics **47** (2016), no. 4, 946–959.

[94] Joshua J Maraj, Kevin PT Haughn, Daniel J Inman, and Stephen A Sarles, *Sensory adaptation in biomolecular memristors improves reservoir computing performance*, Advanced Intelligent Systems (2023), 2300049.

[95] Joshua J Maraj, Joseph S Najem, Jessie D Ringley, Ryan J Weiss, Garrett S Rose, and Stephen A Sarles, *Short-term facilitation-then-depression enables adaptive processing of sensory inputs by ion channels in biomolecular synapses*, ACS Applied Electronic Materials **3** (2021), no. 10, 4448–4458.

[96] Jesus Martinez-del Rincon, Maria J Santofimia, Xavier del Toro, Jesus Barba, Francisca Romero, Patricia Navas, and Juan C Lopez, *Non-linear classifiers applied to eeg analysis for epilepsy seizure detection*, Expert Systems with Applications **86** (2017), 99–112.

[97] J Martinez-Rincon, Massimiliano Di Ventra, and Yu V Pershin, *Solid-state memcapacitive system with negative and diverging capacitance*, Physical Review B **81** (2010), no. 19, 195430.

[98] Robert M May, *Simple mathematical models with very complicated dynamics*, Nature **261** (1976), 459–467.

[99] Mark Mayford, Steven A Siegelbaum, and Eric R Kandel, *Synapses and memory storage*, Cold Spring Harbor perspectives in biology **4** (2012), no. 6, a005751.

[100] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al., *A million spiking-neuron integrated circuit with a scalable communication network and interface*, Science **345** (2014), no. 6197, 668–673.

[101] MGA Mohamed, HyungWon Kim, and Tae-Won Cho, *Modeling of memristive and memcapacitive behaviors in metal-oxide junctions*, The Scientific World Journal **2015** (2015).

[102] John Moon, Wen Ma, Jong Hoon Shin, Fuxi Cai, Chao Du, Seung Hwan Lee, and Wei D Lu, *Temporal data classification and forecasting using a memristor-based reservoir computing system*, Nature Electronics **2** (2019), no. 10, 480–487.

[103] John Moon, Yuting Wu, and Wei D Lu, *Hierarchical architectures in reservoir computing systems*, Neuromorphic Computing and Engineering **1** (2021), no. 1, 014006.

[104] Murugappan Murugappan, Nagarajan Ramachandran, Yaacob Sazali, et al., *Classification of human emotion from eeg using discrete wavelet transform*, Journal of biomedical science and engineering **3** (2010), no. 04, 390.

[105] Joseph S Najem, Md Sakib Hasan, R Stanley Williams, Ryan J Weiss, Garrett S Rose, Graham J Taylor, Stephen A Sarles, and C Patrick Collier, *Dynamical nonlinear memory capacitance in biomimetic membranes*, Nature communications **10** (2019), no. 1, 3239.

[106] Joseph S Najem, Graham J Taylor, Nick Armendarez, Ryan J Weiss, Md Sakib Hasan, Garrett S Rose, Catherine D Schuman, Alex Belianinov, Stephen A Sarles, and C Patrick Collier, *Assembly and characterization of biomolecular memristors consisting of ion channel-doped lipid membranes*, JoVE (Journal of Visualized Experiments) (2019), no. 145, e58998.

[107] Joseph S Najem, Graham J Taylor, Ryan J Weiss, Md Sakib Hasan, Garrett Rose, Catherine D Schuman, Alex Belianinov, C Patrick Collier, and Stephen A Sarles, *Memristive ion channel-doped biomembranes as synaptic mimics*, ACS nano **12** (2018), no. 5, 4702–4711.

[108] Rodrigo Neves, *An overview of deep learning strategies for time series prediction*, (2018).

[109] Vivek Prakash Nigam and Daniel Graupe, *A neural-network-based detection of epilepsy*, Neurological research **26** (2004), no. 1, 55–60.

[110] A Nishad, A Upadhyay, G Ravi Shankar Reddy, and V Bajaj, *Classification of epileptic eeg signals using sparse spectrum based empirical wavelet transform*, Electronics Letters **56** (2020), no. 25, 1370–1372.

[111] Christian Oestreicher, *A history of chaos theory*, Dialogues in clinical neuroscience (2022).

[112] Mark JL Orr et al., *Introduction to radial basis function networks*, 1996.

[113] Edward Ott, *Chaos in dynamical systems*, Cambridge university press, 2002.

[114] Rami J Oweis and Enas W Abdulhay, *Seizure classification in eeg signals utilizing hilbert-huang transform*, Biomedical engineering online **10** (2011), no. 1, 1–15.

[115] TN Palmer, A Döring, and G Seregin, *The real butterfly effect*, Nonlinearity **27** (2014), no. 9, R123.

[116] Sinno Jialin Pan and Qiang Yang, *A survey on transfer learning*, IEEE Transactions on knowledge and data engineering **22** (2010), no. 10, 1345–1359.

[117] Narendra K Pareek, Vinod Patidar, and Krishan K Sud, *Image encryption using chaotic logistic map*, Image and vision computing **24** (2006), no. 9, 926–934.

[118] Partha Sarathi Paul, Anurag Dhungel, Maisha Sadia, Md Razuan Hossain, and Md Sakib Hasan, *Self-parameterized chaotic map for low-cost robust chaos*, Journal of Low Power Electronics and Applications **13** (2023), no. 1, 18.

[119] Partha Sarathi Paul, Parker Hardy, Maisha Sadia, and MD Sakib Hasan, *A 2d chaotic oscillator for analog ic*, IEEE Open Journal of Circuits and Systems **3** (2022), 263–273.

[120] Partha Sarathi Paul, Maisha Sadia, Anurag Dhungel, Parker Hardy, and Md Sakib Hasan, *Split-slope chaotic map providing high entropy across wide range*, 2023 24th International Symposium on Quality Electronic Design (ISQED), IEEE, 2023, pp. 1–6.

[121] Partha Sarathi Paul, Maisha Sadia, and Md Sakib Hasan, *Nonlinear parameter modulation for low-cost enhancement of chaotic entropy*, 2023 IEEE 16th Dallas Circuits and Systems Conference (DCAS), IEEE, 2023, pp. 1–4.

[122] Partha Sarathi Paul, Maisha Sadia, Md Razuan Hossain, Barry Muldrey, and Md Sakib Hasan, *Design of a low-overhead random number generator using cmos-based cascaded chaotic maps*, Proceedings of the 2021 on Great Lakes Symposium on VLSI, 2021, pp. 109–114.

[123] ———, *Cascading cmos-based chaotic maps for improved performance and its application in efficient rng design*, IEEE Access **10** (2022), 33758–33770.

[124] Louis M Pecora and Thomas L Carroll, *Synchronization in chaotic systems*, Physical review letters **64** (1990), no. 8, 821.

[125] G Pedretti, V Milo, S Ambrogio, R Carboni, S Bianchi, A Calderoni, N Ramaswamy, AS Spinelli, and D Ielmini, *Memristive neural network for on-line learning and tracking with brain-inspired spike timing dependent plasticity*, Scientific reports **7** (2017), no. 1, 1–10.

[126] Yuriy V Pershin and Massimiliano Di Ventra, *Memcapacitive neural networks*, Electronics letters **50** (2014), no. 3, 141–143.

[127] Yuriy V Pershin, Steven La Fontaine, and Massimiliano Di Ventra, *Memristive model of amoeba learning*, APS March Meeting Abstracts, vol. 2010, 2010, pp. Z10–002.

[128] Zhou Ping, Cheng Yuan-Ming, and Kuang Fei, *Synchronization between fractional-order chaotic systems and integer orders chaotic systems (fractional-order chaotic systems)*, Chinese Physics B **19** (2010), no. 9, 090503.

[129] Kemal Polat and Salih Güneş, *Classification of epileptiform eeg using a hybrid system based on decision tree classifier and fast fourier transform*, Applied Mathematics and Computation **187** (2007), no. 2, 1017–1026.

[130] Mark Rees, Dave Kelly, and Ottar N Bjørnstad, *Snow tussocks, chaos, and the evolution of mast seeding*, The American Naturalist **160** (2002), no. 1, 44–59.

[131] J Requena and DA Haydon, *The lippmann equation and the characterization of black lipid films*, Journal of Colloid and Interface Science **51** (1975), no. 2, 315–327.

[132] John JM Reynolds, James S Plank, Catherine D Schuman, Grant R Bruer, Adam W Disney, Mark E Dean, and Garrett S Rose, *A comparison of neuromorphic classification tasks*, Proceedings of the International Conference on Neuromorphic Systems, 2018, pp. 1–8.

[133] Ali Rodan and Peter Tino, *Minimum complexity echo state network*, IEEE transactions on neural networks **22** (2010), no. 1, 131–144.

[134] Andrea Roli and Luca Melandri, *Introduction to reservoir computing methods.*

[135] Francisco J Romero, Akiko Ohata, Alejandro Toral-Lopez, Andres Godoy, Diego P Morales, and Noel Rodriguez, *Memcapacitor and meminductor circuit emulators: A review*, Electronics **10** (2021), no. 11, 1225.

[136] Michael Rosenblum and Jürgen Kurths, *Synchronization: a universal concept in nonlinear science*, Cambridge University Press, 2003.

[137] Nikolai F Rulkov, Mikhail M Sushchik, Lev S Tsimring, and Henry DI Abarbanel, *Generalized synchronization of chaos in directionally coupled chaotic systems*, Physical Review E **51** (1995), no. 2, 980.

[138] Maisha Sadia, Partha Sarathi Paul, and Md Sakib Hasan, *Compact analog chaotic map designs using soi four-gate transistors*, IEEE Access (2023).

[139] Maisha Sadia, Partha Sarathi Paul, Md Razuan Hossain, and Md Sakib Hasan, *Design and analysis of a multi-parameter discrete chaotic map using only three soi four-gate transistors*, SoutheastCon 2021, IEEE, 2021, pp. 1–7.

[140] ______, *Design and analysis of a multi-parameter discrete chaotic map using only three soi four-gate transistors*, SoutheastCon 2021, IEEE, 2021, pp. 1–7.

[141] Maisha Sadia, Partha Sarathi Paul, Md Razuan Hossain, Barry Muldrey, and Md Sakib Hasan, *Robust chaos with novel 4-transistor maps*, IEEE Transactions on Circuits and Systems II: Express Briefs **70** (2022), no. 3, 914–918.

[142] Kaveh Samiee, Peter Kovacs, and Moncef Gabbouj, *Epileptic seizure classification of eeg time-series using rational discrete short-time fourier transform*, IEEE transactions on Biomedical Engineering **62** (2014), no. 2, 541–552.

[143] Yulia Sandamirskaya, Mohsen Kaboli, Jorg Conradt, and Tansu Celikel, *Neuromorphic computing hardware and neural architectures for robotics*, Science Robotics **7** (2022), no. 67, eabl8419.

[144] Stephen A Sarles and Donald J Leo, *Regulated attachment method for reconstituting lipid bilayers of prescribed size within flexible substrates*, Analytical chemistry **82** (2010), no. 3, 959–966.

[145] Murat H Sazli, *A brief review of feed-forward neural networks*, Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering **50** (2006), no. 01.

[146] Johannes Schemmel, Daniel Brüderle, Andreas Grübl, Matthias Hock, Karlheinz Meier, and Sebastian Millner, *A wafer-scale neuromorphic hardware system for large-scale neural modeling*, 2010 ieee international symposium on circuits and systems (iscas), IEEE, 2010, pp. 1947–1950.

[147] Catherine D Schuman, Shruti R Kulkarni, Maryam Parsa, J Parker Mitchell, Prasanna Date, and Bill Kay, *Opportunities for neuromorphic computing algorithms and applications*, Nature Computational Science **2** (2022), no. 1, 10–19.

[148] Aysha S Shanta, Md Sakib Hasan, Md Badruddoja Majumder, and Garrett S Rose, *Design of a lightweight reconfigurable prng using three transistor chaotic map*, 2019 IEEE 62nd international midwest symposium on circuits and systems (MWSCAS), IEEE, 2019, pp. 586–589.

[149] Aysha S Shanta, Md Badruddoja Majumder, Md Sakib Hasan, and Garrett S Rose, *Physically unclonable and reconfigurable computing system (purcs) for hardware security applications*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **40** (2020), no. 3, 405–418.

[150] Aysha S Shanta, Md Badruddoja Majumder, Md Sakib Hasan, Mesbah Uddin, and Garrett S Rose, *Design of a reconfigurable chaos gate with enhanced functionality space in 65nm cmos*, 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS), IEEE, 2018, pp. 1016–1019.

[151] Alex Sherstinsky, *Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network*, Physica D: Nonlinear Phenomena **404** (2020), 132306.

[152] Troy Shinbrot, William Ditto, Celso Grebogi, Edward Ott, Mark Spano, and James A Yorke, *Using the sensitive dependence of chaos (the "butterfly effect") to direct trajectories in an experimental chaotic system*, Physical review letters **68** (1992), no. 19, 2863.

[153] Patrice Y Simard, David Steinkraus, John C Platt, et al., *Best practices for convolutional neural networks applied to visual document analysis.*, Icdar, vol. 3, Edinburgh, 2003.

[154] Caterina Soldano, Saikat Talapatra, and Swastik Kar, *Carbon nanotubes and graphene nanoribbons: potentials for nanoscale electrical interconnects*, Electronics **2** (2013), no. 3, 280–314.

[155] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting*, The journal of machine learning research **15** (2014), no. 1, 1929–1958.

[156] Terrence C Stewart, *A technical overview of the neural engineering framework*, University of Waterloo **110** (2012).

[157] Steven H Strogartz, *Nonlinear dynamics and chaos: With applications to physics, biology*, Chemistry and Engineering **441** (1994).

[158] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams, *The missing memristor found*, nature **453** (2008), no. 7191, 80–83.

[159] Abdulhamit Subasi, *Eeg signal classification using wavelet feature extraction and a mixture of expert model*, Expert Systems with Applications **32** (2007), no. 4, 1084–1093.

[160] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose, *Recent advances in physical reservoir computing: A review*, Neural Networks **115** (2019), 100–123.

[161] Graham J Taylor, Guru A Venkatesan, C Patrick Collier, and Stephen A Sarles, *Direct in situ measurement of specific capacitance, monolayer tension, and bilayer tension in a droplet interface bilayer*, Soft matter **11** (2015), no. 38, 7592–7605.

[162] BH Tongue, *Characteristics of numerical simulations of chaotic systems*, (1987).

[163] Alexandros T Tzallas, Markos G Tsipouras, and Dimitrios I Fotiadis, *Epileptic seizure detection in eegs using time–frequency analysis*, IEEE transactions on information technology in biomedicine **13** (2009), no. 5, 703–710.

[164] Sundarapandian Vaidyanathan and Christos Volos, *Advances and applications in chaotic systems*, vol. 636, Springer, 2016.

[165] Martijn P Van Den Heuvel, Cornelis J Stam, René S Kahn, and Hilleke E Hulshoff Pol, *Efficiency of functional brain networks and intellectual performance*, Journal of Neuroscience **29** (2009), no. 23, 7619–7624.

[166] David Verstraeten, Benjamin Schrauwen, Michiel d'Haene, and Dirk Stroobandt, *An experimental unification of reservoir computing methods*, Neural networks **20** (2007), no. 3, 391–403.

[167] Marcelo Viana, *Lectures on lyapunov exponents*, vol. 145, Cambridge University Press, 2014.

[168] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus, *Regularization of neural networks using dropconnect*, International conference on machine learning, PMLR, 2013, pp. 1058–1066.

[169] Deng Wang, Duoqian Miao, and Chen Xie, *Best basis-based wavelet packet entropy feature extraction and hierarchical eeg classification for epileptic detection*, Expert Systems with Applications **38** (2011), no. 11, 14314–14320.

[170] Feng Wang and David MJ Tax, *Survey on the attention based rnn model and its applications in computer vision*, arXiv preprint arXiv:1601.06823 (2016).

[171] Guanyu Wang, Dajun Chen, Jianya Lin, and Xing Chen, *The application of chaotic oscillators to weak signal detection*, IEEE Transactions on industrial electronics **46** (1999), no. 2, 440–444.

[172] Laiyuan Wang, Tao Zhang, Junhao Shen, Jin Huang, Wen Li, Wei Shi, Wei Huang, and Mingdong Yi, *Flexibly photo-regulated brain-inspired functions in flexible neuromorphic transistors*, ACS Applied Materials & Interfaces (2023).

[173] Xiaomin Wang, Wenfang Zhang, Wei Guo, and Jiashu Zhang, *Secure chaotic system with application to chaotic ciphers*, Information Sciences **221** (2013), 555–570.

[174] Zhongrui Wang, Mingyi Rao, Jin-Woo Han, Jiaming Zhang, Peng Lin, Yunning Li, Can Li, Wenhao Song, Shiva Asapu, Rivu Midya, et al., *Capacitive neural network with neuro-transistors*, Nature communications **9** (2018), no. 1, 3208.

[175] Ryan Weiss, Joseph S Najem, Md Sakib Hasan, Catherine D Schuman, Alex Belianinov, C Patrick Collier, Stephen A Sarles, and Garrett S Rose, *A soft-matter biomolecular memristor synapse for neuromorphic systems*, 2018 IEEE Biomedical Circuits and Systems Conference (BioCAS), IEEE, 2018, pp. 1–4.

[176] R Stanley Williams, *How we found the missing memristor*, IEEE spectrum **45** (2008), no. 12, 28–35.

[177] Alan Wolf, Jack B Swift, Harry L Swinney, and John A Vastano, *Determining lyapunov exponents from a time series*, Physica D: nonlinear phenomena **16** (1985), no. 3, 285–317.

[178] Jianxin Wu, *Introduction to convolutional neural networks*, National Key Lab for Novel Software Technology. Nanjing University. China **5** (2017), no. 23, 495.

[179] Tianyi Xiong, Changwei Li, Xiulan He, Boyang Xie, Jianwei Zong, Yanan Jiang, Wenjie Ma, Fei Wu, Junjie Fei, Ping Yu, et al., *Neuromorphic functions with a polyelectrolyte-confined fluidic memristor*, Science **379** (2023), no. 6628, 156–161.

[180] Weilin Xu, Jingjuan Wang, and Xiaobing Yan, *Advances in memristor-based neural networks*, Frontiers in Nanotechnology **3** (2021), 645995.

[181] J Joshua Yang, Dmitri B Strukov, and Duncan R Stewart, *Memristive devices for computing*, Nature nanotechnology **8** (2013), no. 1, 13–24.

[182] Yuchao Yang, Peng Gao, Linze Li, Xiaoqing Pan, Stefan Tappertzhofen, ShinHyun Choi, Rainer Waser, Ilia Valov, and Wei D Lu, *Electrochemical dynamics of nanoscale metallic inclusions in dielectrics*, Nature communications **5** (2014), no. 1, 4232.

[183] Huang Yi, Sun Shiyu, Duan Xiusheng, and Chen Zhigang, *A study on deep neural networks framework*, 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), IEEE, 2016, pp. 1519–1522.

[184] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze, *Comparative study of cnn and rnn for natural language processing*, arXiv preprint arXiv:1702.01923 (2017).

[185] Tiangui You, Laveen Prabhu Selvaraj, Huizhong Zeng, Wenbo Luo, Nan Du, Danilo Bürger, Ilona Skorupa, Slawomir Prucnal, Alexander Lawerenz, Thomas Mikolajick, et al., *An energy-efficient, bifeo3-coated capacitive switch with integrated memory and demodulation functions*, Advanced Electronic Materials **2** (2016), no. 3, 1500352.

[186] Aaron R Young, Mark E Dean, James S Plank, and Garrett S Rose, *A review of spiking neuromorphic hardware communication systems*, IEEE Access **7** (2019), 135606–135620.

[187] Lei Zhang, Shuai Wang, and Bing Liu, *Deep learning for sentiment analysis: A survey*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8** (2018), no. 4, e1253.

[188] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay, *Deep learning based recommender system: A survey and new perspectives*, ACM computing surveys (CSUR) **52** (2019), no. 1, 1–38.

[189] Xinjiang Zhang, Anping Huang, Qi Hu, Zhisong Xiao, and Paul K Chu, *Neuromorphic computing with memristor crossbar*, physica status solidi (a) **215** (2018), no. 13, 1700875.

[190] Zhewei Zhang, Huzi Cheng, and Tianming Yang, *A recurrent neural network framework for flexible and adaptive decision making based on sequence learning*, PLOS Computational Biology **16** (2020), no. 11, e1008342.

[191] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu, *Object detection with deep learning: A review*, IEEE transactions on neural networks and learning systems **30** (2019), no. 11, 3212–3232.

[192] Qilin Zheng, Zongwei Wang, Nanbo Gong, Zhizhen Yu, Cheng Chen, Yimao Cai, Qianqian Huang, Hao Jiang, Qiangfei Xia, and Ru Huang, *Artificial neural network based on doped hfo 2 ferroelectric capacitors with multilevel characteristics*, IEEE Electron Device Letters **40** (2019), no. 8, 1309–1312.

[193] Yanan Zhong, Jianshi Tang, Xinyi Li, Bin Gao, He Qian, and Huaqiang Wu, *Dynamic memristor-based reservoir computing for high-efficiency temporal signal processing*, Nature communications **12** (2021), no. 1, 1–9.

[194] Yicong Zhou, Long Bao, and CL Philip Chen, *Image encryption using a new parametric switching chaotic system*, Signal processing **93** (2013), no. 11, 3039–3052.

[195] Ye Zhuo, Rivu Midya, Wenhao Song, Zhongrui Wang, Shiva Asapu, Mingyi Rao, Peng Lin, Hao Jiang, Qiangfei Xia, R Stanley Williams, et al., *A dynamical compact model of diffusive and drift memristors for neuromorphic computing*, Advanced Electronic Materials (2021), 2100696.

VITA

Md Razuan Hossain

Email: mhossai3@go.olemiss.edu

Education

- Ph.D. in Electrical & Computer Engineering, University of Mississippi.

August 2019 – July 2023

CGPA: 3.65 out of 4.00

- M.Sc. in Electrical & Computer Engineering, North Dakota State University.

January 2017 – August 2019

CGPA: 3.88 out of 4.00

- Bachelor of Science in Electrical, Electronics, and Communication Engineering, Military Institute of Science and Technology (MIST).

January 2011 – December 2014

CGPA: 3.67 out of 4.00

Experience

- Research Assistant, Electrical & Computer Engineering, University of Mississippi.

August 2019 – July 2023

- Teaching Assistant, Electrical & Computer Engineering, University of

Mississippi.

August 2019 – July 2023

- Lab Instructor, Electrical & Computer Engineering, University of

Mississippi.

Fall 2021, 2022

- Teaching Assistant, Electrical & Computer Engineering, North Dakota

State University.

January 2017 – May 2019

- Research Assistant, Electrical & Computer Engineering, North Dakota State University.

Summer 2018

- Lecturer, Electrical and Electronics Engineering, Prime University.

department.

April 2016- December 2016

- Lecturer and Co-Ordinator, Electrical and Electronics Engineering department, European

University of Bangladesh.

March 2015 – March 2016

Skills

- Microsoft office packages: MS Word, MS Excel, MS PowerPoint, MS Visio
- Application Software: MATLAB, LabVIEW, Simulink, Xilinx Vivado
- Programming & Markup Languages: C, C++, Python, HTML, CSS
- Design Software: OrCAD Capture, OrCAD Pspice, Dsch2, Microwind, LT spice, Cadence, AutoCAD, SolidWorks
- Hardware description language: VHDL, SystemVerilog
- Statistical & Machine Learning Tools: NumPy, SciPy, pandas, Matplotlib, TensorFlow, scikit-learn, Keras
- Operating Systems: Windows, Linux

Selected Projects

- "Bio-inspired memory element modeling with short-term plasticity using Memristors and Memcapacitor" (current work). (Tools: MATLAB, LTspice, Cadence, Machine learning tools)
- "Highly Sensitive Room Temperature Sensor Based on Nanostructured K2W7O22 for Diabetes Diagnosis", 2019. (Tools: MATLAB, LabView, Arduino)
- "Network flow optimization by Genetic Algorithm and load flow analysis by Newton Raphson method in power system," 2014. (Tool: MATLAB)

Journal Papers

- "1D and 2D Chaotic Time Series Prediction using Hierarchical Reservoir Computing System," Md Razuan Hossain, Anurag Dhungel, Maisha Sadia, Partha Sarathi Paul, Joseph S. Najem, Md Sakib Hasan, submitted to International Journal of High Speed Electronics and Systems, April 2023
- "Design, Analysis, and Application of Flipped Product Chaotic System," Md Sakib Hasan, Partha Sarathi Paul, Anurag Dhungel, Maisha Sadia, Md Razuan Hossain. Accepted in IEEE Access, 2022.
- "Robust Chaos with Novel 4-Transistor Maps", Maisha Sadia, Partha Sarathi Paul, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan. IEEE Transactions on Circuits and Systems II, 2022
- "Cascading CMOS-Based Chaotic Maps for Improved Performance and Its Application in Efficient RNG Design," Partha Sarathi Paul, Maisha Sadia, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan. IEEE Access, vol: 5, 2022.
- "The Temperature Effect on the Performance of Acetone Sensor Based on K2W7O22 Nanorods", Md Razuan Hossain, Anna Marie Schornack, Michael Johnson, Qifeng Zhang, Danling Wang. Journal of Nanoscience and

Nanotechnology Applications, 5(1), ScholArena, 2021.

- “Investigation of Different Materials as Acetone Sensors for Application in Type-1 Diabetes Diagnosis”, Md Razuan Hossain, Qifeng Zhang, Michael Johnson, Obinna Ama, and Danling Wang, BJSTR, 2019.
- “Highly Sensitive Room-Temperature Sensor Based on Nanostructured K2W7O22 for Application in the Non-Invasive Diagnosis of Diabetes”, Md Razuan Hossain, Qifeng Zhang, Michael Johnson, and Danling Wang, Sensors 2018.
- “High Sensitive Breath Sensor Based on K2W7O22 Nanorods for Diabetes”, Danling Wang, Qifeng Zhang, Md. Razuan Hossain, and Michael Johnson (2018), IEEE Sensor Journal.
- “Investigation of humidity cross-interference effect on acetone breath sensor based on Nanostructured K2W7O22”, MR.Hossain, Q. Zhang, M. Johnson, and D. Wang. Engineering Press-1(1), 2017.

Conference Papers

- “EEG Signal Classification using Memristor-based Reservoir Computing System,” Md Razuan Hossain, Nicholas X. Armendarez, Ahmed S. Mohamed, Anurag Dhungel, Joseph S. Najem, Md Sakib Hasan, accepted to IEEE Dallas Circuits And Systems Conference, February 2023.
- “Memristor-based Reservoir Network for Chaotic Time Series Prediction,” Md Razuan Hossain, Partha Sarathi Paul, Maisha Sadia, Anurag Dhungel, Joseph S. Najem, Md Sakib Hasan, International Conference on Electrical, Computer, Communications, and Mechatronics Engineering, December 2022.
- “Self-Parameterized Chaotic Map: A Hardware-efficient Scheme Providing Wide Chaotic Range”, Partha Sarathi Paul, Anurag Dhungel, Maisha Sadia, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan, IEEE International

Conference on Electronics, Circuits & Systems 2021, December 2021.

- “Design and Application of a Novel 4-Transistor Chaotic Map with Robust Performance”, Maisha Sadia, Partha Sarathi Paul, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan. IEEE International Conference on Electronics, Circuits and Systems 202, December 2021.
- “Reservoir Computing System using Biomolecular Memristor”, Md Razuan Hossain, Joseph Najem, Tauhidur Rahman, Md Sakib Hasan. IEEE Nano Conference, August 2021.
- “Design of a Low-Overhead Random Number Generator Using CMOS-based Cascaded Chaotic Maps”, Partha Sarathi Paul, Maisha Sadia, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan. Proceedings of the 2021 on Great Lakes Symposium on VLSI, Pages 109-114, June 2021.
- “Design of a Weighted Average Chaotic System for Robust Chaotic Operation”,- Md Sakib Hasan, Partha Sarathi Paul, Maisha Sadia, Md Razuan Hossain, Accepted on MWSCAS conference 2021, May 2021.
- “Design of a Low-Overhead Random Number Generator Using CMOS-based Cascaded Chaotic Maps”- Partha Sarathi Paul, Maisha Sadia, Md Razuan Hossain, Barry Muldrey, Md Sakib Hasan. Accepted on GLSVLSI conference 2021, April 2021.
- “Design and Analysis of a Multi-Parameter Discrete Chaotic Map Using Only Three SOI Four-Gate Transistors”- Maisha Sadia, Partha Sarathi Paul, Md Razuan Hossain, and Md Sakib Hasan. Published on Southeast Conference 2021, January 2021.
- “Integrated Circuit Design of an Improved Discrete Chaotic Map by Averaging Multiple Seed Maps”- Md Sakib Hasan, Partha Sarathi Paul, Maisha Sadia, and Md Razuan Hossain. Published on Southeast Conference 2021, January 2021.

- “Network Flow Optimization by Genetic Algorithm and Load Flow Analysis by Newton Raphson Method in Power System.” - Md Razuan Hossain, M. Shamim Kaiser, Fahmid Iftekher Ali, Md Monjurul Alam Rizvi. Published at 2nd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), May 2015.

Book Chapter

- “Semiconductor Device Modeling and Simulation for Electronic Circuit Design” -, Samira Shamsir, Md Sakib Hasan, Omiya Hassan, Partha Sarathi Paul, Md Razuan Hossain, Syed K Islam. Book – “Modeling and Simulation in Engineering”, Published at “IntechOpen”. Publication date: 4/29/2020.

Reviewer for International Journal and Conferences

- 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI 2020) (sub reviewer) - IEEE International Conference on Nanotechnology, (IEEE NANO 2021,2022)
- 2nd International Conference on Computing Advancement (ICCA 2022)

Co-curricular

- member at IEEE-HKN (2017- Present)
- member at Engineering in Medicine and Biology Society (EMBS) (2018-2019)
- member at IEEE (2020- Present)

Activities

- volunteer at ICEEICT conference – 2014
- organizer of EEE club at European University of Bangladesh-2015
- volunteer at International Conference on Recent Advancements in Medicine

and Medical Science -2019

Honors and Awards

- “Outstanding Teaching Assistant “ in the Engineering department (2022).

- “MIST Dean List” in level 2 and 3, (2012 2013).

- “MIST Commandant List” in level 4, (2014).