Memcapacitive Reservoir Computing Architectures

Dat Tien Tran
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/open_access_etds

Part of the Computer Engineering Commons

Let us know how access to this document benefits you.

Recommended Citation
https://doi.org/10.15760/etd.6877

This Dissertation is brought to you for free and open access. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.
Memcapacitive Reservoir Computing Architectures

by

Dat Tien Tran

A dissertation submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical and Computer Engineering

Dissertation Committee:
Christof Teuscher, Chair
Marek Perkowski
Jun Jiao
Raj Solanki

Portland State University
2019
ABSTRACT

In this thesis, I propose novel brain-inspired and energy-efficient computing systems. Designing such systems has been the forefront goal of neuromorphic scientists over the last few decades. The results from my research show that it is possible to design such systems with emerging nanoscale memcapacitive devices.

Technological development has advanced greatly over the years with the conventional von Neumann architecture. The current architectures and materials, however, will inevitably reach their physical limitations. While conventional computing systems have achieved great performances in general tasks, they are often not power-efficient in performing tasks with large input data, such as natural image recognition and tracking objects in streaming video. Moreover, in the von Neumann architecture, all computations take place in the Central Processing Unit (CPU) and the results are saved in the memory. As a result, information is shuffled back and forth between the memory and the CPU for processing, which creates a bottleneck due to the limited bandwidth of data paths. Adding cache memory and using general-purpose Graphic Processing Units (GPUs) do not completely resolve this bottleneck.

Neuromorphic architectures offer an alternative to the conventional architecture by mimicking the functionality of a biological neural network. In a biological neural network, neurons communicate with each other through a large number of dendrites and synapses. Each neuron (a processing unit) locally processes the information that is stored
in its input synapses (memory units). Distributing information to neurons and localizing computation at the synapse level alleviate the bottleneck problem and allow for the processing of a large amount of data in parallel. Furthermore, biological neural networks are highly adaptable to complex environments, tolerant of system noise and variations, and capable of processing complex information with extremely low power.

Over the past five decades, researchers have proposed various brain-inspired architectures to perform neuromorphic tasks. IBM’s TrueNorth is considered as the state-of-the-art brain-inspired architecture. It has $10^6$ CMOS neurons with $256 \times 256$ programmable synapses and consumes about $60nW/neuron$. Even though TrueNorth is power-efficient, its number of neurons and synapses is nothing compared to a human brain that has $10^{11}$ neurons and each neuron has, on average, 7,000 synaptic connections to other neurons. The human brain only consumes $2.3nW/neuron$.

The memristor brought neuromorphic computing one step closer to the human brain target. A memristor is a passive nano-device that has a memory. Its resistance changes with applied voltages. The resistive change with an applied voltage is similar to the function of a synapse. Memristors have been the prominent option for designing low power systems with high-area density. In fact, Truong and Min reported that an improved memristor-based crossbar performed a neuromorphic task with 50% reduction in area and 48% of power savings compared to CMOS arrays. However, memristive devices, by their nature, are still resistors, and the power consumption is bounded by their resistance. Here, a memcapacitor offers a promising alternative. My initial work indicated that memcapacitive networks performed complex tasks with equivalent performance, compared to memristive networks, but with much higher energy efficiency.

A memcapacitor is also a two-terminal nano-device and its capacitance varies with applied voltages. Similar to a memristor, the capacitance of the memcapacitor changes
with an applied voltage, similar to the function of a synapse. The memcapacitor is a storage device and does not consume static energy. Its switching energy is also small due to its small capacitance (nF to pF range). As a result, networks of memcapacitors have the potential to perform complex tasks with much higher power efficiency.

Several memcapacitive synaptic models have been proposed as artificial synapses. Pershin and Di Ventra illustrated that a memcapacitor with two diodes has the functionality of a synapse. Flak suggested that a memcapacitor behaves as a synapse when it is connected with three CMOS switches in a Cellular Nanoscale Network (CNN). Li et al. demonstrated that when four identical memcapacitors are connected in a bridge network, they characterize the function of a synapse as well.

Reservoir Computing (RC) has been used to explain higher-order cognitive functions and the interaction of short-term memory with other cognitive processes. Rigotti et al. observed that a dynamic system with short-term memory is essential in defining the internal brain states of a test agent. Although both traditional Recurrent Neural Networks (RNNs) and RC are dynamical systems, RC has a great benefit over RNNs due to the fact that the learning process of RC is simple and based on the training of the output layer. RC harnesses the computing nature of a random network of nonlinear devices, such as memcapacitors.

Appeltant et al. showed that RC with a simplified reservoir structure is sufficient to perform speech recognition. Fewer nonlinear units connecting in a delay feedback loop provide enough dynamic responses for RC. Fewer units in reservoirs mean fewer connections and inputs, and therefore lower power consumption.

As Goudarzi and Teuscher indicated, RC architectures still have inherent challenges that need to be addressed. First, theoretical studies have shown that both regular and random reservoirs achieve similar performances for particular tasks. A random reservoir,
however, is more appropriate for unstructured networks of nanoscale devices. What is the role of network structure in RC for solving a task (Q1)?

Secondly, the nonlinear characteristics of nanoscale devices contribute directly to the dynamics of a physical network, which influences the overall performance of an RC system. To what degree is a mixture of nonlinear devices able to improve the performances of reservoirs (Q2)?

Thirdly, modularity, such as CMOS circuits in a digital building, is an essential key in building a complex system from fundamental blocks. Is hierarchical RCs able to solve complex tasks? What network topologies/hierarchies will lead to optimal performance? What is the learning complexity of such a system (Q3)?

My research goal is to address the above RC challenges by exploring memcapacitive reservoir architectures. The analysis of memcapacitive monolithic reservoirs addresses both questions Q1 and Q2 above by showing that *Small-World Power-Law* (SWPL) structure is an optimal topological structure for RCs to perform time series prediction (NARMA-10), temporal recognition (Isolate Spoken Digits), and spatial task (MNIST) with minimal power consumption. On average, the SWPL reservoirs reduce significantly the power consumption by a factor of $1.21 \times$, $31 \times$, and $31.2 \times$ compared to the regular, the random, and the small-world reservoirs, respectively. Further analysis of SWPL structures underlines that high locality $\alpha$ and low randomness $\beta$ decrease the cost to the systems in terms of wiring and nanowire dissipated power but do not guarantee the optimal performance of reservoirs. With a genetic algorithm to refine network structure, SWPL reservoirs with optimal network parameters are able to achieve comparable performance with less power. Compared to the regular reservoirs, the SWPL reservoirs consume less power, by a factor of $1.3 \times$, $1.4 \times$, and $1.5 \times$. Similarly, compared to the random topology, the SWPL reservoirs save power consumption by a factor of $4.8 \times$, $1.6 \times$, and $2.1 \times$, respectively. The
simulation results of mixed-device reservoirs (memristive and memcapacitive reservoirs) provide evidence that the combination of memristive and memcapacitive devices potentially enhances the nonlinear dynamics of reservoirs in three tasks: NARMA-10, Isolated Spoken Digits, and MNIST.

In addressing the third question (Q3), the kernel quality measurements show that hierarchical reservoirs have better dynamic responses than monolithic reservoirs. The improvement of dynamic responses allows hierarchical reservoirs to achieve comparable performance for Isolated Spoken Digit tasks but with less power consumption by a factor of $1.4x, 8.8x, 9.5x$, and $6.3x$ for delay-line, delay-line feedback, simple cycle, and random structures, respectively. Similarly, for the CIFAR-10 image tasks, hierarchical reservoirs gain higher performance with less power, by a factor of $5.6x, 4.2x, 4.8x$, and $1.9x$. The results suggest that hierarchical reservoirs have better dynamics than the monolithic reservoirs to solve sufficiently complex tasks.

Although the performance of deep mem-device reservoirs is low compared to the state-of-the-art deep Echo State Networks, the initial results demonstrate that deep mem-device reservoirs are able to solve a high-dimensional and complex task such as polyphonic music task. The performance of deep mem-device reservoirs can be further improved with better settings of network parameters and architectures.

My research illustrates the potentials of novel memcapacitive systems with SWPL structures that are brain-inspired and energy-efficient in performing tasks. My research offers novel memcapacitive systems that are applicable to low-power applications, such as mobile devices and the Internet of Things (IoT), and provides an initial design step to incorporate nano memcapacitive devices into future applications of nanotechnology.
ACKNOWLEDGEMENTS

First, I would like to express my sincere gratitude to my advisor, Dr. Teuscher, for the continuous support of my Ph.D. study and for his patience, motivation, wisdom, and knowledge. His guidance has helped me in the time of my research and in the time of writing my dissertation. I also thank the committee members for their guidance and their support of my dissertation.

Secondly, I would like to express my gratitude to the Society of Jesus, in general, and to the Jesuit West Province, in particular, who have been the major source of support and nourishment, not only for my vocation as a Jesuit but also for my motivation to pursue a doctoral study in Electrical Engineering.

Thirdly, I would like to thank Fr. Roger de la Rosa, S. J. for his generous time and effort in helpful discussions and in proofreading my dissertation.

Fourthly, I want to thank my fellow colleagues in the Teuscher lab for stimulating discussions, for collaborating to work on projects and articles, for developing helpful tools, and for the fun that we have had in the last several years.

Lastly, I would like to thank my family members and friends who always encourage me to overcome challenges and difficulties in my study and in my ministry with their love, support, and care.
# TABLE OF CONTENTS

Abstract .......................... i

Acknowledgements ..................... vi

List of Tables ........................ x

List of Figures ........................ xi

Statutory Declaration ................. xix

1 Motivation .......................... 1

2 Research Overview .................. 12
   2.1 Network Dynamics in Memcapacitive Reservoir Computing .......... 13
   2.2 Network Dynamics in Hierarchical Reservoirs .................... 16

3 Background ......................... 18
   3.1 Memristive Devices ................. 18
   3.2 Memcapacitive Devices ............... 20
      3.2.1 Bipolar Memcapacitive System with Threshold ............... 21
      3.2.2 Memcapacitive Characteristics of Metal-Oxide Junctions .. 23
      3.2.3 Bistable Elastic-Membrane Memcapacitor .................. 28
      3.2.4 Multilayer Memcapacitive System ....................... 29
   3.3 Artificial Neural Networks ........ 33
   3.4 Reservoir Computing ................. 35
      3.4.1 Separation Property ................... 38
      3.4.2 Approximation Property .................. 38
      3.4.3 Fading Memory ......................... 39
      3.4.4 Spectral Radius ....................... 39
   3.5 Terminology ........................ 40
# TABLE OF CONTENTS

## 4 Related Work

4.1 Memcapacitive Synapse .................................................. 44
  4.1.1 Memcapacitive Synapse in Spiking Network .................... 44
  4.1.2 Memcapacitive Synapse in Cellular Network .................... 45
  4.1.3 Bridge Memcapacitive Synapse ................................. 46

4.2 Hardware Reservoir Architectures ................................. 47
  4.2.1 CMOS-Based Reservoirs ........................................ 47
  4.2.2 Memristive Crossbar Reservoirs .............................. 48
  4.2.3 Memristive Reservoirs ........................................ 52

4.3 Hierarchical Reservoirs ............................................ 54
  4.3.1 Memristive Hierarchical Reservoir ........................... 54
  4.3.2 Multiple Layer Reservoir Computing .......................... 55
  4.3.3 Deep Reservoir Computing .................................. 56

## 5 Memcapacitive Reservoir Computing Architecture ............. 58

5.1 Memcapacitive Reservoir ........................................... 58
5.2 Small-world Networks ................................................ 62
5.3 Small-world Power-law Networks .................................. 63

## 6 Methodology ............................................................ 66

6.1 Volatile Memcapacitive Devices ................................... 66
  6.1.1 Modification of Memcapacitive Biolek Model ................. 66
  6.1.2 Modification of Memcapacitive Mohamed Model .............. 68
  6.1.3 Summary of Device Models .................................... 72

6.2 Reservoir Property Measurements ................................ 72
  6.2.1 Fading Memory ................................................ 73
  6.2.2 Separation .................................................... 73
  6.2.3 Kernel Quality ............................................... 74
  6.2.4 Lyapunov Exponent .......................................... 75

6.3 Genetic Algorithm .................................................... 75
6.4 Benchmarks and Validation ........................................ 76
  6.4.1 NARMA-10 as a Time-series Task ............................... 76
  6.4.2 Isolated Spoken Digits as a Temporal Task .................. 77
  6.4.3 MNIST Recognition as a Spatial Task ......................... 78
  6.4.4 CIFAR-10 Recognition as a Complex Spatial Task ........... 80

6.5 Baseline Reference Networks ...................................... 81
6.6 Nano Wire Model and Power Calculation .......................... 82

## 7 Network Dynamics in Memcapacitive Reservoir Computing .... 84

7.1 Fading Memory Effect ............................................... 85
7.2 Influence of Topological Structure on Power Consumption . 88
## TABLE OF CONTENTS

7.3 Property of Small-world Power-law Networks .................................. 91
7.4 Optimizing Parameters for Specific Tasks ......................................... 100
7.5 Mixed Mem-device Reservoirs ......................................................... 105
7.6 Spiking Input Signals ..................................................................... 110
7.7 Device Faults and Variations ......................................................... 114
    7.7.1 Fault and Variation Model ..................................................... 115
    7.7.2 Open-fault Simulation Results ............................................. 116
    7.7.3 Short-fault Simulation Results ............................................ 117
    7.7.4 Device Variation Simulation Results .................................... 118

8 Network Dynamics in Hierarchical Reservoirs ..................................... 120
    8.1 Hierarchical Reservoir Computing ............................................. 120
    8.1.1 Hierarchical Structure ......................................................... 121
    8.1.2 Hierarchical Reservoir Dynamics ....................................... 123
    8.1.3 Performance and Power Consumption of Hierarchical Reservoirs 125
    8.1.4 Optimizing Parameters for Hierarchical Reservoirs ............... 127
    8.1.5 Mixed Mem-device Hierarchical Reservoirs ......................... 130
    8.2 Deep Reservoir Computing ....................................................... 135
    8.2.1 Deep Memcapacitive Reservoir Architecture ....................... 136
    8.2.2 Polyphonic Music Task ....................................................... 137
    8.2.3 Simulation Results ............................................................. 139

9 Conclusions and Future Work ........................................................... 142
    9.1 Conclusion ........................................................................... 142
    9.2 List of Contributions .............................................................. 145
    9.3 Future Work ......................................................................... 148
    9.4 List of Publications ............................................................... 149

Bibliography ......................................................................................... 150
# LIST OF TABLES

Table 6.1 Physical Characteristics of Mem-devices ........................................ 72
Table 7.1 Device State Parameters ................................................................. 85
Table 7.3 Time Step Settings for Reservoir State .............................................. 88
Table 7.5 Input and Output Settings for Reservoirs .......................................... 89
Table 7.7 Input Scaling Factors for Simulation Sweeps ................................... 92
Table 7.9 Optimized Parameters for NARMA-10 Task ....................................... 101
Table 7.11 Optimized Parameters for Spoken Digit Recognition ........................ 102
Table 7.13 Optimized Parameters for Classifying MNIST Images ....................... 102
Table 7.15 Setting Parameters for Mixed-device Reservoirs .............................. 106
Table 7.17 Optimal Percentage of Memristive Devices ..................................... 110
Table 7.19 Spike Settings ................................................................................. 112
Table 8.1 Optimized Parameters for Spoken Digit Recognition Task ................. 128
Table 8.3 Optimized Parameters for CIFAR-10 Image Task ............................... 129
Table 8.5 Parameters for Mixed-device Reservoirs on Spoken Digit Recognition Task ........................................................................................................ 132
Table 8.7 Parameters for Mixed-device Reservoirs on Spoken Digit Recognition Task ........................................................................................................ 134
Table 8.9 Summary for Mixed-device Reservoirs on Spoken Digit Recognition Task ........................................................................................................ 134
Table 8.11 Piano-rolls in Music Datasets ......................................................... 138
Table 8.13 Network Parameters of Deep-mem Reservoirs for the Music Task .... 140
Figure 1.1 Power computation per neuron: SpiNNaker ($3.8 \times 10^9 nW$), BrainScaleS ($5 \times 10^6 nW$), Blue Brain ($2.3 \times 10^7 nW$), ROLLS ($1.6 \times 10^4 nW$), Neurogrid ($5 \times 10^3 nW$), TrueNorth ($60 nW$), and Mammal Brain ($2.3 nW$). The ultimate goal of neuromorphic platforms is to achieve a similar power performance as that of a mammalian brain.

Figure 3.1 Memristor fundamentals. (a) The memristor as the fourth two-terminal circuit element; (b) the nonlinear dependent current through a memristor; (c) the resulting hysteresis loop in the IV plane; (d) the change of device state over time. The device responses (b, c, d) are created from the model presented in [163].

Figure 3.2 Biolek Spice memcapacitive model. All dependent current and voltage sources are defined as: $E_q = V(x)V_{in}$, $G_x = f(V_{in})W(V(x), V_{in})$, and $G_{mc} = ddt(V(q))$. The capacitance $C$ is 1F and the resistance $R$ is $100M\Omega$. $V_{in}$ is the input voltage to the device. The two functions $f()$ and $W()$ are defined in Equations 3.4.

Figure 3.3 The characteristic of Biolek memcapacitive model [22]. The device was simulated with a sine wave of 0.55V at 50kHz: (a) the device exhibited a hysteresis response of electrical charge $Q$ with applied voltage $V$ and (b) the capacitance of the device varied between $C_{min}$ (1pF) and $C_{max}$ (100pF).

Figure 3.4 Metal-oxide junction device. (a) Device formation in a crossbar structure; (b) Filament formation with $x$ is the growth length and $m$ the growth area. Reprinted from [131].

Figure 3.5 The characteristic of Mohamed memcapacitive model [131]. The device was simulated with a sine wave of 1.0V at 1Hz: (a) the device exhibited a hysteresis response of electrical charge $Q$ with applied voltage $V$ and (b) the capacitance of the device varied between $C_{min}$ (1nF) and $C_{max}$ (10nF). The constants of the model were modified for a low input signal of 1Hz: $K_G = 0.4775$, $K_S = 0.6$, $B_G = 2.75$, $B_S = 2.75$, $x_{min} = 0.4$, $x_{max} = 0.9$, $m_{min} = 0.1$, $m_{max} = 0.9$, $d_1 = 5 \times 10^{-10}$, and $d_2 = 5 \times 10^{-10}$. 
Figure 3.6  Bi-stable elastic-membrane memcapacitor. The membrane is changed
between two stable states (state "0" or state "1") when an external voltage
is applied to the plates. .......................................................... 28

Figure 3.7  Responses of bistable elastic-membrane memcapacitor: (a) Charge-
voltage response, (b) capacitance-voltage response. The capacitance of
the device fluctuates between 7pF and 20pF with an applied sine wave
signal. .......................................................... 28

Figure 3.8  Multilayer memcapacitive system. In Figure (a), ±q are the elec-
tric charge of the external plates whereas Qk signifies the electric charge
of an internal metal layer. .......................................................... 30

Figure 3.9  Responses of the two-layer memcapacitive system: (a) Charge-
voltage response, (b) capacitance-voltage response. The device is not
applicable for the function of a synapse due to the infinite capacitance
values when the applied voltage is close to 0V. ............................... 32

Figure 3.10  An example of a 3 × 4 × 2 fully-connected feedforward artificial
neural network. The circles are neurons. In general, the network may
have many hidden layers depending on the task to be solved. ............... 34

Figure 3.11  An example of a reservoir network showing an input layer, the
randomly structured reservoir performing the computation, and the read-
out layer that interprets the internal dynamics of the reservoir. Only links
(Wout) from the reservoir to the readout layer will be trained, not the in-
ternal links (Wres) of the reservoir. .......................................... 36

Figure 3.12  Neural network and random memcapacitive network. (a) The cir-
cles are neurons, wi,j denotes a synaptic connection of strength w connect-
ing neuron i to neuron j; (b) in a memcapacitive network, an electrical
node performs a similar operation of weighted sum as a neuron. .......... 41

Figure 4.1  Memcapacitive synapses in a spiking network. The RC block
functions as an integrate-and-fire neuron that produces a spiking signal
at Vout when the charging voltage on C exceeds a threshold voltage Nout.
Reprinted from [142]. .......................................................... 45

Figure 4.2  Memcapacitive synapses and activation threshold in a network
cell. The cell state can have negative, zero, or positive values depend-
ing on the values of ABi,j, Vyi,j, and Cm. Reprinted from [60]. .......... 46

Figure 4.3  Memcapacitive bridge synapse. (a) The synaptic weight is pro-
grammed with a current source Iin; (b) the three transistors function as
a differential amplifier that converts a voltage difference to an equivalent
current. Reprinted from [113]. .......................................... 46

Figure 4.4  Hardware architecture of a Liquid State Machine (LSM). The left
figure illustrates the memristive crossbar and the connections. The right
figure shows the data flow between units in an LSM. Reprinted from [170]. 49
Figure 4.5 2×4×1 ESN reservoir computing. (a) The reservoir is constructed with random connections; (b) the reservoir is implemented with a memristive double arrays for the input layer and the reservoir. The output layer is also a memristive crossbar. Matched colors for layers are presented. Reprinted from [76]. .......................................................... 50

Figure 4.6 Training process for memristive crossbar ESN. The reservoir output is trained using MATLAB code. Reprinted from [76]. .............. 50

Figure 4.7 Memristive-based reservoir. The reservoir has less than 40 nodes. The output layer is trained with the Genetic Algorithm (GA). Reprinted from [106]. ......................................................... 52

Figure 4.8 3x3 mesh reservoir with internal nodes. Reprinted from [31]. ... 53

Figure 4.9 Physical random network of memristive devices. Various silver nanowires create a self-assembled network with complex interconnections. Reprinted from [166]. .............................................. 54

Figure 4.10 Memristive and hierarchical reservoirs. (a) Random electrical memristive network ; (b) each cluster node is a random memristive network. Reprinted from [30]. .................................................. 55

Figure 4.11 Multiple-layer reservoir computing. Reprinted from [186]. .... 56

Figure 4.12 Architecture of deepESN with $N_L$ layers. $u(t)$ is the input signal, $W_i^{in}$ is the input-to-reservoir weight matrix for layer $i$, $\hat{W}^{(i)}$ is the recurrent reservoir weight matrix for layer $i$, $x^{(i)}(t)$ is the state of the reservoir of layer $i$, and $\theta^{(i)}$ is the weight vector associated to the unitary input bias for layer $i$. Reprinted from [62]. ........................................................................ 57

Figure 5.1 A memcapacitive reservoir network. The reservoir is an electrical network in which memcapacitors are randomly connected through nanowires. The node voltages are the time-dependent state $x(t)$ of the reservoir that provide signals to the readout layer. $W^{\text{res}}(t)$ is the state matrix containing capacitance of memcapacitive devices at time $t$. Only the readout layer is trained by a simple algorithm. .................. 59

Figure 5.2 Artificial neuron and electrical node. (a) an artificial neuron $k$ performs a weighted sum on its inputs; (b) an electrical node $k$ performs a summation on its input voltages. ......................................................... 60

Figure 5.3 Process of creating a small-world power-law network from a regular network. The regular grid network (a), which has an uniform probability distribution between nodes (c), is transformed into a small-power network (b) by rewiring local connections with global links under the power-law probability distribution (d). ........................................ 63
Figure 5.4 SW power-law networks deriving from the same 8x4 grid network with different network parameters. The parameters for each network: (a) $\alpha = 0.4, \beta = 0.8, \gamma = 20, \delta = 2$; (b) $\alpha = 1.2, \beta = 0.56, \gamma = 10, \delta = 4$; (c) $\alpha = 1.7, \beta = 0.22, \gamma = 17, \delta = 1$; and (d) $\alpha = 2, \beta = 0.9, \gamma = 8, \delta = 2$. 64

Figure 6.1 State volatility with the added decay function to the original Biolek model [23]. At $t > 40 \mu s$, without an applied signal, the state of the device will decline gradually over time. 67

Figure 6.2 State volatility of the modified Mohamed model. (a) Without an applied signal, when $t > 1.7s$, the filament declines steadily over time and the filament gap $x(t)$ returns to $x_{max}$; (b) Without any stimulus, the cross sectional area $m(t)$ of the filament declines over time. 70

Figure 6.3 Volatile device capacitance. 71

Figure 6.4 Converting pixel images into input vectors. Each input vector $\vec{x}_k$ contains pixel values of a $m \times n$ image. Pixel images as spatial information are applied to the reservoir in a temporal process from $t_0 \rightarrow t_k$. 78

Figure 6.5 An example of a CIFAR-10 image. A CIFAR-10 color image (a) is converted into a grayscale image (b) for dimension reduction. The original color image (a) has a high dimension of $32 \times 32 \times 3$ whereas the grayscale image has a lower dimension of $32 \times 32$. 80

Figure 7.1 Fading memory of mem-device reservoirs. The settings of reservoir parameters were: $\alpha = 1.02, \beta = 0.71, \delta = 0.0, \gamma = 0.0$; (a) $Vin_2$, one of 117 input pulses, shows the time intervals where the inputs were active [0.0ms, 0.4ms] and inactive [0.4ms, 1.4ms]. Since the internal states and the response time are device-dependent, the measurements of the device states and the response time were scaled to fit into a single display for a comparison. The scaling factors of device states and times are in Table 7.1. All device states were displayed with time measurements in ms, except the SPESN whose states were shown with respect to the number of time steps; (b) the plots of the Euclidean distance of the Chang, Mohamed, Oblea and SPESN reservoirs were shifted downward for clarity. For the SPESN reservoir, the unit for x-axis is the number of time steps ($\times 1e9$). The vertical line is where the Euclidean distances of the reservoir states were approaching a cyclic steady state, indicating that the initial states of mem-devices no longer had any effect on the reservoir states. 86
Figure 7.2 Performance of mem-device reservoirs for different tasks and network topologies. The performance of the ESNs (blue dashed line) and the power consumption of SWPL reservoirs (green dashed line) are the baselines for comparisons. The results suggest that selecting an appropriate topology for the reservoirs, such as regular or SWPL topology, will reduce power consumption without sacrificing their performance.

Figure 7.3 Dynamic measurements and classifications of reservoirs ($\alpha = 1.2$ and $\beta = 0.5$). Reservoirs were simulated using Spoken digit dataset (a, c, and e) and MNIST dataset (b, d, and f). The Lyapunov exponent estimations show a consistent pattern and provide a useful metric to predict the mem-device reservoir performance.

Figure 7.4 Lyapunov exponent estimation, classifications, and wire dissipated powers of reservoirs for the $\beta$ sweep. The Biolek reservoirs were a mem-capacitive type whereas the Chang reservoirs were a memristive type. Reservoirs were simulated with three topological types: highly global ($\alpha = 0.0$), small-world power-law ($\alpha = 1.2$), and highly local ($\alpha = 2.4$). The change in $\beta$ does not affect the performance of the reservoirs but contributes linearly to the dissipated power in nanowires.

Figure 7.5 Lyapunov exponent estimation, classifications, and wire dissipated powers of reservoirs for the $\alpha$ sweep. The Biolek reservoirs were a mem-capacitive type whereas the Chang reservoirs were a memristive type. Reservoirs were simulated with three topological structures: regular ($\beta = 0.0$), small-world power-law ($\beta = 0.5$), and totally random ($\beta = 1.0$). Similar to the $\beta$ sweep, the change in $\alpha$ only contributes to the dissipated power in the nanowires but not the performance.

Figure 7.6 Dissipated power in nanowire resistance of Biolek reservoirs (a) and Chang reservoirs (b) for the spoken digit task. The optimal settings of $\alpha$ and $\beta$ is the blue region with high values of $\alpha$ and low values of $\beta$ that will minimize the dissipated power in the nanowires, and, therefore, the cost of the systems.

Figure 7.7 Performance and performance-power ratios of regular, random, and SWPL reservoirs on three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; in general, the SWPL reservoirs were able to achieve similar performances (a), but their performance-power ratios were higher (b) compared to regular and random reservoirs. SWPL reservoirs with optimized parameters yield the highest performance in term of the performance-power ratio.
Figure 7.8  Number of devices and nanowires utilized by regular, random, and SWPL reservoirs for three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; the SWPL reservoirs used fewer devices (a) and a fewer number of nanowires (b) while maintaining similar performances compared to the regular and random reservoirs. SWPL reservoirs with optimized parameters utilize the least number of devices for the same tasks. 104

Figure 7.9  Total nanowire length and power consumption of regular, random, and SWPL reservoirs for three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; although the SWPL topology required a longer nanowire length (a), the SWPL reservoirs consumed less the total power (b). The results suggest that the number of devices within a reservoir significantly contributes to the total power consumption of the systems. 104

Figure 7.10  Performance and power consumption of mixed mem-device Biolek-Chang and Mohamed-Chang reservoirs with respect to the percentage of the memristive Chang devices in the reservoirs. The performances of the reservoirs were based on the three benchmark tasks: NARMA-10, Isolated Spoken Digits, and MNIST. 107

Figure 7.11  Performance and power consumption of mixed mem-device Biolek-Oblea and Mohamed-Oblea reservoirs with respect to the percentage of the memristive Oblea devices in the reservoirs. The performances of the reservoirs were based on the three benchmark tasks: NARMA-10, Isolated Spoken Digits, and MNIST. 109

Figure 7.12  An example of spiking reservoir computing architecture. The Spike Converter translates spatial stimuli, such as MNIST, into spike trains as inputs to the reservoir. The Spike Accumulator converts the spike train outputs of the reservoir within a time interval \( \Delta t \) to analog values for the Readout Layer. The Readout Layer utilizes a simple learning algorithm, such as ridge regression technique, during the training phase. 111

Figure 7.13  Performances, power consumption, and simulation times of analog and spiking reservoirs. The horizontal dashed and green lines mark the lowest measurements of power consumption. The spiking reservoirs perform better than analog reservoirs in terms of power consumption but also require longer simulation time. 113

Figure 7.14  Power-time products of mem-device reservoirs. The analog reservoirs, in general, performed better than the spiking reservoirs in terms of the power-time product. Although the simulation time is not considered as a significant factor in designing a reservoir, the long simulation time of the reservoir may not be desirable in an optimizing process to find optimal network parameters for a particular task. In this respect, the power-time product offers a trade-off factor in choosing a memcapacitive system. 114
Figure 7.15 Fault and Variation Model; for the open fault, the device is removed (a) whereas the device is replaced with a 0V source (b) to prevent convergent errors; (c) A Gaussian white noise with a standard deviation $\sigma$ is injected into the reservoir as a way to simulate device variations [68].

Figure 7.16 The performance of mem-device reservoirs under the effect of the open-fault. The results indicate that the reservoirs were tolerant to a high percentage of open faults.

Figure 7.17 The performance of mem-device reservoirs under the effect of the open-fault. The impact of short faults is much more severe to the performance of the reservoirs than the open faults.

Figure 7.18 The performance of mem-device reservoirs under the effect of device variations.

Figure 8.1 Hierarchical topologies. (a) Hierarchical reservoir using delay-line topology; (b) hierarchical reservoir using delay-line with feedback topology; (c) hierarchical reservoir using simple cycle topology; (d) hierarchical reservoir using random topology. Each cluster is a SWPL network. (a), (b), and (c) are reprinted from [153].

Figure 8.2 Kernel quality of reservoirs (monolithic and hierarchical) with respect to the number of devices. Given the same number of devices, the hierarchical reservoirs have a better dynamics than the monolithic ones.

Figure 8.3 Performance and power consumption of monolithic and hierarchical reservoirs on different tasks. The power measurements are normalized to the power measurements of the monolithic reservoirs for comparisons. For the CIFAR-10 image task (c), the training and testing color images from the dataset were converted into grayscale images to reduce their dimensions. The hierarchical reservoirs, on average, consumed less power compared to monolithic reservoirs for both isolated spoken digit and CIFAR-10 tasks while maintaining comparable performance. For the MNIST task, the hierarchical reservoirs were not able to compete with the monolithic reservoirs.

Figure 8.4 Kernel quality measurements of mixed reservoirs with respect to the percentage of memristive devices. The dynamic response of the hierarchical reservoirs increases when the percentage of the memristive devices is above 50%.
Figure 8.5  Deep memcapacitive reservoir computing architecture. The architecture is organized in a stacked structure of $N$ layers. Each layer is a small-world power-law reservoir. At each time step $t$, the input signal $u(t)$ stimulates the first reservoir layer and the output of each successive layer $x(t)^n$, scaled by a fixed input vector $W^{(n+1)}$, excites a higher one in the stack. The outputs of all layers $\{y_1(t), y_2(t), \ldots, y_N(t)\}$ are collected as the input signal to the Readout node. $W^{\text{res}(n)}$ is the internal weight of a memcapacitive reservoir layer. Only the Readout node is trained. Reprinted from [63].

Figure 8.6  Performance and power consumption of deep mem-device reservoirs for the music note prediction task on two datasets: (a) piano-midi.de and (b) MuseData. In general, deep mem-device reservoirs achieved comparable performance to that of the RNN-RBM, expect the Biolek deep reservoirs.
STATUTORY DECLARATION

I hereby formally declare that I am the sole author of this dissertation and I have not used any source or aid apart from the stated ones. This thesis was neither presented in equal nor in similar form to another examining board at any other university. I cited all used references observing actual academic rules.

In Portland, OR, May 9, 2019

Dat Tran, S. J.
MOTIVATION

Over the last several decades, conventional computing architectures have developed into powerful systems, composed of billions of transistors operating at GHz frequencies connecting in multi-core structures that are capable of large scale parallelism. These architectures opened a new horizon for more compact and powerful electronic devices that transformed the daily life activities in our modern world. With a growing demand for higher speed and lower power consumption, multicore architectures are approaching their limits as described by Moore’s Law. Advanced technology allows more cores per chip, however, power constraints prevent powering all cores at their full speed, which, in turn, undermines the overall speed of the chip [57]. The current multicore architectures and the devices will inevitably reach their physical limitations [58].

While any algorithm can be computed with a von Neumann architecture, the separation between a Central Processing Unit (CPU) and memory units becomes a road block that prevents computing efficiency in performing basic tasks. Since all computations take place in the CPU, information is transferred back and forth between the memory units and the CPU for processing. The limited bandwidth of data paths becomes a bottleneck in which the CPU is often idle, waiting for data to transfer to and from memory units. For a neuromorphic task, such as processing a large data of natural images, the bottleneck prevents the CPU from running at full capacity. Adding cache memory and using
general-purpose Graphic Processing Units (GPUs) do not completely solve the bottleneck problem [84].

Neuromorphic architectures are brain-inspired architectures that mimic the functionality of a biological neural network. In a biological neural network, neurons communicate with each other through a massive number of dendrites and synapses. Computations are decentralized to neurons and each neuron (a processing unit) processes locally the information that is stored in its input synapses (memory units). Decentralizing information and localizing computations at the synapse level resolve the bottleneck problem and allow the system to process a large amount of data in parallel. Moreover, biological neural networks are highly adaptable to complex environments, tolerant to system noise and variations, and capable of processing complex information with extremely low power [159].

Various brain-inspired architectures have been proposed as computational architectures for neuromorphic computing. In 2011, IBM constructed the Watson supercomputer with an evidence-based learning capability [191]. It was composed of 90 of IBM’s Power-750 servers; each server contained a 3.5-GHz Power7 eight-core processor and consumed 0.22 MW of power. IBM’s Blue Gene/P supercomputer was employed to perform biologically sub-cortical simulations of a cat’s brain [77]. The Blue Gene/P supercomputer has 147,456 CPUs with 144 TB (tetra bytes) of memory and consumed up to 2.3MW. The Blue Gene supercomputer simulates up to $10^8$ neurons with an average power consumption of $2.3 \times 10^7 nW/neuron$. Other neuromorphic platforms are also reported in [136]. The Spiking Neural Network Architecture (SpiNNaker) at Manchester University is a parallel, low-power, and neuromorphic computer that has $2 \times 10^4$ neurons and consumes about 75kW on average ($3.8 \times 10^9 nW/neuron$). Stanford University’s Neurogrid, which has 16 neuro-core chips, simulates $10^6$ neurons with 5W of power ($5 \times 10^3 nW/neuron$).

Reconfigurable On Line Learning Spiking neuromorphic processor (ROLLS) is a neu-
Figure 1.1: Power computation per neuron: SpiNNaker ($3.8 \times 10^9 nW$), BrainScaleS ($5 \times 10^6 nW$), Blue Brain ($2.3 \times 10^7 nW$), ROLLS ($1.6 \times 10^4 nW$), Neurogrid ($5 \times 10^3 nW$), TrueNorth ($60 nW$), and Mammal Brain ($2.3 nW$). The ultimate goal of neuromorphic platforms is to achieve a similar power performance as that of a mammalian brain.
Finding new architectures with high power efficiency has become one of the important goals in neuromorphic computing. Software designs have emerged to simulate biological neural networks of a million neurons [16, 17, 101]. These approaches provide models to study the interactions of large and independent neural circuits within a single application. Although these approaches allow savings in area and power compared with conventional architectures, they are still not adequate architectures for biological brains. In addition to finding new hardware architectures, circuits modeling neurons and synapses represent an important effort in neuromorphic computing. New designs of artificial neurons, such as multi-compartment neurons of CMOS-based components and MOSFET synapses, are a step closer to biological neurons with much less power consumption [82, 148, 189, 196]. However, realizing these designs for a large network with high power efficiency, good performance, and high area density is still a challenge [75].

With the emergence of new nanodevices, particularly memristive devices and their successful fabrication in crossbar arrays, designing neuromorphic architectures has taken a new direction. Similar to the functionality of biological synapses, memristive devices, which are capable of \textit{Spike-Time-Dependent Plasticity} (STDP), provide a basis for storing and changing the weights of synaptic connections among neurons [35]. This capability allows in-memory computation, such as parallel vector-matrix multiplication [115], and open up a new door for hybrid designs of CMOS neurons and memristive devices as synaptic weights to further reduce chip area and power consumption [83, 150, 162]. Truong and Min reported that an improved memristor-based crossbar performs a neuromorphic task with 50\% reduction in area and 48\% of power savings compared to CMOS arrays [187]. For the past decades, memristive networks have shown to be power-efficient [81, 103, 104, 178, 212] and were capable of performing complex tasks [29, 127, 156].
Memristive devices, however, are still resistors by their nature and they consume power. As a result, there is a limit to the power savings that memristive networks is able to achieve. Here, memcapacitors is an alternative option. A recent study has shown that memcapacitive networks were able to achieve comparative performance to memristive networks but with much higher energy-efficiency [181].

A memcapacitor is a two-terminal nanodevice and its capacitance varies with applied voltages. Similar to a memristor, a memcapacitor has memory and functions as an artificial synapse. The memcapacitor also exhibits the STDP behavior [142], which describes the learning process and memory in a human brain [125]. The memcapacitor has a small form factor, similar to memristors [131]. Moreover, a memcapacitor is a capacitor, a storage device, that does not consume static energy. The switching or dynamic energy \( E = CV^2/2 \) of a memcapacitor is small due to its small capacitance (nF to pF range). With an input voltage pulse of 0.8V, for instance, the switching energy of a memcapacitor [23] is about 0.1nJ, whereas a memristor [35] consumes close to 30nJ. As a result, a memcapacitor offers a more power-efficient device for neuromorphic architectures.

Recent studies have shown that a memcapacitive response has been found in nano structure of \textit{ITO}/\textit{HfO}_x/\textit{Si} [139], in \textit{Au}/BFO/Pt structure [165], in a nano compound of \textit{Pt}/\textit{HfO}_x/n-indium-gallium-zinc-oxide (IGZO) [203], in a heterostructure of two chemical compounds \textit{MoO}_x/\textit{MoS}_2 and \textit{WO}_x/\textit{WS}_2 sandwiched between two silver electrodes [19], in a mono-layer \textit{MoS}_2 metal-insulator devices [100], in a metal-insulator composite of \textit{Si}_3\textit{N}_4/p—\textit{Si} and \textit{BiFeO}_3 [206], in inorganic polymer layers embedded with graphene sheets [140], in a nanodevice of polyvinyl alcohol/cadmium sulfide [155], and in a hafnium oxide (\textit{HfO}_x) on n-type Si substrate [204]. Mohamed \textit{et al.} discovered that it is possible to construct a memcapacitive device from a memristive metal-oxide composite by adjusting physical parameters, such as the shape factor of the device [131]. The
memcapacitive characteristics of the device solely depend on the behavior shape factor, which is controllable during the fabrication process. From this finding, Mohamed et al. derived a mathematical model that describes the response of a metal-oxide device based on its device state, its capacitive current, and its tunneling current. When the Behavior Shape Factor (BSF) is less than 0.1, the capacitive current becomes dominant and the device operates as a memcapacitor [131]. Biolek et al. designed a SPICE model that describes the correlation between electrical charge $q$ and voltage $V_C$ using dependent voltage-controlled current source [22].

Several memcapacitive synaptic models have been proposed as an artificial synapse. Pershin and Di Ventra demonstrated that a synaptic model, composed of a memcapacitor and two diodes, mimics the behavior of a synapse in an integrate-and-fire network. Depending on the polarity of the two diodes, the synapse manifests an excitatory behavior or an inhibitory response [142]. For an excitatory synapse system, positive pulses (or spikes) are employed as driving signals whereas negative pulses are employed as input signals to an inhibitory network. Flak suggested that a memcapacitor with three CMOS switches behaves as a synapse in a cellular nanoscale network [60]. The correlation of closing and opening switches allows external control signals to set appropriate synaptic strengths during a learning phase. The activation circuit for each cell in a cellular neural network is also composed of two memcapacitors and eight switches that activate an output signal when the weighted sum of input signals exceeds a threshold. Li et al. illustrated that when four identical memcapacitors are connected in a bridge network, they operate as a synapse [113]. Unlike previous models, the synapse can have a positive or negative value.

Reservoir Computing (RC) has been used to explain higher-order cognitive functions and the interaction of short-term memory with other cognitive processes. A study by Rigotti et al. demonstrated that a dynamic system with a short-term memory capacity is
essential in defining the internal brain states of a test agent [152]. Although both traditional Recurrent Neural Networks (RNNs) and RC provide such a dynamic system, RC is preferable due to the simplicity of its training mechanism. Recurrent neural networks require training all weights with a complex algorithm, whereas RC employs a simpler training mechanism of the output layer only. On the system level, RC is able to harness the computing nature of a random network, composed of nonlinear devices such as memcapacitors, in which the precise structure does not need to be controlled. Demis et al. discovered that it is possible for self-assembled memristive nanowire networks to compute [49]. Reservoir computing allows for larger networks and better parameter scaling while still maintaining comparable performances on various tasks [80, 91, 92, 169]. RC has been successful in time series prediction [199], natural language processing [78], and harmonic generations [11], and is considered as a biologically plausible model for neural computing [55]. Moreover, RC with simplified structures is suitable for low-power applications [9].

Although other studies have contributed new insights to RC in various research areas [28, 30, 67, 117], Goudarzi and Teuscher have noted that there remain specific challenges in RC architectures that need to be addressed [70]:

Q1 A random structure is a common architecture for reservoirs. Theoretical studies have shown that a regular structure of reservoirs offers a similar performance to random reservoirs for common tasks. In comparison, a random network is more suitable to an unstructured network of memcapacitive devices in building a reservoir substrate. What is the role of network structure in RC for a task?

Q2 In a physical network, the dynamics and the RC performance are governed by the physical property of a device. Combinations of nonlinear devices enhance the dy-
namics of the reservoirs. From a theoretical perspective, to what degree does a mixture of nonlinear devices improve the performance of reservoirs?

Q3 Modularity is a key to building a hierarchical system of CMOS circuits. Bürger et al. demonstrated that hierarchical reservoirs outperformed single reservoirs for about 20% in certain tasks [30]. Can hierarchical RCs solve complex tasks? What network topologies/hierarchies lead to optimal performance? What is the learning complexity of such a system?

Given the potential of memcapacitive RC, my research aims to address the above questions (Q1, Q2, and Q3) by exploring memcapacitive network dynamics in monolithic and hierarchical reservoirs. Preliminary results have shown that memcapacitive networks are dynamic systems and that monolithic memcapacitive reservoirs are capable of performing tasks [180,181]. However, there are many areas of memcapacitive network dynamics that are unexplored. The following sub-questions serve as a guide to my research to explore those areas in order to address questions Q1 and Q2:

Q1-2.1 The reservoir network topology significantly affects the network dynamics and, therefore, the overall performance of a reservoir. At what network topology does a memcapacitive reservoir achieve a comparable performance to other reservoirs? Does a specific topology reduce power consumption?

Q1-2.2 What is the design space from regular to random networks that lead to high performance and low power reservoirs? What is the network density of regularity or randomness that reservoirs are still able to preserve their performances while minimizing their power consumption?

Q1-2.3 Insufficient or excessive input and output signals reduces the overall performance
of the network. How many input nodes need to be perturbed in order to generate a sufficient dynamic change in a reservoir and how many output nodes need to be observed for a given task?

Q1-2.4 What representation of input signals (spikes, analog, or Boolean) leads to high performance and low power consumption for a memcapacitive reservoir? What is the complexity of the training process with respect to a representation of input signals, such as spiking inputs?

Q1-2.5 Both memristive and memcapacitive devices exhibit nonlinear behaviors. What combination of memristive and memcapacitive devices leads to high performance and low power reservoirs?

Q1-2.6 Defining a suitable task for a particular reservoir topology is a significant step to scale up a network for better performances. Among the common tasks, such as MNIST, CIFAR-10, prediction tasks (e.g., NARMA), temporal control tasks (e.g., Santa Fe trail), or voice recognition, what task do mem-device reservoirs efficiently solve with minimal power consumption?

Q1-2.7 Fading memory is an important attribute of a reservoir. Without applied signals, the internal state of a reservoir will decrease to a reset state due to the state volatility of a physical device. Both the memcapacitive models used in my research do not have a state volatility. How are the models modified to implement the state volatility of a physical device?

Q1-2.8 To what degree do device faults and variations start to affect the performance of memcapacitive reservoirs? Do variations offer a useful computation?
Evidence has shown that hierarchical RC achieved similar performance compared to state-of-the-art deep Markov networks [186] and outperformed monolithic reservoirs in waveform generation tasks [30]. In my research, I analyze the dynamics of hierarchical memcapacitive networks by exploring the following sub-questions with an attempt to address the question Q3:

Q3.1 A hierarchical network of reservoirs can perform better a complex task than a monolithic reservoir. For a given task, what are the possible hierarchical reservoirs? What are cluster size, number of clusters, inter-cluster communications, and intra-cluster communications? How are these hierarchical reservoirs trained and what is the learning complexity of the training process?

Q3.2 If a combination of memristive and memcapacitive devices potentially leads to a more dynamic network, what mixture of memristive and memcapacitive cluster nodes in a hierarchical structure enhances the performance and leads to low power reservoirs?

Q3.3 Are memcapacitive devices employed for other architectures than RC, such as multiple-layer RC or deep learning RC? What is the structural complexity of such hierarchical structures? What is the training process of such systems?

My research offers answers to these questions (Q1, Q2, and Q3) and establishes a more detailed understanding of how RCs utilize the nonlinear computing nature of memcapacitive devices in building neuromorphic hardware to perform tasks. My research work:

(a) studies the network dynamics of memcapacitive reservoirs (Chapters 7.1 and 7.2),
(b) establishes a new understanding of how small-world power-law networks evolve with the change of network parameters and how the topological structure influences the performance of memcapacitive reservoirs (Chapter 7.3),

(c) defines optimizing parameters for memcapacitive RCs as power-sufficient systems to perform specific tasks (Chapter 7.4),

(d) describes the enhancement of dynamic response when different nonlinear devices are combined within reservoirs (Chapter 7.5),

(e) illustrates how input signal representation influences the performance of memcapacitive reservoirs in terms of classification and simulation time (Chapter 7.6),

(f) investigates the impact of device faults and variations on the performance of memdevice reservoirs (Chapter 7.7),

(g) enables a range of applications from image and speech recognition to time-series prediction (Chapter 7.4), and

(h) reviews how hierarchical memcapacitive reservoirs and deep memcapacitive RC overcome the limitations of monolithic reservoirs in solving more complex tasks (Chapters 8.1 and 8.2).
Reservoir Computing (RC) is a machine learning approach that uses transient dynamics of a system (i.e., reservoir) to translate spatiotemporal input signals to a higher dimensional space and to perform a task by means of a linear readout layer [87, 121]. Reservoir computing has been successful in building large networks [169], in time series prediction [199], and in natural language processing [78]. Despite significant progress, as Goudarzi and Teuscher indicated [70], important theoretical and experimental questions still remain unanswered and need to be addressed. The goal of my research is to address the following three specific questions:

Q1 A random structure is a common architecture for reservoirs. Theoretical studies have shown that a regular reservoir structure offers similar performance for particular tasks [51, 76]. In comparison, random network is a more probable platform for reservoirs since random network is easy to fabricate [29, 156]. What is the role of network topology in reservoir computing for a particular task?

Q2 In a physical network, the dynamics and the RC performance are governed by the physical property of a device. Combinations of nonlinear devices enhance the reservoir dynamics. From a theoretical perspective, to what degree does a mixture of nonlinear devices improve the performance of reservoirs?
2.1. NETWORK DYNAMICS IN MEMCAPACITIVE RESERVOIR COMPUTING

Q3 Modularity is a key to building a hierarchical system of CMOS circuits. Bürger et al. demonstrated that hierarchical reservoirs outperform single reservoirs for about 20% in certain tasks [30]. Are hierarchical RCs able to solve a complex task? What network topologies/hierarchies lead to optimal performance? What is the learning complexity of such a system?

Given the potential of saving power in using memcapacitive devices for reservoirs, my research aims to address the above questions (Q1, Q2, and Q3) by exploring memcapacitive network dynamics in monolithic and hierarchical RCs.

2.1 NETWORK DYNAMICS IN MEMCAPACITIVE RESERVOIR COMPUTING

The functionality of RC depends on the nonlinear dynamic behavior and the fading memory of a reservoir. The nonlinear and dynamic behavior allows spatiotemporal information to be translated into a spatial representation in the reservoir. The fading memory, on the other hand, ensures that the memory of inputs will fade away over time. For the past decades, reservoirs have been implemented with software platforms. Recent studies have shown that a memristive hardware implementation of a reservoir is possible [11,166,173]. Memristive reservoirs perform various tasks due to their nonlinear dynamics and memory capability [31,32,106]. Similar to a memristor, a memcapacitor is a nonlinear device that retains its state with an external input. A memcapacitive network, which is a nonlinear and dynamic network, offers a computational substrate for a reservoir. Do memcapacitive networks have sufficient dynamics to address the challenges of the RC questions (Q1 and Q2)?
2.1. NETWORK DYNAMICS IN MEMCAPACITIVE RESERVOIR COMPUTING

Research Topic 1. Memcapacitive Network Dynamics for Reservoir Computing

The network topology is defined by the number of nodes and their connections (memcapacitive devices). Not all random memcapacitive reservoirs perform to the same degree. The level of randomness, network density (number of connections at each node), and device parameters strongly affect the dynamic response of a reservoir. Initial experiments have shown that there is a high correlation between the network topology and its performance [181]. The criteria for selecting optimal topologies for the performance and the power consumption of reservoirs are unexplored. The following sub-questions will address the challenge RC questions Q1 and Q2.

**Question 1.1.** Fading memory is an important attribute of a reservoir. Without applied signals, the internal state of a reservoir will return to a reset state due to the state volatility of the physical devices. Both memcapacitive models in my research do not have a state volatility. What modification(s) do the models need to express the volatile property at the device level and at the system level? See Chapter 7.1 for answer.

**Question 1.2.** Information is transferred from one node to another through memcapacitive links. The network topology affects significantly the network dynamics and, therefore, the overall performance of a reservoir. At what network topology does a memcapacitive reservoir achieve comparable performance to other reservoirs? Does a specific topology reduce the power consumption? See Chapter 7.2 for answer.

**Question 1.3.** Memcapacitive devices function as synapses between nodes within a reservoir. Through the correlations of the synapses in a reservoir, input signals are transformed into a higher dimensional space where a readout layer is used to train
2.1. NETWORK DYNAMICS IN MEMCAPACITIVE RESERVOIR COMPUTING

for a specific task. Expanding the number of synaptic devices and nodes potentially improves the richness of the dimensional space but also increases the power consumption of reservoirs as well. For a neuromorphic task, what is the design space from regular to random networks that lead to high performance and low power reservoirs? What network density of regularity or randomness allows reservoirs to maintain their performances while minimizing their power consumption? See Chapter 7.3 for answer.

**Question 1.4.** Input signals to the input nodes of a reservoir stimulate dynamical changes in a reservoir. Excessive inputs cause an over-stimulated response whereas insufficient inputs lead to an under-stimulated state. The output nodes of a reservoir provide output signals to an output layer where the output layer is trained for a particular task. Inadequate output signals will reduce the overall performance of the network. How many input nodes need to be perturbed in order to generate a sufficient dynamic change in a reservoir and how many output nodes need to be observed for a given task? See Chapter 7.4 for answer.

**Question 1.5.** The ability to translate input signals to a higher dimensional space depends on the nonlinear characteristics of devices within a reservoir. Both memristive and memcapacitive devices exhibit nonlinear behaviors. What combination of memristive and memcapacitive devices leads to high performance and low power reservoirs? See Chapter 7.5 for answer.

**Question 1.6.** The input representation (spikes, analog, or Boolean) also contributes to the power consumption of reservoirs. What representation of input signals leads to high performance and low power consumption for a memcapacitive reservoir? What
is the complexity of the training process with respect to a representation of input signals, such as spiking inputs? See Chapter 7.6 for answer.

**Question 1.7.** Reservoirs are often configured to solve specific tasks. Defining a suitable task for a particular reservoir topology is a significant step toward improving the performance of networks. Among the common tasks such as MNIST, CIFAR-10, prediction tasks (e.g., NARMA), temporal control tasks (e.g., Santa Fe trail), or voice recognition, what class of tasks do mem-device reservoirs efficiently solve with minimal power consumption? See Chapter 7.4 for answer.

**Question 1.8.** It has been shown in [30,31] that memristive reservoirs are tolerant to faults and variations, even in a noisy environment [67,68]. In a similar way, memcapacitive reservoirs have the potential to tolerate faults and variations. To what degree do device faults and variations affect the performance of memcapacitive reservoirs? See Chapter 7.7 for answer.

### 2.2 NETWORK DYNAMICS IN HIERARCHICAL RESERVOIRS

Modularity is a key to a hierarchical structure. Circuits of CMOS logic gates, for instance, provide fundamental blocks in a hierarchical structure to build a larger digital system. Most reservoir networks are monolithic. They are not easily scalable by simply increasing the number of nodes within a reservoir and, therefore, limited in their computational capacities. Is it possible to connect memcapacitive networks in a hierarchical structure in order to expand their computational capacities and to solve more complex tasks (RC question Q3)?


2.2. NETWORK DYNAMICS IN HIERARCHICAL RESERVOIRS

Research Topic 2. Hierarchical Structures for Memcapacitive Reservoirs

**Question 2.1.** It has been shown in [30] that a hierarchical structure of reservoirs performs a complex task better than a monolithic reservoir. The isolation between cluster nodes in a hierarchical reservoir allows more dynamic changes and, therefore, fewer nodes within a cluster are required. For a given task, what are the possible hierarchical reservoirs? What is the cluster size, the number of clusters, the inter-cluster communications, and the intra-cluster communications? What is the complexity of the training algorithm for these hierarchical reservoirs? See Chapter 8.1.3 for answer.

**Question 2.2.** The positions of an input node and an output node within an RC cluster node affect the dynamical behavior of the cluster and how the dynamic information is transferred to another cluster. How are the input and output nodes of an RC cluster chosen without sacrificing the overall performance of a system? See Chapter 8.1.4 for answer.

**Question 2.3.** Combinations of memristive and memcapacitive devices potentially lead to a network with richer dynamics. In a hierarchical structure, what mixture of memristive and memcapacitive cluster nodes in a hierarchical structure enhances the performance and leads to low power reservoirs? See Chapter 8.1.5 for answer.

**Question 2.4.** Are memcapacitive devices employed for other architectures besides RC, such as multiple-layer RC or deep learning RC? What is the structural complexity of such hierarchical structures? What is the training process of such systems? See Chapter 8.2 for answer.
The discovery of a memristive device in 2008 [175] realized Chua’s prediction of the fourth basic circuit element [43]. Chua argued that the correlations of the four fundamental variables – current $i$, voltage $v$, electric charge $q$, and flux-linkage $\phi$ – are capable of generating five known relationships. The first three correlations are expressed in the three basic elements: resistor $R (v = Ri)$, inductor $L (\phi = Li)$, and capacitor $C (q = Cv)$. The last two correlations describe the relationship between the charge $q$ and the current $i$, governed by $q(t) = \int_0^t i(\tau)d\tau$, and the relationship between the flux-linkage $\phi$ and the voltage $v$, dictated by $\phi(t) = \int_{-\infty}^t v(\tau)d\tau$. These two correlations specify the new set of a mem-device family, namely memristor, memcapacitor, and meminductor.

### 3.1 MEMRISTIVE DEVICES

One of the missing correlations, as Chua pointed out, is the relationship between the flux-linkage $\Phi$ and the charge $q$. This relationship suggests a new device, the fourth circuit element called a memristive device (or memristor) $M$. A memristor is a memory device that is characterized as a resistor whose resistance is time-dependent of the current going through it (Figure 3.1a).

Figures 3.1b, 3.1c, and 3.1d show the fundamental characteristics of a memristor. An
3.1. MEMRISTIVE DEVICES

Figure 3.1: Memristor fundamentals. (a) The memistor as the fourth two-terminal circuit element; (b) the nonlinear dependent current through a memristor; (c) the resulting hysteresis loop in the IV plane; (d) the change of device state over time. The device responses (b, c, d) are created from the model presented in [163].

applied signal (a sine wave) causes a current to go through the device. Unlike a fixed resistor, the current correlates nonlinearly to the applied voltage due to the nonlinear change of the device state (Figure 3.1d). In other words, the resistance of the device changes according to the applied voltage. This nonlinear relationship between the current and the voltage results in a pinched hysteresis loop as shown in Figure 3.1c.

Since the discovery of the first memristive device until now, many memristive devices
3.2. MEMCAPACITIVE DEVICES

have been introduced with more robust mathematical models describing physical properties [1, 15, 128, 138, 205]. Scientists have embarked on various studies, looking for new approaches to integrating memristive devices into future computing architectures. For example, a memristive device with nonvolatile property and fast switching time provides a much higher memory density at lower energy costs [71]. Logic gates using memristive devices have better energy efficiency and higher integration densities than CMOS gates [3, 40, 108].

In the area of neuromorphic computing, memristive devices have been prominent candidates for imitating biological synapses [35, 36]. Recent studies have shown that a crossbar network as a two-layer feedforward neural network using memristive devices as artificial synapses is able to perform image processing with energy efficiency and area density [32, 99, 147, 149].

3.2 MEMCAPACITIVE DEVICES

Recent studies indicate that memcapacitive behavior has been found in many new nano-composites: metal-oxide [131], reactive electrode (Mo, Al) and hafnium oxide (HfO_x) on n-type Si substrate [204], nanocomposites of polyvinyl alcohol/cadmium sulfide [155], nanocomposite of Pt/HfO_x/n-IGZO [203], monolayer MoS_2 metal-insulator [100], graphene composite [141], and BiFeO_3-coated nano switch [206]. These studies open a new possibility for memcapacitive applications.

The concept of memory in memristive devices is also generalized to a capacitor and defines a memcapacitive device. The memcapacitive device describes a relationship between electric charge $q$ and applied voltage $v$. This relationship is based on two fundamental equations that express the mathematical relation between voltage $v$, charge $q$, and
current $i$ in a capacitor:

$$i_C = C \frac{dv_c}{dt},$$

$$q = Cv.$$

The memcapacitive device is a memory device whose capacitance changes according to the internal state of the device. In general, the characteristics of a memcapacitive device is described in [22]:

$$q = C (x, V, t) V,$$

$$\frac{dx}{dt} = f (x, V, t),$$

where $q$ is the charge on the device at time $t$, $V$ is the applied voltage, $x$ is the internal state of the device, $C$ is the capacitance, which depends on the internal state $x$, and the function $f()$ describes how the behavior of the internal state $x$ changes.

### 3.2.1 Bipolar Memcapacitive System with Threshold

Biolek et al. derived a generic model of a memcapacitive device from a mathematical model of the threshold voltage of a memristive device [22]. Within this generic model, the capacitance $C$ takes on the role of the internal state variable $x$ and the device state is defined as:

$$q = CV_c,$$

$$\frac{dC}{dt} = f (V_c) W (C, V_c),$$
where \( f() \) is a function characterizing the change of device state and \( W() \) is a window function setting limits to the internal state \( x \). These functions are prescribed as:

\[
f(V_C) = \beta (V_C - 0.5 * \left( |V_C + V_l| + |V_C - V_l| \right)), \tag{3.4a}
\]

\[
W(C, V_C) = \theta (V_C) \cdot \theta (C_{\text{max}} - C) + \theta (-V_C) \cdot \theta (C - C_{\text{min}}). \tag{3.4b}
\]

In Equations 3.4, the step function \( \theta() \) limits the capacitance of the device to the range \([C_{\text{min}}, C_{\text{max}}]\), \( \beta \) is a constant rate of change when \(|V_C|\) is greater than the threshold voltage \( V_t \), and \( C_{\text{min}} \) and \( C_{\text{max}} \) are the minimum and maximum values of the device capacitance.

![Biolek Spice memcapacitive model](image)

**Figure 3.2**: Biolek Spice memcapacitive model. All dependent current and voltage sources are defined as: \( E_q = V(x) V_{\text{in}} \), \( G_x = f(V_{\text{in}}) W(V(x), V_{\text{in}}) \), and \( G_{mc} = ddt(V(q)) \). The capacitance \( C \) is 1F and the resistance \( R \) is 100M\( \Omega \). \( V_{\text{in}} \) is the input voltage to the device. The two functions \( f() \) and \( W() \) are defined in Equations 3.4.

Figure 3.2 depicts a Spice model for the Biolek memcapacitive device. Using dependent current and voltage sources is a common way to implement the state equations of a memcapacitive device. The values of the dependent current and voltage sources follow the correlation of the input voltage and the state variable specified by Equations 3.3 and
3.2. MEMCAPACITIVE DEVICES

3.4.

Figure 3.3: The characteristic of Biolek memcapacitive model [22]. The device was simulated with a sine wave of 0.55V at 50kHz: (a) the device exhibited a hysteresis response of electrical charge Q with applied voltage V and (b) the capacitance of the device varied between $C_{\text{min}}$ (1pF) and $C_{\text{max}}$ (100pF).

Similar to a memristive device, the correlation between the electric charge $q$ and the applied voltage $V_{in}$ follows a pinched hysteresis path, illustrated in Figure 3.3a. The capacitance of the device also varied with the applied voltage $V_{in}$, shown in Figure 3.3b. The memory capability of the device is its capacitance.

3.2.2 Memcapacitive Characteristics of Metal-Oxide Junctions

Mohamed et al. observed that the effect of the growth or the shrinkage of thin filaments exhibits memcapacitive behavior when some physical parameters of a metal-oxide device are adjusted during the fabrication process [131]. A metal-oxide junction device is a common structure that manifests memristive patterns [202]. However, with additional layers of sandwiched materials of high permittivity, the device starts to alter its behavior to a memcapacitive response.
3.2. MEMCAPACITIVE DEVICES

An ionic device displays memristive behavior with a structure of Ag₂S being sandwiched between platinum electrodes as shown in Figure 3.4(a). The behavior changes to a memcapacitive response when a high-K dielectric material is employed [131]. Figure 3.4(b) illustrates the process of filament formation when a potential voltage is applied across the device. The formation process is determined by two factors: the filament gap $x$ and the cross sectional area $m$ of the filament. A metal-oxide device, in general, experiences the effect of two dynamic currents: the tunneling current $i_t(t)$ and the capacitive current $i_c(t)$ [6]. Modeling a metal-oxide device as a memcapacitive device entails three steps of calculations: device state, capacitive current, and tunneling current.

**Device State**

Ion migrations in a metal-oxide device cause a growth or a shrinkage of the filament formed with respect to applied voltages. This process affects simultaneously both the filament length (or gap $x$) and the filament cross sectional area $m$, which determines the
3.2. MEMCAPACITIVE DEVICES

The internal state of the device. The filament gap $x$ is described in [131]:

\[
\frac{dx(t)}{dt} = -K_s \text{stp}[i(t)] \exp\left[ \frac{-E + D_s i(t)}{k_B T} \right] * f_{wx}(x, i, t) + K_g \text{stp}[-i(t)] \exp\left[ \frac{-E - D_g i(t)}{k_B T} \right] * f_{wx}(x, i, t),
\]

where $K_s$ and $D_s$ are shrinkage constants, $K_g$ and $D_g$ are growth constants, $E$ is the activation energy, $k_B$ is the Boltzmann constant, and $T$ is the operating temperature. The tunnel current $i(t)$ describes the electric tunnel effect of electrons moving across potential barriers [167]. This current is a function of applied voltage $v$ and the ion deposits of the filament length $x(t)$ and its cross sectional area $m(t)$, which is explained in [131]. The exponential function window $f_{wx}(x, i, t)$ is introduced to set the boundary conditions on the growth of the filament gap $x$:

\[
f_{wx}(x, i, t) = \delta_x - \theta_x \sqrt{[\text{stp}(-i(t)) - x(t)]^2} \exp \left\{ -P \sqrt{[\text{stp}(i(t)) - x(t)]^2} \right\},
\]

where $\delta_x = x_{max}$ and $\theta_x = x_{max} - x_{min}$.

In addition to the filament gap, the filament cross sectional area also influences the internal state of the device. The cross sectional area $m$ follows a mathematical expression [131]:

\[
\frac{dm(t)}{dt} = B_s \text{stp}[i(t)] \exp\left[ \frac{-E + W_s i(t)}{k_B T} \right] * f_{wm}(m, i, t) + B_g \text{stp}[i(t)] \exp\left[ \frac{-E + W_g i(t)}{k_B T} \right] * f_{wm}(m, i, t),
\]

where $B_s, W_s, B_g$, and $W_g$ are growth and shrinkage coefficients, $f_{wm}(m, i, t)$ is the window function that limits the boundary of the change in the cross sectional area $m(t)$. This
3.2. MEMCAPACITIVE DEVICES

window function is defined:

\[
f_{wm}(m, i, t) = \delta_m - \theta_m \sqrt{\left[stp(i(t)) - m(t)\right]^2} \times \exp\left\{-P \sqrt{\left[stp(-i(t)) - m(t)\right]^2}\right\},
\]

where \(\delta_m = m_{\text{max}}\) and \(\theta_m = m_{\text{max}} - m_{\text{min}}\).

Device Capacitance

The capacitive current of a metal-oxide device is determined by total capacitance \(C_{\text{total}}\). The total capacitance \(C_{\text{total}}\) is composed of capacitance between filaments \(C_{\text{filament}}\), gap capacitance \(C_{\text{gap}}\), and oxide capacitance \(C_{\text{oxide}}\) [131]:

\[
C_{\text{filament}} = \frac{A}{d_1} \varepsilon_{\text{gap}} \frac{m(t)}{x(t)},
\]

\[
C_{\text{gap}} = \frac{A}{d_1} \varepsilon_{\text{gap}} [1 - m(t)],
\]

\[
C_{\text{oxide}} = \frac{A}{d_1} \varepsilon_{\text{oxide}} [1 - m(t)],
\]

\[
C_{\text{tot}} = C_{\text{filament}} + \frac{C_{\text{gap}} C_{\text{oxide}}}{C_{\text{gap}} + C_{\text{oxide}}},
\]

where \(\varepsilon_{\text{gap}}\) is the gap permittivity, \(\varepsilon_{\text{oxide}}\) is the oxide permittivity, and \(d_1\) and \(d_2\) are the gap thickness and the oxide thickness.

Tunneling Current

The tunneling current \(i(t)\) going through the metal-oxide junctions is a function of applied voltage, defined in [167], gap thickness, and barrier heights at the interface of electron and the gap between the metal layers. The detailed equation for the tunneling current \(i(t)\) is described in [131].

Mohamed et al. observed that the correlation of the tunneling current \(i(t)\) and the
3.2. MEMCAPACITIVE DEVICES

Figure 3.5: The characteristic of the Mohamed memcapacitive model [131]. The device was simulated with a sine wave of 1.0V at 1Hz: (a) the device exhibited a hysteresis response of electrical charge $Q$ with applied voltage $V$ and (b) the capacitance of the device varied between $C_{\text{min}}$ (1nF) and $C_{\text{max}}$ (10nF). The constants of the model were modified for a low input signal of 1Hz: $K_G = 0.4775, K_S = 0.6, B_G = 2.75, B_S = 2.75, x_{\text{min}} = 0.4, x_{\text{max}} = 0.9, m_{\text{min}} = 0.1, m_{\text{max}} = 0.9, d_1 = 5 \times 10^{-10},$ and $d_2 = 5 \times 10^{-10}$.

capacitive current $i_c(t)$ determines the response pattern of a metal-oxide device. The behavioral shape factor (BSF) formulates the ratio of root mean square (RMS) currents:

$$BSF = \frac{\text{RMS} \left[ i(t) \right]}{\text{RMS} \left[ i_c(t) \right]}.$$  

The device conducts as a memristive device when BSF is substantially large (e.g., $BSF > 10$). However, when BSF is considerably small such as $BSF < 0.1$, the device manifests itself as a memcapacitor.

Figure 3.5 shows the responses of the metal-oxide device. As shown in Figure 3.5a, the device behaves as a memcapacitive device when $BSF$ is less than 0.1 and its Q-V relationship portrays a pinched hysteresis loop, which is a distinctive feature of mem-devices. Figure 3.5b displays how the capacitance of the device varies between $C_{\text{min}}$ and $C_{\text{max}}$ ($C_{\text{min}} = 1\text{nF}, C_{\text{max}} = 10\text{nF}$). The constants of the model were modified for a low
input signal of 1Hz: $K_G = 0.4775, K_S = 0.6, B_G = 2.75, B_S = 2.75, x_{\text{min}} = 0.4, x_{\text{max}} = 0.9, m_{\text{min}} = 0.1, m_{\text{max}} = 0.9, d_1 = 5 \times 10^{-10}, \text{ and } d_2 = 5 \times 10^{-10}$.

### 3.2.3 Bistable Elastic-Membrane Memcapacitor

![Diagram of bistable elastic-membrane memcapacitor](image)

Figure 3.6: Bi-stable elastic-membrane memcapacitor. The membrane is changed between two stable states (state "0" or state "1") when an external voltage is applied to the plates.

![Graphs of charge-voltage (a) and capacitance-voltage (b) responses](image)

Figure 3.7: Responses of bistable elastic-membrane memcapacitor: (a) Charge-voltage response, (b) capacitance-voltage response. The capacitance of the device fluctuates between 7pF and 20pF with an applied sine wave signal.

It was demonstrated by Pershin et al. that a parallel-plate capacitor with one of its plates replaced by an elastic membrane exhibits the characteristics of a memcapacitor.
3.2. MEMCAPACITIVE DEVICES

When a voltage is applied to the two plates, the elastic membrane is changed to either state "0" or state "1", corresponding to the logic states of a binary signal. The mathematical model that describes the behavior of a voltage-controlled system illustrates a second-order relation between charge \( q(t) \), the capacitance \( C(y) \), and the applied voltage \( V_C(t) \) in terms of these equations [144]:

\[
q(t) = C(y)V_C(t),
\]

\[
\frac{dy}{d\tau} = \dot{y},
\]

\[
\frac{y}{d\tau} = -4\pi^2y\left[\left(\frac{y}{y_0}\right)^2 - 1\right] - \Gamma\dot{y} - \left(\beta(\tau)\right)\left(\frac{1}{1+y}\right),
\]

where \( C(y) = \frac{C_0}{1+y} \), \( y = z/d \), \( z \) is the position of the membrane with the middle position, \( d \) is the separation between the metal plates and the equilibrium position of the membrane, \( y_0 = z_0/d \), \( \Gamma = 2\pi\gamma/\omega_0 \), \( \beta(t) = 2\pi/(\omega_0 d) \sqrt{C_0/(2m)}V_C(t) \), and \( \tau = t\omega_0/(2\pi) \). Furthermore, \( \pm z_0 \) are the equilibrium positions of the membrane, \( \gamma \) is the damping constant, \( \omega_0 \) is the natural frequency of the system, \( m \) is the mass of the membrane, and \( C_0 = \varepsilon_0 S/d \).

3.2.4 Multilayer Memcapacitive System

A multilayer memcapacitive system is composed of \( N \) metal layers embedded between the parallel plates of a regular capacitor, as shown in Figure 3.8a. In this structure, metallic layers are separated by insulators that create a transport tunnel for electric charges between layers [126]. An external voltage causes a distribution of charge between the embedded layers, which forms a tunnel allowing the transport of charge from the external plates.

A multilayer memcapacitive system, in general, is formulated as mathematical ex-
3.2. MEMCAPACITIVE DEVICES

Figure 3.8: Multilayer memcapacitive system. In Figure (a), $\pm q$ are the electric charge of the external plates whereas $Q_i$ signifies the electric charge of an internal metal layer.

expressions [126]:

$$y(t) = g(x, u, t)u(t),$$

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

where $u(t)$ is the input, $y(t)$ is the output, $g()$ is the response of the system, $x$ is a set of state variables describing the internal state, and $f()$ is a continuous n-dimensional function. These mathematical expressions are modelled further as a charge-controlled system as [126]:

$$V_C(t) = C^{-1}(x, q, t) q(t),$$

$$\dot{x} = f(x, q, t),$$

where $q(t)$ is the charge on the capacitor at time $t$, $V_C(t)$ is the applied voltage, and $C$ is
the memcapacitance which depends on the current state of the system. The total internal charge is defined as [126]:

$$\sum_{k=1}^{N} Q_k(t) = 0$$

where \(Q_k(t)\) is the charge on the internal layers at time \(t\).

The external plates cause a uniform electric field to arise in the direction perpendicular to the plates with a magnitude \(E = \sigma / (2\varepsilon_0\varepsilon_r)\), where \(\sigma = q/S\) is the surface charge density, \(\varepsilon_0\) is the vacuum permittivity, and \(\varepsilon_r\) is the relative dielectric constant of the insulator. The voltage applied to the external plates is [126]:

$$V_C = 2dE_q + \delta E_1 + [\delta - 2\delta_1] E_2 + [\delta - 2(\delta_1 + \delta_2)] E_3 + \ldots$$

$$+ [\delta - 2(\delta_1 + \ldots + \delta_{k-1})] E_k + \ldots - \delta E_n$$

where \(E_q = q/(2S\varepsilon_0\varepsilon_r)\) is the electric field generated by the charge \(q\) at the external plate and \(E_k = Q_k/(2S\varepsilon_0\varepsilon_r)\) is the electric field of the charge \(Q_k\) at the \(k^{th}\) embedded metal layer. The voltage across the device is rewritten as follows [126]:

$$V_C = \frac{q}{C_0} \left[ 1 + \Delta \frac{Q_1}{2a} + (\Delta - 2\Delta_1) \frac{Q_2}{2q} + (\Delta - 2\Delta_{k-1}) \frac{Q_k}{2q} \ldots - \Delta \frac{Q_N}{2q} \right],$$

where \(\Delta = \delta/d, \Delta_i = \sum_{j=1}^{i} \delta_j/d\) for \(i = 1, 2, \ldots, N - 1, \Delta_0 = 0,\) and \(C_0 = \varepsilon_0\varepsilon_1 S/d\) is the capacitance of the system without internal metal layers. The capacitance of the whole system is [126]:

$$C = \frac{q}{V_C} = \frac{2C_0}{2 + \sum_{i=1}^{N} (\Delta - 2\Delta_{i-1}) \frac{Q_i}{q}} \quad (3.7)$$

Equation 3.7 underlines that the capacitance is infinite, negative, and positive at some instances in time. Figure 3.8b shows an example of a two-layer memcapacitive SPICE
model. The values of the capacitors are defined as [126]:

\[ C_1 = \frac{\varepsilon_0\varepsilon_1 S}{(d - \delta)/2} = \frac{2C_0}{1 - \Delta}, \]
\[ C_2 = \frac{\varepsilon_0\varepsilon_1 S}{\delta} = \frac{C_0}{\Delta}, \]

The total capacitance and current are [126]:

\[ C = \frac{q}{V_c} = \frac{C_1C_2}{2C_2 + C_1 \frac{Q'}{q}} = \frac{C_0}{1 + \Delta \frac{Q'-q}{a}}, \]
\[ \frac{d(Q' - q)}{dt} = -I_R. \]

Once again, the total capacitance of this two-layer system is positive, negative, or infinite depending on the quantity \( 1 + \Delta \frac{Q'-q}{a} \). This is the distinctive feature of a multilayer memcapacitive system.

![Graphs](a) Q-V Response. (b) C-V Response.

Figure 3.9: Responses of the two-layer memcapacitive system: (a) Charge-voltage response, (b) capacitance-voltage response. The device is not applicable for the function of a synapse due to the infinite capacitance values when the applied voltage is close to 0V.

It is interesting to note that the Q-V response of a system of two layers, shown in Figure 3.9a, follows a pinched path of hysteresis, but the response does not cross the 0V
3.3. ARTIFICIAL NEURAL NETWORKS

reference. Even when the external voltage is removed, the electric charge still remains in the internal metal layers. Figure 3.9b illustrates the variant capacitance of the system. The response verifies that the capacitance of the system, indeed, is positive, negative, or infinite depending on how the electric charge is distributed between layers. Infinite capacitance poses a constraint on a neuromorphic application in which the finite values of memcapacitors are used for the weight updating phase, the essential stage for the training of an image recognition network.

So far, memcapacitors have been only considered for digital system applications, such as low-power crossbar memory [54] and memory computing [184], as well as analog circuits such as a memcapacitor-based oscillator circuit [195] and an adjustable transmission line [143]. However, it is possible that memcapacitors are able to function as synapses for neuromorphic networks [60, 113, 142]. In fact, the results of my previous work have shown that memcapacitors as synapses in a crossbar structure are able to perform a dot product of input signals and synaptic weight matrix [180].

3.3 ARTIFICIAL NEURAL NETWORKS

Artificial Neural Networks (ANNs) are computing systems inspired by the biological neural networks of a human brain. An artificial neural network is a collection of computing units, called artificial neurons (analogous to axons in the human brain) that communicate with each other through connections (or synapses). A signal from one neuron (the presynaptic neuron) is transmitted through synapses to other neurons (postsynaptic neurons) that, in turn, perform a weighted sum operation and propagate the result to other neurons in the network. The weighted sum is determined by synaptic strengths, which change during a learning process.
3.3. ARTIFICIAL NEURAL NETWORKS

Figure 3.10: An example of a $3 \times 4 \times 2$ fully-connected feedforward artificial neural network. The circles are neurons. In general, the network may have many hidden layers depending on the task to be solved.

Typically, neurons are organized in layers, as shown in Fig. 3.10. Neuron layers perform different transformations on input signals as the signals travel from the input layer to the output layer. An artificial neural network may have many hidden layers depending on the problem or task. Neural networks are often trained to perform a specific task. Fig. 3.10 shows an example of a three-layered, fully connected, and feed-forward neural network ($3 \times 4 \times 2$). There are three major training approaches: supervised, unsupervised, and reinforcement learning. In supervised training, the network is presented with input signals and target outputs. Errors are calculated based on the actual and target outputs. The errors are propagated back through the system by adjusting the synapse strengths or weights, which produce the output signals of a neuron. The process, which is called back-propagation [154], is repeated for all training inputs until corrected weights are refined. The back-propagation algorithm is improved over the years to provide a faster convergence with new techniques, such as Momentum Term [86], Variable Learning Rate [193], and Barzilai and Borwein Learning [13].

In unsupervised training, the network is only presented with input signals. Consequently, the network relies on statistical outputs to adjust its weights appropriately. The
most common unsupervised training approach is cluster analysis, in which statistical data are analyzed to find hidden patterns of grouping in the data. Common cluster algorithms include hierarchical clustering [93], k-means clustering [194], Gaussian mixture [21], self-organizing maps [168], and hidden Markov models [174].

In reinforcement learning, the network is trained to react to its environment in such a way that maximizes its cumulative reward. For example, teaching a dog to perform a new trick by a system of reward or punishment is reinforcement training. The dog has to figure out how to do the trick in order to maximize the reward. Many reinforcement learning algorithms have been introduced such as Ant-Q learning [64], a probable approximation of correction [210], and deep Q-Learning with model-based acceleration [74].

3.4 RESERVOIR COMPUTING

The basic implementation of an ANN is a Multilayer Perceptron (MLP), consisting of an input layer, one or more hidden layers, and an output layer. With the gradient descent back-propagation algorithm, MLP has proven its potential in speech and image recognition [2,209]. However, due to their inherent nature of feedforward structure, MLPs suffer a limitation in which they do not retain the previous state of input data within a current context. An alternative approach is a Recurrent Neural Network (RNN), in which recurrent connections are added. On the one hand, recurrent connections allow the previous information to be retained in the network so that the network learns better from both the previous and current data. On the other hand, recurrent connections significantly increase the complexity of a training algorithm since a small change in recurrent connections has a potential to cause a drastic behavior of the network.

To avoid the training of large-scale recurrent networks, Reservoir Computing (RC)
3.4. RESERVOIR COMPUTING

has been proposed as an alternative. Reservoir computing is based on the assumption that input sequences are sufficiently mapped to the dynamic states of RNNs and that training an output layer is able to achieve an adequate performance for a particular task.

Figure 3.11: An example of a reservoir network showing an input layer, the randomly structured reservoir performing the computation, and the readout layer that interprets the internal dynamics of the reservoir. Only links ($W^{out}$) from the reservoir to the readout layer will be trained, not the internal links ($W^{res}$) of the reservoir.

Figure 3.11 shows an example of a network with an input layer, a random structured reservoir functioning as an untrained computational layer, and an output layer that translates the internal dynamics of the reservoir. In general, any untrained computational substrate that has enough dynamics is able to function as a reservoir. A bucket of water, for example, is capable of performing nonlinear computations through the waves on the surface [134]. The input layer provides the input signal $u(t)$ to the reservoir nodes (a reservoir node is a computational unit, such as a sigmoid neuron) using the input matrix weight $W^{in}$ of fixed values. Within the reservoir, each node is connected to another through the fixed
value weight matrix $W^{res}$ and the time-dependent state $x(t)$ of the reservoir is defined as:

$$x(t + 1) = f \left( W^{res} x(t) + W^{in} u(t) \right),$$  \hspace{1cm} (3.8)

where $f$ is the activation function of a reservoir node, such as $tanh$ function. The output $y(t)$ at the output layer is the inner product of the reservoir state $x(t)$ and the output weight matrix $W^{out}$:

$$y(t) = x(t) W^{out}.$$  \hspace{1cm} (3.9)

Since both the matrices $W^{in}$ and $W^{res}$ are fixed, the learning process only takes place by adjusting the output weight matrix $W^{out}$. This is a major advantage of the RC approach over training RNNs. Furthermore, there are no restrictions in the weight matrices ($W^{in}$, $W^{res}$, and $W^{out}$), input stimulus, internal connections, nodes in the reservoir, and outputs. These parameters are set for a specific task.

There are two main approaches in RC: Echo State Networks (ESNs), developed by Jaeger [89], and Liquid State Machines (LSMs), introduced by Maass [121]. ESNs rely on certain algebraic properties of a recurrent neural network that allow the training process to take place in the readout layer only. ESNs accomplish an excellent performance for practical applications within a short training time. On the other hand, LSMs tailor to more sophisticated, realistic, and biological models for neurons and dynamic synaptic connections within a reservoir [122, 135]. For readout layers, LSMs originally could operate with multilayer feedforward neural networks or linear readouts similar to ESNs. Although both ESN and LSM approaches perform equally well, some researchers have found that a large number of synaptic weights was required for LSMs in order to guarantee a reasonable performance, while it is not the case with ESNs [96].

The computational power of an RC network is based on a short-term memory created
3.4. RESERVOIR COMPUTING

by the reservoir [90] and the ability to maintain the temporal information from distinct signals over time [121,133]. These two characteristics are guaranteed by the four essential properties of a reservoir: separation, approximation, fading memory, and spectral radius.

3.4.1 Separation Property

According to Maass et al. [119, 121], the separation property refers to the ability to represent different input signals for the output layer to learn correctly. The separation property ensures that two different input stream signals $u(.)$ and $v(.)$ will cause two significantly different changes $x^{M}_u(t)$ and $x^{M}_v(t)$ in the internal states of the reservoir. However, there is no general rule on how a reservoir implements sufficiently the signal separation. In a later experiment, Maass et al. proposed an empirical method to analyze the separation property of a reservoir for a given task. The degree of separation is based on the rank of a matrix. The rank of a matrix is defined as the number of rows and columns that are linearly independent within a matrix. A $n \times m$ matrix $M$, for instance, maintains the reservoir state vector $n_i$ for different the input $m_i$. A perfect separation is achieved when each state vector $n_j$ is linearly independent of all other state vectors $n$.

3.4.2 Approximation Property

The approximation property describes the capability of a reservoir system to classify the same output from noisy or unseen inputs of the same class. This property is essential to how the output layer approximates any function of the reservoir states to some degrees of accuracy. Similar to the separation property, there are no set rules for determining how well a reservoir approximates the data of the same class inputs. Recent studies have shown that the approximation ability of an ESN reservoir is closely related to the characteristics of a specific application [69, 132].
3.4.3 Fading Memory

Fading memory is what Jaeger called "the continuity property" for the ESN [88]. Fading memory describes the state decay of reservoir states so that similar input streams of finite length are mapped to the same reservoir state. The current states of a reservoir depend on its initial states and the applied input stream. An input stream $u(.)$, for example, perturbs the reservoir and causes a change in the internal reservoir states from an initial state. After a period of being undisturbed, the reservoir state degrades eventually to a default state so that if the same input stream $u(.)$ is applied again, the reservoir will change to the same state. Fading memory ensures that a reservoir approximates similar input streams to a similar degree of its internal state. This is the crucial requirement for the approximation property of a reservoir.

For both LSM and ESN, the implementation of fading memory is determined by the weight matrix initialization. This weight matrix defines the connection strength between reservoir nodes. After a period of being undisturbed, each value of the weight matrix will eventually decay and become a small value. For a memristive reservoir, the fading memory is inherent in the volatile state of devices.

3.4.4 Spectral Radius

Due to the recurrent connections within a reservoir, the signal at a particular time is scaled by connection strengths as it travels from one node to another. If a connection strength is larger than one, the signal is amplified when it goes through the connection. If the connection strength is less than one, the signal weakens as it passes through that connection. Arbitrary values of connection strengths lead to chaotic network responses with invalid outputs. Setting the correct values of connection strengths ensures the proper
response of a reservoir network. The connection strengths are specified by weight matrices.

For LSMs, the criterion for weight matrices is not clearly defined, but the constraints applied to the separation and approximation properties of a reservoir indirectly set the validity of weight matrices [121]. For ESNs, on the other hand, the constraint for weight matrices was put forward by Jaeger [87]. For a reservoir of \( N \) nodes, for instance, there is a possible weight matrix \( M \) of size \( N \times N \) that allows full connectivity within the reservoir, which provides echo states for that reservoir with a maximum spectral radius \( |\lambda_{max}| \) of one [87].

A matrix, by definition, has its Eigenvalues of \( \lambda_i \) where \( i \) is the index of the Eigenvalue. The largest eigenvalue is the spectral radius of that matrix. For a weight matrix \( M \), \( w_{ij} \) is the weight value at row \( i \) and column \( j \), the spectral radius is the products of weights \( w_{ij} \) that forms recurrent loops:

\[
|\lambda_{max}| = \begin{cases} 
\lambda_i < 1 & \text{if } \prod w_{ij} < 1 \\
\lambda_i = 1 & \text{if } \prod w_{ij} = 1 \\
\lambda_i = 1 & \text{if } \prod w_{ij} > 1
\end{cases} \tag{3.10}
\]

As Jaeger indicated, the spectral radius of weight matrices must be less than or equal to one in order to ensure proper responses of an ESN [87].

3.5 TERMINOLOGY

My research is involved with networks of mem-devices used as reservoirs. There are common terms that are referred to throughout the document. An overview of these terms is described below.
Figure 3.12: Neural network and random memcapacitive network. (a) The circles are neurons, \( w_{i,j} \) denotes a synaptic connection of strength \( w \) connecting neuron \( i \) to neuron \( j \); (b) in a memcapacitive network, an electrical node performs a similar operation of weighted sum as a neuron.

**Node**

In a neural network, a node typically refers to an artificial neuron as shown in Fig. 3.12a. A node performs a computation by applying an activation function to the weighted sum of its inputs. The function of a neuron node is similar to a biological neuron in a human brain.

In an electrical network (Fig. 3.12b), in contrast, a node has a different connotation. A node is an electrical point at which electrical components (i.e., resistor, memristor, memcapacitor) are connected to and from. Electrical nodes are able to perform a similar weighted sum operation as a neuron (the detail is described in Figure 5.2b). They are seen as wire connections that define the topology of a network.

**Weight**

A weight is normally associated with the synaptic strength, a scaling factor of an input signal that determines the contribution of that input to a neuron. Weights are referred to
as the scaling factors of inputs \((w_{1,2}, w_{2,1}, w_{3,1}, \ldots, w_{n,m})\). In Fig. 3.12a, for example, \(w_{2,5}\) is a weight attribute connecting neuron 2 to neuron 5.

For a mem-device network, a memristive network or memcapacitive network, weights often refer to a unit-less parameter \(w\), which is the internal state that is associated with the resistance of a memristor or with the capacitance of a memcapacitor. In mem-device networks, a mem-device state has two functions: (a) providing a signal path for the current from one node to another and (b) scaling the voltage across the device for a memristive device or the electric charge for a memcapacitor.

**Weight Update**

In a neural network, weight update refers to a process of updating the individual weight attributes (or synaptic strengths) \(w_{i,j}\) of the network based on a learning rule, such as the back-propagation algorithm, so that the network is able to learn a new pattern over time. With feedback errors from a learner, the weight attributes are adjusted to minimize the error function between the actual and the target outputs.

In mem-device networks, weight updating is a process to update the resistance or the capacitance of memristors or memcapacitors. Based on feedback from a learner the updating process individually adjusts each mem-device with electrical pulses, characterized by the mathematical description of the physical properties of the mem-devices. The updated pulse is device-dependent.

**Network**

A neural network is composed of nodes (neurons) and links (edges). Each link has an attribute \(w_{i,j}\) that defines the strength of a signal going from node \(i\) to node \(j\). Often a neural network is implemented with a software framework in which matrices represent
scaling factors of inputs, outputs, and network states. Computation takes place as input signals travel through the network nodes.

A mem-device network, from an electrical standpoint, is equivalent to a resistive network or a capacitive network. In this electrical network, the currents and the voltages are interrelated as a complex computational system that is based on the physical connections between nodes and mem-device properties.

**Network State**

In a recurrent neural network, a network state is the output result of computations by neurons. These results are the inputs to the next layer or to a readout layer in the case of reservoir computing in order to determine the network response to applied input signals.

In a mem-device network, the network state is determined by two factors: the mem-device state and node voltages. While the dynamic state of mem-devices contributes significantly to the network activity, the dynamic state is not always accessible due to physical constraints. As a result, the node voltages, which are the presentations of the dynamic states, serve as the network state.
4 RELATED WORK

4.1 MEMCAPACITIVE SYNAPSE

Currently, only a few models have been proposed as memcapacitive synapses. I will describe these models as an initial step for designing memcapacitive synapses in RC architectures.

4.1.1 Memcapacitive Synapse in Spiking Network

Pershin and Di Ventra proposed that it is possible to design a synapse composed of a memcapacitor and two diodes, as illustrated in Fig. 4.1 [142]. In the figure, the RC block behaves as an integrate-and-fire neuron. A voltage pulse from an input $N_i$ will charge the integrating capacitor $C$ in proportion to the capacitance $C_i$ of a memcapacitive synapse. Once the charging voltage of the capacitor $C$ exceeds the threshold $N_{out}$, the capacitor discharges a forward pulse and the control switch $S$ resets it to an initial state. The function of the diodes, connecting to the ground, is to discharge $C_1$ to $C_N$ after the firing pulse. Depending on the connections of the diodes and the polarity of input voltages, the memcapacitive synapse behaves as an excitatory synapse or an inhibitory synapse (see the inset in Fig. 4.1). Pershin and Di Ventra also observed that the synapse experiences the Spike-time-dependent Plasticity (STDP) phenomenon under the effect of Long-term
4.1. MEMCAPACITIVE SYNAPSE

Figure 4.1: Memcapacitive synapses in a spiking network. The $RC$ block functions as an integrate-and-fire neuron that produces a spiking signal at $V_{out}$ when the charging voltage on $C$ exceeds a threshold voltage $N_{out}$. Reprinted from [142].

Depression (LTD) and Long-term Potential (LTP) pulses.

4.1.2 Memcapacitive Synapse in Cellular Network

Flak showed that a memcapacitor with three CMOS switches ($pr_{s_i,j}$ and $AB_{i,j}$) mimics the function of a synapse in a cellular nanoscale network [60]. The $AB_{i,j}$ switch controls how the voltage of a synapse contributes to the cell state $V_x$. The signal $pr_{s_i,j}$ sets the programming voltages $V_{pr1}$ and $V_{pr2}$ and along with the control signals $pr_{t1}$ and $pr_{t2}$ adjusts the value of the synapse during learning phase. The baseline voltage is controlled by $th_1$ and $th_2$ from the threshold circuit, composed of two memcapacitors and eight CMOS switches. The simulations resulting from Flak’s study illustrate that the memcapacitor synapse produces a forward pulse when the weighted sum of input signals exceeds the baseline voltage.
4.1. MEMCAPACITIVE SYNAPSE

4.1.3 Bridge Memcapacitive Synapse

Figure 4.2: Memcapacitive synapses and activation threshold in a network cell. The cell state can have negative, zero, or positive values depending on the values of $AB_{i,j}, Vy_{i,j}$, and $C_M$. Reprinted from [60].

Figure 4.3: Memcapacitive bridge synapse. (a) The synaptic weight is programmed with a current source $I_{in}$; (b) the three transistors function as a differential amplifier that converts a voltage difference to an equivalent current. Reprinted from [113].

According to Li et al., when four identical memcapacitors are connected as a bridge network, they work as a synapse [113]. The bridge synapse is shown in Fig. 4.3. The
4.2 HARDWARE RESERVOIR ARCHITECTURES

The synapse is programmed with a current pulse source, illustrated in Fig. 4.3a. The value of the bridge synapse (Fig. 4.3b) is defined by the capacitance-voltage relationship [113]:

\[
V_{out} = V_{AB} = \left( \frac{D_{M_2}}{D_{M_1} + D_{M_2}} - \frac{D_{M_4}}{D_{M_3} + D_{M_4}} \right) V_{in},
\]

\[
\psi = \left( \frac{D_{M_2}}{D_{M_1} + D_{M_2}} - \frac{D_{M_4}}{D_{M_3} + D_{M_4}} \right),
\]

\[
V_{out} = \psi V_{in},
\]

where \(D_{M_1}, D_{M_2}, D_{M_3},\) and \(D_{M_4}\) are the capacitance of memcapacitor and \(\psi\) is the synaptic strength.

When a positive pulse is applied to its input, the forward-biased voltage will increase the capacitance of \(D_{M_2}\) and \(D_{M_3}\) and decrease the capacitance of \(D_{M_1}\) and \(D_{M_4}\). Consequently, \(V_A\) is larger than \(V_B\) and the synapse \(\psi\) has a positive weight. When a negative pulse is applied, the process is reversed and \(V_A\) is smaller than \(V_B\). In this case, the weight of the synapse \(\psi\) is negative.

Having both positive and negative weights is a major advantage of the bridge synapse. Compared to a memristive bridge synapse, the memcapacitive bridge synapse, as Li et al. has shown, exhibits more nonlinear characteristics and consumes much less power.

4.2 HARDWARE RESERVOIR ARCHITECTURES

4.2.1 CMOS-Based Reservoirs

The first hardware implementation of RC, based on the design of Schrauwen et al., was a digital FPGA that implements a Liquid State Machine (LSM) to perform a speech recognition task [158]. The reservoir had 200 spiking neurons with the same threshold
4.2. HARDWARE RESERVOIR ARCHITECTURES

and reset values. The input weights were randomly selected from a set of \{-0.1, 0.1\}. The internal weights were normally distributed and scaled by an optimal spectral radius of 0.1. Each neuron had 12 input connections, 8 from other neurons in the reservoir and 4 from the inputs. The spike outputs were converted to analog values using an exponential low-pass filter and the output layer was trained with a simple linear regression algorithm. The LSM was able to process real-time speech signals using a new serial data processing scheme.

Li et al. introduced a spiking reservoir using the delayed feedback topology [114]. With the novel design of a timing-dependent encoder integrated into the reservoir, the architecture did not need Analog-to-digital Converters (ADCs) and operational amplifiers (Op-Amps). As a result, the new RC design reduced the power consumption by 16× and improved the area density of the CMOS chip.

4.2.2 Memristive Crossbar Reservoirs

Soures et al. proposed neuromorphic hardware using CMOS and memristors as a computing substrate, a neuro-memristive system. With a small form factor of memristive devices, this system is stable and tolerant to noise [170]. The neuro-memristive system utilizes a memristive crossbar as a dynamical platform for a Liquid State Machine (LSM). Fig. 4.4 describes the architecture of a memristive LSM. On the left, the crossbar structure provides the basic structure of the liquid, which is composed of two memristive crossbars: one crossbar represents the synaptic connections from the input buffer to the liquid and the other denotes the synaptic connections from the liquid to the output layer. Memristive synaptic weights are randomly initialized with binary weight distributions and only the synaptic weights of the second crossbar are trained. Excitatory and inhibitory neurons (LIF neurons) respectively represent positive and negative synaptic potentials. External
4.2. HARDWARE RESERVOIR ARCHITECTURES

Figure 4.4: Hardware architecture of a Liquid State Machine (LSM). The left figure illustrates the memristive crossbar and the connections. The right figure shows the data flow between units in an LSM. Reprinted from [170].

stimuli from the input buffer are converted into spike trains as input signals. At the output stage, the analog signals of the operational amplifiers, which carry out the subtraction operations between excitatory and inhibit currents, are converted into digital signals for the training unit. On the right of Fig. 4.4, the block diagrams demonstrate the data flow between different units during the training of the LSM. The LSM, as Soures et al. observed, was able to perform speech recognition tasks with a high tolerance to noise and device variations, even in a case of 15% faulty neurons.

Hassan et al. designed a hardware architecture of an echo state network that takes advantage of memristive double crossbar arrays [76]. Figure 4.5 shows a memristive crossbar architecture of a $2 \times 4 \times 1$ ESN reservoir computing. As shown in Fig. 4.5a, the input synapses $w_{i,r}$ connect the two neurons (blue neurons) in the input layer to the
4.2. HARDWARE RESERVOIR ARCHITECTURES

Figure 4.5: $2 \times 4 \times 1$ ESN reservoir computing. (a) The reservoir is constructed with random connections; (b) the reservoir is implemented with a memristive double arrays for the input layer and the reservoir. The output layer is also a memristive crossbar. Matched colors for layers are presented. Reprinted from [76].

Figure 4.6: Training process for memristive crossbar ESN. The reservoir output is trained using MATLAB code. Reprinted from [76].

four neurons (gray neurons) in the reservoir in a $2 \times 4$ fully-connected structure. The reservoir synapses $w_{r-r}$ are the recurrent connections of the neurons in the reservoir. The output synapses $w_{r-o}$ map the signals to the output layer (green neuron). All synaptic weights ($w_{i-r}$, $w_{r-r}$, and $w_{r-o}$) are initialized with random values between -1 and 1. Only
the output weight $w_{r-o}$ is trained. Each synapse is described as the sum of two memristive conductances, $G^+$ and $G^-$, in a crossbar structure. The dynamical behavior of the reservoir is illustrated through the reservoir state $x(t)$ and the reservoir output $y(t)$ [76]:

$$x(t) = f \left[ w_{i-r}u(t) + w_{r-r}x(t - 1) \right],$$
$$y(t) = g \left[ w_{r-o}x(t) \right],$$

where $u(t)$ is the input signal, $f()$ is the nonlinear activation function of the reservoir unit, and $g()$ the activation function of the output layer.

In Fig. 4.5b, the large crossbar is a double memristive crossbar that implements the input and the recurrent connections. The bottom crossbar is a $2 \times 8$ crossbar that represents synaptic connections from the input layers (2 blue neurons) to the reservoir (4 gray neurons). The top crossbar ($4 \times 8$) and additional connection wires show the recurrent connections between neurons within the reservoir. The $4 \times 2$ crossbar contains the synaptic weights between the reservoir and the output layer (green neuron). The color codes of memristive devices at the crossbar junctions designate three different types of weights (Fig. 4.5a): the input weight $w_{i-r}$, the reservoir weights $w_{r-r}$, and the output weight $w_{r-o}$.

Fig. 4.6 demonstrates the training process of the reservoir. The procedure utilizes MATLAB to initialize synaptic weights, to process the outputs, and to adjust the output weights. Hassan et al. observed that the memristive crossbar ESN performed the Mackey-Glass task with comparative results to those of the software-based ESN. In addition, the memristive crossbar ESN was tolerant to device and process variations.
4.2. HARDWARE RESERVOIR ARCHITECTURES

4.2.3 Memristive Reservoirs

While there are many applications of memristive crossbar architectures [8, 95, 102, 104], there are only a few studies that proposed memristive random networks as reservoirs in reservoir computing architectures [30, 31, 49, 106].

Kulkarni and Teuscher presented a memristive-based reservoir of fewer than 40 nodes that performed a recognition of triangular-square patterns and associative memory tasks [106]. Fig. 4.7 describes the overall architecture of the memristive reservoir. The reservoir, the computing unit, was a memristive random network. Electrical signals provided stimuli to the reservoir through the input layer. Output signals from reservoir nodes were forwarded to the readout layer, which was trained with a Genetic Algorithm (GA). The simulation results illustrate the computational power of unstructured memristive net-
works.

![3x3 mesh reservoir with internal nodes](image)

Figure 4.8: 3x3 mesh reservoir with internal nodes. Reprinted from [31].

Instead of a random network, Bürger et al. proposed a regular mesh network of memristive devices as a reservoir, as shown in Fig. 4.8 [31]. A frequency-modulated signal of 3V was applied to node n2 and output signals were extracted from other nodes. Similar to the approach of Kulkarni, the readout layer was trained with a GA to detect the modulated frequency of the input signal. The simulation results affirmed that regular memristive reservoirs not only performed a detection task but also accepted device variations.

Sillin et al. observed that a physical network of random memristive devices exhibits a high level of dynamic changes, which is suitable for RC [166]. This random network, as Sillin et al. showed, carried out the function of a waveform generator, in which higher harmonics were generated from a 10Hz input signal.
4.3 HIERARCHICAL RESERVOIRS

Reservoirs are dynamical systems and their collective states are used to reconstruct desired outputs. Unlike monolithic reservoirs, whose computing capability are limited to due to signal interdependence and correlated outputs, the uncorrelated clusters within hierarchichal reservoirs enhance dynamic response and, therefore, enable hierarchical reservoirs to solve more complex tasks. Here, I will explore available hierarchical structures and develop basic steps for building hierarchical memcapacitive reservoirs.

4.3.1 Memristive Hierarchical Reservoir

Using the results of simplified networks in [153], Bürger et al. designed a hierarchical reservoir based on a simple-cycle-reservoir topology (Fig. 4.10b). In the hierarchical reservoir, each reservoir node was a cluster of a random memristive network. The readout layer extracted the outputs from cluster nodes and is trained with a linear regression algorithm. The vector $W^{in}$ represented the scaling factors of input signals. The vector $W^{in}$ had
4.3. HIERARCHICAL RESERVOIRS

![Diagram of a random memristive network and a hierarchical reservoir]

(a) Random memristive network.  
(b) Hierarchical reservoir.

Figure 4.10: Memristive and hierarchical reservoirs. (a) Random electrical memristive network; (b) each cluster node is a random memristive network. Reprinted from [30].

A fixed value of $\nu$ with signs chosen according to Bernoulli distributions. The $N \times N$ matrix $W^{res}$ denoted the connections between cluster nodes. The weight values of the matrix $W^{res}$ were randomly assigned using the spectral radius $|\lambda|$, the largest absolute eigenvalue of the weight matrix. At the beginning of each simulation, the memristive devices within cluster nodes were initialized with random values between 0 and 1.

The simulation results confirmed that the hierarchical reservoir outperformed monolithic networks by at least 20% on waveform generation tasks. For the NARMA-10 task, the hierarchical reservoir reduced the error by a factor of $2 \times$, whereas single memristive networks were unable to produce correct results.

4.3.2 Multiple Layer Reservoir Computing

In automatic speech recognition, the Hidden Markov Model (HMM) has been the forefront acoustic model for state-of-the-art systems that uncover the association between the acoustic signals and the phonemes in a spoken language. Deep Belief Network [130] and Long Short-Term Memory recurrent networks [73], for example, have gained great
successes in phoneme recognition, but their training process has been extremely complex. Triefenbach et al. overcame the training complexity of those networks by introducing multiple-layer reservoir computing, shown in Fig. 4.11. In this architecture, each reservoir was an echo state network of leaky integrator neurons and its readout layer was trained with a ridge regression technique. The pre-processing unit converted an input speech waveform into a sequence of acoustic feature vectors. The feature vectors, in turn, became the inputs to Reservoir 1. The output results of Reservoir 1 were forwarded to the next reservoir in the pipeline. The decoding unit, implemented with a bigram phoneme language model, performed a Viterbi search to reconstruct a phonemic sequence based on the result outputs of the last reservoir. The multiple-layer reservoir, as the authors concluded, achieved comparable performance to that of state-of-the-art systems [73, 129, 130, 160].

4.3.3 Deep Reservoir Computing

With efficient learning in temporal domains, RC has provided a suitable platform for a deep Echo State Network (deepESN) [62, 63]. A deep reservoir of ESNs, or a deepESN, was composed of a stack of multiple nonlinear ESN reservoir layers (Fig. 4.12). The deep reservoir exploited the advantages of a hierarchical temporal feature representation at different levels of abstraction while preserving the training efficiency of RC technique.

Figure 4.11: Multiple-layer reservoir computing. Reprinted from [186].
4.3. HIERARCHICAL RESERVOIRS

Figure 4.12: Architecture of deepESN with \( N_L \) layers. \( u(t) \) is the input signal, \( W_{in}^i \) is the input-to-reservoir weight matrix for layer \( i \), \( \hat{W}^{(i)} \) is the recurrent reservoir weight matrix for layer \( i \), \( x^{(i)}(t) \) is the state of the reservoir of layer \( i \), and \( \theta^{(i)} \) is the weight vector associated to the unitary input bias for layer \( i \). Reprinted from [62].

The training process of a deep reservoir was similar to the training of a single ESN with the exception that the readout took into account the state of reservoirs at all layers.

Through the analyses of state dynamics within each layer, Gallicchio et al. established criteria to design a stable deep reservoir. Gallicchio et al. also observed that a higher layer in the stack experienced fewer temporal dynamics compared to a lower one and contributed less to the dynamics of an overall system. This phenomenon provided a measure to limit the number of layers within a deep reservoir.
As Maass pointed out, reservoir computing architecture is one of the neural-inspired architectures that portrays the cortical model of a biological brain [121]. A dynamic system, such as a reservoir in the RC architecture, with short-term memory capacity, is essential in describing the internal brain states [152]. In this chapter, I propose the architecture of novel brain-inspired and energy-efficient computing systems that are based on memcapacitive devices using the RC approach.

5.1 MEMCAPACITIVE RESERVOIR

Figure 5.1 illustrates the architecture of a memcapacitive reservoir, which shows memcapacitors randomly connected through nanowires. The input signal \( u(t) \) is scaled by the input matrix \( W^{in} \) whose values are randomly selected from the set of \([-\nu, +\nu]\). The time-dependent state \( x(t) \) of the reservoir is determined by the internal capacitance of memcapacitive devices at time \( t \). The internal capacitance of memcapacitive devices are presented as the state matrix \( W^{res}(t) \):

\[
W^{res}(t) = [mc_1(t), mc_2(t), mc_3(t), \ldots, mc_n(t)].
\]
5.1. MEMCAPACITIVE RESERVOIR

Figure 5.1: A memcapacitive reservoir network. The reservoir is an electrical network in which memcapacitors are randomly connected through nanowires. The node voltages are the time-dependent state $x(t)$ of the reservoir that provide signals to the readout layer. $W^{res}(t)$ is the state matrix containing capacitance of memcapacitive devices at time $t$. Only the readout layer is trained by a simple algorithm.

The reservoir state $x(t)$ is defined as:

$$x(t + 1) = f \left[ W^{res}(t)x(t) + W^{in}u(t) \right],$$

where $f$ is the transforming function of a reservoir node. The output $y(t)$ at the output layer is the inner product of the reservoir state $x(t)$ and the output weight matrix $W^{out}$:

$$y(t) = x(t)W^{out}$$

In Fig. 5.1, the reservoir is an electrical network of memcapacitors. The time-dependent state $x(t)$ of the reservoir is electrically determined by the combination of the device capacitance as the reservoir undergoes a disturbance caused by an input signal $u(t)$. 
Unlike a node in a traditional artificial neural network, which is an artificial neuron, a node in the memcapacitive reservoir is an electrical node. Its voltage has a similar operation as that found in the artificial neuron.

\[ y_k = \varphi \left( \sum_{j=1}^{m} w_{kj}x_j + \theta_k \right), \]  \hspace{1cm} (5.1)

where \( \theta_k \) is a bias offset. A common function such as a sigmoidal, hyperbolic, logistic, or exponential function is often used as an activation function for the artificial neuron. An electrical node \( k \), shown in Fig. 5.2b, is able to perform an operation similar to the operation of an artificial neuron. Here, the capacitance of memcapacitive devices plays...
the same role as a synapse. The sum of the total charge at node $k$ is:

$$q_{MC_{L1}} + q_{MC_{L2}} + q_{MC_{L3}} + \cdots + q_{MC_{Lm}} = 0$$

$$(v_1 - v_k)c_{k1} + (v_2 - v_k)c_{k2} + (v_3 - v_k)c_{k3} + \cdots + (v_m - v_k)c_{km} = 0$$

$$v_k(c_{k1} + c_{k2} + c_{k3} + \cdots + c_{km}) = v_1c_{k1} + v_2c_{k2} + v_3c_{k3} + \cdots + v_mc_{km}$$

$$v_k = \left( \frac{1}{c_{k1} + c_{k2} + c_{k3} + \cdots + c_{km}} \right) \sum_{j=1}^{m} c_{kj}v_j. \tag{5.2}$$

By rearranging Eq. 5.2, we obtain:

$$v_k = \varphi \left( \sum_{j=1}^{m} c_{kj}v_j \right) \tag{5.3}$$

$$\varphi(x) = \frac{x}{\sum_{j=1}^{m} c_{kj}} \tag{5.4}$$

The total charge at node $k$ is the weighted sum of input voltages and capacitive synapses (Eq. 5.3). The activation function $\varphi(x)$ converts the total charge into a voltage $v_k$ as specified in Eq. 5.4. The voltage at the electrical node $k$, in a certain sense, is similar to the output of an artificial neuron (Eq. 5.1).

Similar to a traditional neural network, in which artificial neurons are connected through synapses to form a network, memcapacitive devices are connected to form a reservoir network. Studies on various network topologies have shown that the structure of a network influences the dynamic response and the performance of reservoirs [29, 51, 76, 153, 156].
5.2 SMALL-WORLD NETWORKS

In general, any system with enough dynamics is able to serve as a reservoir [52, 134, 156]. So far, random networks have been a probable platform for reservoirs due to the simplicity in the fabrication process [29, 156]. However, Regular networks such as cross-bar networks also have had sufficient dynamics to function as reservoirs to perform a simple task [51, 76].

Although a random structure offers a potential computing substrate in the RC architecture [181], it does not reflect the real nature of physical [197] and biological neural networks [12]. Real networks, such as online social networks, cultural networks, and the internet, often share a common feature, which is called Small-world (SW) property, involving a degree of interconnection at both local and global level [145].

Various studies have shown that the structure of SW networks describes the neural connections and the inter-communication lines between different parts of a biological brain [41, 53, 118]. The SW structure explains the connectivity of nervous systems in which the combination of highly clustering short connections and a few long connections enables information to travel through local neighborhoods and an entire network [5]. The data from Magnetic Resonance Imaging (MRI) scans of healthy and diseased brain states, for instance, show that the SW structure of the brain provides a high resilience to overcome neural disease as a result of new and long connections replacing short and damaged connections [41]. SW networks, as neuroscience suggests, allow a biological brain to balance competing constraints in order to minimize the cost of information processing and to maximize the capacity for growth and adaptation [12]. SW networks closely resemble the topological structure of a brain-inspired computing network.
5.3 SMALL-WORLD POWER-LAW NETWORKS

Figure 5.3: Process of creating a small-world power-law network from a regular network. The regular grid network (a), which has an uniform probability distribution between nodes (c), is transformed into a small-power network (b) by rewiring local connections with global links under the power-law probability distribution (d).

Memcapacitive reservoirs are the Small-World Power-law (SWPL) networks, which is a subset of the SW networks. Both the SW and SWPL networks are similar in the sense that they describe the fundamental characteristics of a small-world phenomenon. The main difference, however, is the formation of a network: an SW network arises from a regular ring network whereas an SWPL network is formed from a grid network, shown in Fig. 5.3. In the original SW model [197], nodes in a regular network have a uniform probability distribution to connect to any other nodes within the network. The uniform
5.3. SMALL-WORLD POWER-LAW NETWORKS

distribution, as Petermann pointed out, is not realistic since each connection, local or global, is associated with a cost, such as wire connection, to the system, whose resources are often limited [145].

![Figure 5.4: SW power-law networks deriving from the same 8x4 grid network with different network parameters. The parameters for each network: (a) $\alpha = 0.4, \beta = 0.8, \gamma = 20, \delta = 2$; (b) $\alpha = 1.2, \beta = 0.56, \gamma = 10, \delta = 4$; (c) $\alpha = 1.7, \beta = 0.22, \gamma = 17, \delta = 1$; and (d) $\alpha = 2, \beta = 0.9, \gamma = 8, \delta = 2$.](image)

In the SWPL model, the decay power law, $q(l) = l^{-\alpha}$, governs the formation of a connection. According to the decay power law, distant nodes have a smaller probability to form connections compared to nearby nodes for the reason that long or global connections require more wire lengths than short or local ones. The formation of an SWPL network
reflects the growth of real networks, including biological neural networks, which tend to have more short connections than long connections due to their limited resources [12, 171].

Figure 5.3 illustrates a process of constructing a small-world power-law network from a grid network. The rewiring probability (or randomness factor), $\beta$, determines the probability of replacing local links with global links. The random factor $\beta$ allows a continuous interpolation between a fully regular ($\beta = 0.0$) to a completely random ($\beta = 1.0$) topology. The decay power-law distribution function, $p(l)$, governs connection probability between nodes with distance $l$. The decay exponent or the locality $\alpha$ describes whether it is a highly-local ($\alpha$ is large) or highly-global ($\alpha = 0$) connection network. It has been shown that $\alpha < D + 1$ ($D$ is the dimension of the regular network) is a necessary condition for the emergence of an SWPL network during the formation process [145]. In addition, we modified the SWPL model by adding two extra parameters for flexibility: the percentage of removed links $\gamma$ and the number of added links $\delta$. These extra parameters allow deriving SWPL networks with optimal power consumption. Depending on various settings of the network parameters ($D, \alpha, \beta, \delta$, and $\gamma$), different small-world power-law structures will emerge (Fig. 5.4).
METHODOLOGY

This chapter will describe methods in how to modify memcapacitive models to exhibit state decay property, to measure the property of reservoirs, to find optimal settings of reservoir parameters, to verify the performance of reservoirs with common benchmarks, to select baseline networks for comparison, and to model nanowire connections for power calculations.

6.1 VOLATILE MEMCAPACITIVE DEVICES

In general, RC requires a reservoir, a dynamic system, that maps an input signal to a higher dimensional space by means of nonlinear transformations. It is important that the reservoir in an RC system preserves its fading memory, or its short-term memory, which is described in Section 3.4.3. An RC system with short-term memory is able to approximate the same class of inputs to a similar degree of reservoir states. The short-term memory is attributed to the volatile property of devices.

6.1.1 Modification of Memcapacitive Biolek Model

Currently, both memcapacitive models in my research, the Biolek [23] and Mohamed [131] models, describe the property of non-volatile devices. I modified the models to
Figure 6.1: State volatility with the added decay function to the original Biolek model [23]. At $t > 40\mu s$, without an applied signal, the state of the device will decline gradually over time.
express state volatility by adding a decay term \( f_d() \) to state equations. With the decay term for the Biolek model, the new state equation becomes:

\[
\frac{dx}{dt} = f(V_C) W(x, V_C) - f_d(x),
\]

where \( f() \) is a function characterizing the change of device state and \( W() \) is a window function setting limits on the internal state \( x \). These functions are defined as:

\[
\begin{align*}
f_d(x) &= \frac{a(x - x_{\text{min}})}{b + cx}, \\
f(V_C) &= \beta (V_C - 0.5 \ast |V_C + V_t| + |V_C - V_t|), \\
W(x, V_C) &= \theta(V_C) \ast \theta(x_{\text{max}} - x) + \theta(-V_C) \ast \theta(x - x_{\text{min}}).
\end{align*}
\]

In Equations 6.1, \( f_d(x) \) is the decay function for \( x > 0 \), \( \theta() \) is the step function that limits the capacitance of the device to a range of \([x_{\text{min}}, x_{\text{max}}]\), \( \beta \) is a constant rate of change when \( |V_C| \) exceeds the threshold voltage \( V_t \), and \( x_{\text{min}} \) and \( x_{\text{max}} \) are the minimum and maximum capacitance values of the device. The setting of the constants \( a, b, \) and \( c \) in the decay function (Equation 6.1a) determines the decay rate of the device state.

Figure 6.1 demonstrates the state volatility with the decay function \( f_d() \). Positive input pulses increased the internal state of the device whereas negative input pulses decreased the state. Without an applied signal, the internal state gradually returned to a reset state.

6.1.2 Modification of Memcapacitive Mohamed Model

The Mohamed model [131] has two state variables that characterize the memcapacitive behavior of a physical metal-dioxide device: the filament gap \( x(t) \) between metal layers and the cross sectional area of the filament growth \( m(t) \). Similar to the approach
to the Biolek model, the state decay functions \( f_{gx}() \) and \( f_{md}() \), as extra terms, were added to the state equations of \( x(t) \) and \( m(t) \), respectively. The filament gap \( x(t) \) and the cross sectional area \( m(t) \) are reciprocal in their response: \( x(t) \) increases while \( m(t) \) decreases and vice versa. The new state equation for the filament gap is defined as follows:

\[
\frac{dx}{dt} = g_x(i_t, t)h_x(x, i_t, t) + f_{gx}(x),
\]

where \( g_x(i_t, t) \) is the function that describes the behavior of the filament gap \( x \), the window function \( h_x(x, i_t, t) \) limits the filament gap in the interval \([x_{min}, x_{max}]\), and \( f_{gx}(x) \) is the decay function. These functions are defined as:

\[
f_{gx}(x) = a_x(x_{max} - x) + b_x + c_x x,
\]

\[
g_x(i_t, t) = -K_s \text{stp}[i_t(t)] \exp \left[ \frac{-E + D_s i_t(t)}{k_B T} \right] + K_s \text{stp}[-i_t(t)] \exp \left[ \frac{-E - D_s i_t(t)}{k_B T} \right],
\]

\[
h_x(x, i_t, t) = \delta_x - \vartheta_x \sqrt{[\text{stp}(-i_t(t)) - x(t)]^2} \ast \exp \left\{ -P \sqrt{[\text{stp}(i_t(t)) - x(t)]^2} \right\}.
\]

In Equation 6.2, \( a_x, b_x, \) and \( c_x \) are the growth coefficients determining how the filament gap between two metal layers \( x(t) \) increases over time. Other constants are explained in Section 3.2.2.

Similarly a decay function \( f_{dm}() \) was added to the cross sectional area \( m(t) \) and the new state equation becomes:

\[
\frac{dm}{dt} = g_m(i_t, t)h_m(m, i_t, t) - f_{dm}(m),
\]

where \( g_m(i_t, t) \) and \( h_m(m, i_t, t) \) are the behavioral and windows functions that characterize
the response of the filament and set the growth limits, and $f_{dm}(m)$ is the decay function. These functions are defined as follows:

$$f_{dm}(m) = \frac{a_m (m - m_{min})}{b_m + c_mm}, \quad \text{(6.3)}$$

$$g_m (i, t) = B_s \exp \left[ \frac{-E + w_s i(t)}{k_BT} \right] * f_{wm}(m, i, t)$$

$$+ B_g \exp \left[ \frac{-E + w_g i(t)}{k_BT} \right],$$

$$h_m (m, i, t) = \delta_m - \vartheta_m \exp \left[ stp(i(t)) - m(t) \right] * \exp \left\{ -P \exp \left[ \frac{stp(-i(t)) - m(t)}{k_BT} \right] \right\}. \quad \text{(6.3)}$$

In Equation 6.3, $a_m$, $b_m$, and $c_m$ are the decay constants determining the decay rate of the cross sectional area filament $m(t)$ over time. Other coefficients are described in Section 3.2.2.

Figure 6.2: State volatility of the modified Mohamed model. (a) Without an applied signal, when $t > 1.7s$, the filament declines steadily over time and the filament gap $x(t)$ returns to $x_{max}$; (b) Without any stimulus, the cross sectional area $m(t)$ of the filament declines over time.

Figure 6.2a portrays the filament gap $x(t)$ with the added growth function. Positive pulses stimulated the growth of the filament and, therefore, decreased the filament gap.
Negative pulses, on the other hand, removed ion deposits on the filament and increased the filament gap. Without an applied pulse, the filament steadily declined over time and the filament gap $x(t)$ returned to $x_{\text{max}}$. The volatility of the cross sectional area $m(t)$ is shown in Figure 6.2b. When the input signal was off ($t > 1.7\, s$), the cross sectional area $m(t)$ gradually decayed to a reset value $m_{\text{min}}$.

![Figure 6.3: Volatile device capacitance.](image)

For the Mohamed model, the dynamic change in both the filament gap $x(t)$ and its cross sectional area $m(t)$ constitutes the internal capacitance of the device, which follows the mathematical relationships:

$$C_{\text{filament}} = \frac{A}{d_1} \varepsilon_{\text{gap}} \frac{m(t)}{x(t)}, \quad C_{\text{gap}} = \frac{A}{d_1} \varepsilon_{\text{gap}} [1 - m(t)],$$

$$C_{\text{oxide}} = \frac{A}{d_1} \varepsilon_{\text{oxide}} [1 - m(t)], \quad C_{\text{tot}} = C_{\text{filament}} + \frac{C_{\text{gap}} C_{\text{oxide}}}{C_{\text{gap}} + C_{\text{oxide}}},$$

where $\varepsilon_{\text{gap}}$ is the gap permittivity, $\varepsilon_{\text{oxide}}$ is the oxide permittivity, $A$ is the cross section...
area of metal layers, and \( d_1 \) and \( d_2 \) are the gap thickness and the oxide thickness.

The plot of the device capacitance in Figure 6.3 shows that the changes in the filament gap \( x(t) \) and its cross sectional area \( m(t) \) contributed to the device capacitance. Positive input pulses increased the capacitance whereas the device decreased its capacitance with negative pulses. Without applied pulses, the capacitance of the device capacitance gradually returned to a reset value.

### 6.1.3 Summary of Device Models

Although various memristive models are introduced in the current literature, only two memristive models, Chang and Oblea, were chosen due to their stability and robustness when they are connected in large-scale networks. The simulation results from the memristive devices were used as references to evaluate the results from the memcapacitive devices. The physical characteristics of devices are shown in Table 6.1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Device Type</th>
<th>Volatile</th>
<th>Values at ( W )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek [22] †</td>
<td>memcapacitor</td>
<td>Yes</td>
<td>( 100 , \text{pF} )</td>
</tr>
<tr>
<td>Mohamed [131] †</td>
<td>memcapacitor</td>
<td>Yes</td>
<td>( 6.5 , \text{nF} )</td>
</tr>
<tr>
<td>Chang [94]</td>
<td>memristor</td>
<td>Yes</td>
<td>( 11 , \text{k\Omega} )</td>
</tr>
<tr>
<td>Oblea [138]</td>
<td>memristor</td>
<td>No</td>
<td>( 12 , \text{k\Omega} )</td>
</tr>
</tbody>
</table>

† These models describe the behavior of non-volatile devices and were modified to exhibit a state volatility, the essential property in reservoir computing.
* This model is not based on a physical device.

### 6.2 RESERVOIR PROPERTY MEASUREMENTS

The computational capability of a reservoir is based on two essential properties: the fading memory and the separation property [87, 121].
6.2. RESERVOIR PROPERTY MEASUREMENTS

6.2.1 Fading Memory

Fading memory allows the reservoir to map similar input signals of a finite length to similar internal states, independent of its initial states. For a mem-device reservoir, the fading effect depends on the internal decay state of each device that contributes to the fading memory of the reservoir as a whole system. Both the Biolek and the Mohamed models were modified with specific decay functions to express the state volatility (Sections 6.1.1 and 6.1.2). Here, I represent a general state equation for both models that has a decay term $f_d(x)$ as a negative quantity:

$$\frac{dx}{dt} = f(x, v, t) + f_d(x),$$  \hspace{1cm} (6.4)

where $v$ is the applied voltage, $x$ is the internal state of the device, and the behavior function $f()$ determines the change of the internal state $x$.

6.2.2 Separation

The separation property describes the ability of a reservoir to separate different input signals through its internal states. The separation property ensures that two different input signals $u(t)$ and $v(t)$, for example, are mapped onto two distinctive internal states $x_u(t)$ and $x_v(t)$ of a reservoir so that an output layer is able to classify them correctly. Previous studies have shown that computational performance was improved with a higher measurement of the separation property for a binary echo state network when the average connectivity (in-degree) of the network is small [33]. In my experiments, I used the approach by Gibbons in measuring the separation property of memcapacitive reservoirs. Gibbons proposed that the separation property of a reservoir is determined by measuring the average Euclidean distance between its internal states [66]. The input signals from similar classes
produce a small Euclidean distance whereas the input signals from different classes generate a larger Euclidean distance. Input signals from the same class generate a $K$ set of reservoir states $(x_1, x_2, x_3, \cdots, x_k)$. The center mean $\mu()$ of a particular class $j$ at time $t$ is as [66]:

$$
\mu_j(t) = \frac{\sum_{i=1}^{K} x_i}{K}.
$$

The separation measurement of a reservoir, $Sep_x(t)$, is based on the different class distance $C_d(t)$ and the similar class variance $C_v(t)$ [66]:

$$
C_d(t) = \frac{1}{N^2} \sum_{m=1}^{N} \sum_{n=1}^{N} ||\mu_m(t) - \mu_n(t)||
$$

$$
C_v(t) = \frac{1}{N} \sum_{m=1}^{N} \sum_{i=1}^{K} ||\mu_m(t) - x_i(t)||
$$

$$
Sep_x(t) = \frac{C_d(t)}{C_v(t) + 1},
$$

where $N$ is the number of different classes.

6.2.3 Kernel Quality

The kernel quality, which was introduced in [112], measures the linear separation property of a reservoir. The linear separation describes how a reservoir disassociates different input patterns, independent of a target output. Similar to the separation metric, the kernel quality is based on a set of $n$ reservoir states generated by $m$ different input stimuli. All reservoir states are collected into a matrix $M$ with the size $n \times m$. In this matrix $M$, each column is a state vector corresponding to a particular input. The rank $r$ of this state matrix $M$ is the measurement of the reservoir’s computational power. A reservoir has the highest computational power when the matrix $M$ has a maximum rank.

---

74
or \( r = m \). In this case, each column in the matrix \( M \) is not a linear combination of other columns. In general, a reservoir with a higher rank matrix of its reservoir states has a better kernel quality for separating input patterns.

### 6.2.4 Lyapunov Exponent

The Lyapunov exponent estimates the dynamic state of a reservoir. If the Lyapunov exponent is positive, the system experiences a chaotic behavior where small changes in input signals result in large changes in the reservoir states. A negative value of the Lyapunov exponent, on the other hand, leads to a stable state network where the system converges onto an attractor in responding to input stimuli. It has been shown that optimal performance occurred near the edge of the reservoir’s chaotic state [18, 42, 112]. The Lyapunov exponent was calculated using the equation defined in the work by Gibbons [66]:

\[
\lambda(t) = k \sum_{n=1}^{N} \ln \left( \frac{\|x_j(t) - x_j(t)\|}{\|u_j(t) - u_j(t)\|} \right),
\]

where \( N \) is the number of input signals, \( u_j(t) \) and \( x_j(t) \) are the input vector and the reservoir state respectively, \( u_j(t) \) is the nearest neighbor to \( u_j(t) \) that causes a disturbance state \( x_j(t) \), and \( k \) is a scaling constant. In my experiments, \( k \) was set to 1 for simplifying calculations, similar to the approach in [105].

### 6.3 GENETIC ALGORITHM

Finding optimal structures of reservoirs that yield the highest performance with the lowest power consumption involves an iterative process of fine-tuning reservoir parameters, such as \( \alpha, \beta, \delta, \) and \( \gamma \). This process, which is time-consuming and task-dependent, is extremely difficult for a manual process. Here, I use the particle swarm algorithm to carry
out the optimizing task. Particle swarm algorithm is robust and capable of addressing efficiently various large-scale science and engineering optimization problems compared to traditional genetic technique [38]. Particle swarm algorithm, which is a variant of a swarm intelligence approach that describes the collective behavior of natural and biological systems such as ant colonies, bird flocking, animal herding, or bacterial growth, is an optimization technique to find a global maximum as a point or surface in an n-dimensional space [146]. In my experiments, the global maximum is the point where reservoirs achieve optimal performance in terms of high classification correction and low power consumption. The particle swarm algorithm randomly selects reservoir parameters within range limits and refines them until the global maximum is found. The range limits are based on experimental results. Due to the stochastic nature in forming the network structures, fifty instances of reservoirs with similar parameters are simulated to obtain average results for each iteration.

6.4 BENCHMARKS AND VALIDATION

6.4.1 NARMA-10 as a Time-series Task

*Nonlinear Auto-Regressive Moving Average* (NARMA) is a discrete-time temporal task with a 10th order. NARMA is often used to evaluate the memory capacity and the computational power of recurrent networks [200, 201]. The NARMA task is also a common measure to quantify the dynamical response of an RC network with various reservoir implementations [10, 30, 46]. The NARMA-10 time series is given by:

\[ y(t) = \alpha y(t - 1) + \beta y(t - 1) \sum_{i=1}^{n} y(t - 1) + \gamma u(t - n)u(t - 1) + \delta, \]
where $n = 10, \alpha = 0.3, \beta = 0.05, \gamma = 1.5,$ and $\delta = 0.1$. The input $u(t)$ is taken from a uniform distribution in the interval $[0.0, 0.5]$. The NARMA task is a challenge to any computational system due to the nonlinear dependence of input lagging time. Calculating the task is trivial if a system is capable of retaining the memory of both the inputs and outputs of up to 10 previous time steps. Three different types of reservoirs - the state-of-the-art ESN, memristive, and memcapacitive reservoirs - were set up to perform the NARMA task and their performances were compared.

**6.4.2 Isolated Spoken Digits as a Temporal Task**

The spoken digit recognition is widely used as a benchmark for reservoir computing systems [109, 177]. The benchmark TI-46 digit corpus is often used for isolated-word automatic speech recognition. Due to the license issue of using TI-46 digit corpus, I utilize a non-license digit dataset to verify the functionality of reservoirs. This dataset, created by Jackson [85], contains the recordings of spoken digits in wave file formats at 8kHz. The silences at the beginning and end of each digit recording were trimmed. The dataset has 1,500 recordings of digits 0 to 9 from various English speakers. The dataset was divided into two non-overlapping sets, 1,000 digits for training and 500 digits for testing. The recordings of spoken digits were translated into input vectors of Mel-frequency cepstral coefficients, a common pre-processing technique for a speech recognition task [2, 27, 170]. Each recording segment of a digit was divided into 3:4:3 ratio regions with 13 Mel frequency cepstral features for each region [44]. In addition, each region was calculated for its delta and delta-delta coefficients in order to capture all features, and the final result is a 117-coefficient vector. The coefficient vectors were translated into input voltages and scaled to a range of $[-\nu, \nu]$ to avoid overstimulating reservoirs.
6.4.3 MNIST Recognition as a Spatial Task

*Mixed National Institute of Standards and Technology* (MNIST) patterns are a common benchmark to evaluate the performance of various brain-inspired networks [51, 110, 180, 181]. The dataset is a collection of handwritten digits from various people. The handwritten digits are normalized to fit into \(28 \times 28\) pixel images of grayscale levels. The dataset is divided into two non-overlapping subsets: 60,000 train images and 10,000 test images. Images were flattened into 784-pixel vectors as inputs to reservoirs for training and testing. The amplitudes of the pixel vectors were scaled to a range of \([0, \nu]\) to avoid an overstimulated internal state of the reservoirs.

![Figure 6.4: Converting pixel images into input vectors. Each input vector \(\vec{x}_k\) contains pixel values of a \(m \times n\) image. Pixel images as spatial information are applied to the reservoir in a temporal process from \(t_0 \rightarrow t_k\).](image)

Although reservoirs with memristive devices can accept DC inputs as pixel values, memcapacitive reservoirs do not respond to DC inputs due to their capacitive nature. From an electrical standpoint, the memcapacitive reservoirs are just the electric circuits with capacitive devices that do not allow DC signals to pass through. Consequently, the
6.4. BENCHMARKS AND VALIDATION

Pixel values as DC values are converted into AC signals. The converting process of the pixel values is depicted in Figure 6.4.

Assume that an image $k$ ($m \times n$) is represented by a pixel vector $\vec{x}_k$. The pixel vector $\vec{x}_k$ contains pixel values of $\{x_0, x_1, x_2, ..., x_l\}$ and is applied to a reservoir for training or testing at a constant interval $\Delta t$ ($\Delta t = t_{k+1} - t_k$). Inputs to the reservoir will be streams of values that occur after each $\Delta t$ as follows:

\[
\begin{align*}
  v_0(t) &= \{x_0(t_0), x_0(t_1), x_0(t_2), \ldots, x_0(t_k)\}, \\
  v_1(t) &= \{x_1(t_0), x_1(t_1), x_1(t_2), \ldots, x_1(t_k)\}, \\
  v_2(t) &= \{x_2(t_0), x_2(t_1), x_2(t_2), \ldots, x_2(t_k)\}, \\
  \vdots & \\
  v_i(t) &= \{x_i(t_0), x_i(t_1), x_i(t_2), \ldots, x_i(t_k)\}.
\end{align*}
\]

Consequently, an input stream to the reservoir is expressed as a time function $f_i(t)$:

\[
f_i(t) = v_i(t) = x_i(t) \quad \text{with } t = k\Delta t, k \in \{0, 1, 2, 3, \ldots, l\},
\]

where $\Delta t = t_{k+1} - t_k$ and $l$ is the total number of images for training or testing.

Equation 6.7 describes how pixel images are converted into non-DC inputs for training and testing the reservoir. The amplitudes of pixel vectors are normalized to a range of $[0, \nu]$ to ensure that memcapacitive devices within a reservoir are not overstimulated or under-stimulated. Only subsets of training and testing sets, 1,000 training digits and 400 testing digits, are selected due to the simulation time of large reservoirs. Experiment results show that a higher number of training and testing images than those in the selected
sets does not significantly improve the performance of reservoirs.

### 6.4.4 CIFAR-10 Recognition as a Complex Spatial Task

CIFAR-10 is a dataset collection of 60,000 color images with a size of 32x32 pixels. The images are categorized into 10 classes of objects: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is divided into 50,000 images for training and 10,000 images for testing. Classifying objects in the CIFAR-10 dataset is particularly challenging for brain-inspired networks due to the complexity of image features represented by color intensity. The CIFAR-10 is a common benchmark for reservoirs, convolution neural networks, and deep networks [34, 65, 111].

![CIFAR-10 Image](image1)

(a) A CIFAR-10 image.

![Grayscale Image](image2)

(b) An equivalent grayscale image.

Figure 6.5: An example of a CIFAR-10 image. A CIFAR-10 color image (a) is converted into a grayscale image (b) for dimension reduction. The original color image (a) has a high dimension of 32 \times 32 \times 3 whereas the grayscale image has a lower dimension of 32 \times 32.

The CIFAR-10 color images are represented in Red-Green-Blue (RGB) format, which
has a high dimension of $32 \times 32 \times 3$. Such color images require a large reservoir for the training and testing process, which, in turn, demands a longer simulation time. Converting color images into grayscale images is an alternative to reduce image dimension. Preprocessing color images into grayscale images is one of the common techniques for dimension reduction in training and testing neural networks [151]. An example of image dimension reduction is shown in Fig. 6.5. After a dimension reduction process, the original color image (Fig. 6.5a), which has a high dimension of $32 \times 32 \times 3$, becomes an equivalent grayscale image (Fig. 6.5b) with a lower dimension of $32 \times 32$.

Grayscale images are translated into pixel-value vectors, which are normalized to a range of $[0, \nu]$ to ensure that reservoirs are not overstimulated or under-stimulated. Similar to the MNIST dataset, only subsets of training and testing sets, 1,000 training digits and 400 testing digits, are selected due to the simulation time of large reservoirs. Selecting a higher number of training and testing images than those in the selected sets does not significantly improve the performance of reservoirs, reflected in experiment results.

### 6.5 BASELINE REFERENCE NETWORKS

*Echo State Network* (ESN) is a traditional fully connected network, developed by Jaeger [89], that has been adopted as a general framework for various neural networks [123] and reservoir computing approaches [20]. In my work, ESN was selected as a baseline network for comparisons with the results of mem-device reservoirs. In addition, a modified ESN was also employed as a reference for comparison. This modified ESN is called *Small-world Power-law Echo State Network* (SPESN). In the SPESN, the fully connected random topology was replaced with a small-world power-law structure and the hyperbolic tangent function was the transfer function at the output of each node. The
spectral radius of the SPESN was set to 0.9 with a scaling input of 1.0 in order to avoid a chaotic response [87].

### 6.6 NANO WIRE MODEL AND POWER CALCULATION

As shown in Fig. 5.1, each memcapacitive device was connected to nodes through nanowires. The nanowire resistance was modeled as a resistor $R_{nw}$, whose resistance is approximately $5\Omega/\mu$m [161]. In order to simplify the calculations of nanowire resistance, memcapacitive devices were assumed to have a uniform length of $10\,nm$ [50] and the nanowire length $l_n$ was the Euclidean distance between nodes after subtracting the device length. The total nanowire resistance with a Euclidean distance $l_{nw}$ is: $R_{nw} = l_{nw} \times 5\Omega/1\mu$m.

The nanowire resistor $R_{nw}$ and the memcapacitor $C_m$ form an RC circuit whose transient response in a charging phase is:

\[
\begin{align*}
\nu_{C_m}(t) &= v_{in} + (V_o - v_{in})e^{-\frac{t}{\tau_{nw+m}}} \\
\nu_{R_n}(t) &= (v_{in} - V_o)e^{-\frac{t}{\tau_{nw+m}}} \\
\mathcal{P}_{nw}(n) &= \frac{1}{\Delta t} \int_{t_0}^{t_0 + \Delta t} \frac{v_{R_n}^2(k)}{R_{nw}} \, dk \\
\mathcal{P}_{nw}(n) &= \frac{1}{R_{nw} \Delta t} \int_{t_0}^{t_0 + \Delta t} (v_{in} - V_o)^2 e^{-\frac{2k}{\tau_{nw+m}}} \, dk,
\end{align*}
\]

where $v_{in}$ is the node voltage at the time step $n$, $V_o$ is the initial voltage in the capacitor $C_m$ at the time step $n - 1$, $C_m$ is the capacitance of the memcapacitor at the time step $n$, $\mathcal{P}_{nw}(n)$ is the dissipated power consumed by the nanowire resistor $R_{nw}$, and $\Delta t$ is the duration of a time step.

At each time step, the dissipated power in each nanowire was calculated with Eq. 6.8 using the capacitance of the device at the time step $n$. The calculation is a close
approximation of actual power dissipated in each nanowire at each time step. The total power dissipated in nanowires is the sum of calculated power over all time steps.
The aim of Chapter 7 is to attempt to address the RC questions Q1 and Q2 (Chapter 2) by exploring the memcapacitive network dynamics in RC. Previous works have shown that random memristive networks can be used as reservoirs in reservoir computing [11, 166, 173]. The random memristive networks exhibit both nonlinear dynamics and memory capability, which are the fundamental characteristics of a reservoir. Furthermore, simulation results from other studies underlined that memristive reservoirs not only are possible but also capable of performing simple [49, 51] as well as complex tasks [29–32, 106]. A memristive device, by its nature, is a resistor that dissipates power and limits how far memristive reservoirs are able to reduce their power consumption. This is where memcapacitors, new mem-devices [22, 131], are promising. In this chapter, I will investigate the characteristics of memcapacitive reservoirs in several areas: the fading memory effect of the networks, the influence of topological structure on the power consumption of reservoirs, the property of small-world power-law structures, the dynamic response of mixed mem-device reservoirs, the behavior of spiking reservoirs, and the impact of device faults and variations on the performance of reservoirs.
7.1 FADEING MEMORY EFFECT

The volatile effect of a memcapacitive device is guaranteed by the decay term $f_d()$ in Eq. 6.4. It is not clear, however, whether or not a reservoir as a collective and nonlinear network of memcapacitive devices expresses a fading memory effect. Here, I investigated the fading phenomenon of memcapacitive reservoirs at the device level and at the system level. The simulation results will address research question 1.1 (See Section 2.1).

<table>
<thead>
<tr>
<th>Device Model</th>
<th>Reservoir Type</th>
<th>Decay $\theta$</th>
<th>$Ts$ (µs)</th>
<th>Scaling Factors $\dagger$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek [22]</td>
<td>Memcapacitive</td>
<td>3.00</td>
<td>1.0</td>
<td>$S$</td>
</tr>
<tr>
<td>Mohamed [131]</td>
<td>Memcapacitive</td>
<td>3.00</td>
<td>1.0</td>
<td>$S/1e4 + 0.65451$</td>
</tr>
<tr>
<td>Chang [94]</td>
<td>Memristive</td>
<td>1.13</td>
<td>1.0</td>
<td>$S/1e2 + 0.486$</td>
</tr>
<tr>
<td>Oblea [138]</td>
<td>Memristive</td>
<td>0.15</td>
<td>0.1</td>
<td>$S/2e1 + 0.449$</td>
</tr>
<tr>
<td>SPESN</td>
<td>Software</td>
<td>-</td>
<td>-</td>
<td>$S/5e1$</td>
</tr>
</tbody>
</table>

$\dagger$ The scaling factors for device states shown in Fig. 7.1a. Since the device state and the response time are sensitive to the physical property of a mem-device, they are scaled so that the device state response of mem-devices can fit into a single display.

Four random spoken digits were selected as inputs to simulate the fading memory effect of five reservoirs: Biolek, Mohamed, Chang, Oblea, and SPESN. The parameters of the reservoirs are as follows: $\alpha = 1.02, \beta = 0.71, \delta = 0.0$, and $\gamma = 0.0$. The decay factor and the time step were device-dependent and the settings are shown in Table 7.1. To maintain a consistent response, all reservoirs had the same initial states. Four 117-coefficient vectors as the input signals were scaled to an input range of $[-\nu, \nu]$ with $\nu = 8.0$ to provide enough stimuli to the reservoirs. Each coefficient was translated into a pulse of 400-time steps, which is a common technique to test the fading memory of a reservoir [52]. The input signals were active for 400-time steps and inactive for 1,000-time steps. Since the response time depends on the physical property of a mem-device, the inactive
7.1. FADING MEMORY EFFECT

state of the input signals (1,000-time steps) was long enough to illustrate the fading effect of all devices.

![Fading Memory at the Device Level](chart1)

(a) Fading memory at the device level.

![Fading Memory at the System Level](chart2)

(b) Fading memory at the system level.

Figure 7.1: Fading memory of mem-device reservoirs. The settings of reservoir parameters were: \( \alpha = 1.02, \beta = 0.71, \delta = 0.0, \) and \( \gamma = 0.0; \) (a) \( Vin_2 \), one of 117 input pulses, shows the time intervals where the inputs were active \([0.0ms, 0.4ms]\) and inactive \([0.4ms, 1.4ms]\). Since the internal states and the response time are device-dependent, the measurements of the device states and the response time were scaled to fit into a single display for a comparison. The scaling factors of device states and times are in Table 7.1. All device states were displayed with time measurements in ms, except the SPESN whose states were shown with respect to the number of time steps; (b) the plots of the Euclidean distance of the Chang, Mohamed, Oblea and SPESN reservoirs were shifted downward for clarity. For the SPESN reservoir, the unit for x-axis is the number of time steps \((\times 1e9)\). The vertical line is where the Euclidean distances of the reservoir states were approaching a cyclic steady state, indicating that the initial states of mem-devices no longer had any effect on the reservoir states.

The corresponding average device states are shown in Fig. 7.1a when the reservoirs were simulated with four input coefficient vectors. Since the internal state and the response time are unique for each device, the measurements of the device states and their response time were scaled to fit into a single display for clarity. The scaling factors are indicated in Table 7.1. The input signals were active in \([0.0ms, 0.4ms]\) and inactive in \([0.4ms, 1.4ms]\), indicated by \( Vin_2 \) as an example of the 117 input pulses. During the
7.1. FADEING MEMORY EFFECT

time interval [0.0ms, 0.4ms], the internal states of mem-devices within the reservoirs converged to a certain state. Once the input signals were off \((t \geq 0.4ms)\), the internal states of mem-devices gradually declined over time to a reset state, except the Oblea devices whose internal states remained unchanged. The response of the Oblea reservoir is due to the fact that the Oblea device is a non-volatile device, indicated in Table 6.1. The gradual decay of internal states without active input signals characterizes the fading memory effect of the reservoirs at a device level. Without applied signals, the reservoir state, constituted by the device states, gradually returns to a reset state.

Next, mem-device reservoirs were investigated to see how they exhibited the fading memory effect at a system level by calculating the Euclidean distance between the trajectories of the reservoir states for the same input conditions but different initial states of reservoir devices. Calculating the Euclidean distance between state trajectories is a common approach to understanding the dynamic property of a reservoir [190]. At a time \(t\), the Euclidean distance \(d_{u,v}(t)\) between the trajectories of two reservoir states \(U(t)\) and \(V(t)\) is defined as:

\[
d^2_{u,v}(t) = \sum_{i=1}^{N} [u_i(t) - v_i(t)]^2,
\]

where \(u_i(t)\) and \(v_i(t)\) are the reservoir states (node voltages) at time \(t\) and \(N\) is the number of the reservoir nodes.

Each input signal was a series of twenty similar sequences and each sequence was a combination of ten random sounds of the same class digit. Each reservoir was simulated for 100-time steps and the reservoir states were collected at each time step. The input sequence and the simulated time steps are sufficient to illustrate the fading effect of the reservoirs. The Euclidean distance was calculated using the reservoir states of the same reservoirs at two different runs with different initial states. All reservoirs had similar
7.2. INFLUENCE OF TOPOLOGICAL STRUCTURE ON POWER CONSUMPTION

network parameters $\alpha, \beta, \delta,$ and $\gamma$. The duration of the time step, however, was adjusted to smaller values in order to capture the cyclic states of the reservoirs.

Table 7.3: Time Step Settings for Reservoir State

<table>
<thead>
<tr>
<th>Device Model</th>
<th>Reservoir Type</th>
<th>Time Step (ns)</th>
<th>Time Scale $\times 1e7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek [22]</td>
<td>Memcapacitive</td>
<td>1.0</td>
<td>100.0</td>
</tr>
<tr>
<td>Mohamed [131]</td>
<td>Memcapacitive</td>
<td>10.0</td>
<td>10.0</td>
</tr>
<tr>
<td>Chang [94]</td>
<td>Memristive</td>
<td>100.0</td>
<td>1.0</td>
</tr>
<tr>
<td>Oblea [138]</td>
<td>Memristive</td>
<td>100.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

$\dagger$ Time scaling factors are used to fit all Euclidean distance plots in a single display for clarity, as shown in Fig. 7.1b.

Figure 7.1b presents the Euclidean distances of reservoir states when the reservoirs were simulated using the isolated spoken digit 3. The plots of the Euclidean distances of the Chang, Mohamed, Oblea and SPESN reservoirs were shifted downward for clarity. The vertical line at $t = 5 \text{ns}$ illustrates the time when the Euclidean distances of the reservoir states were approaching a cyclic steady state. The cyclic steady-state phenomenon suggests that the information contained in the initial conditions of the reservoir states is no longer relevant and that the fading memory effect took place at a system level.

7.2 INFLUENCE OF TOPOLOGICAL STRUCTURE ON POWER CONSUMPTION

Studies have shown that both the random and regular structures provide sufficient dynamics for reservoirs to perform tasks [29, 51] although the random topology is a preferable option for reservoirs due to the simple process in fabrication [49]. Is it possible to find a network topology where mem-device reservoirs achieve comparable performance to other reservoirs while reducing the power consumption? This is research question 1.2. As mentioned in Sections 1.2 and 5.2, SW network structures reflect the nature of physical
7.2. INFLUENCE OF TOPOLOGICAL STRUCTURE ON POWER CONSUMPTION

networks [197] and closely resemble the neural structure of a biological brain [53, 172]. Here, I want to explore whether the SW topology can offer a solution.

Five network types were selected as the structures for reservoirs in my experiments: ESN is a software reservoir used as a reference for comparison, regular topology based on a 2D-grid structure, randomly connected networks using Erdös-Rényi’s approach [56], SW organized network [197], and SWPL topology. The reservoirs were trained and tested with three benchmarks – NARMA10, Isolated spoken digits, and MNIST (See Section 6.4)– for their performances in root-mean-square errors and accuracy classifications. The inputs, outputs, and the time step $Ts$ were task-specific and depended on the sensitivity of the devices. The settings of these parameters are shown in Table 7.5.

Table 7.5: Input and Output Settings for Reservoirs.

<table>
<thead>
<tr>
<th>Device Model</th>
<th>Type†</th>
<th>NARMA-10</th>
<th>Spoken Digits</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>In</td>
<td>Out</td>
<td>In</td>
</tr>
<tr>
<td>ESN</td>
<td>-</td>
<td>3</td>
<td>9</td>
<td>117</td>
</tr>
<tr>
<td>Biolek [22]</td>
<td>MC</td>
<td>3</td>
<td>9</td>
<td>117</td>
</tr>
<tr>
<td>Mohamed [131]</td>
<td>MC</td>
<td>18</td>
<td>4</td>
<td>117</td>
</tr>
<tr>
<td>Chang [94]</td>
<td>MR</td>
<td>5</td>
<td>3</td>
<td>117</td>
</tr>
<tr>
<td>Oblea [138]</td>
<td>MR</td>
<td>6</td>
<td>2</td>
<td>117</td>
</tr>
</tbody>
</table>

† MC denotes memcapacitive reservoirs and MR is memristive reservoirs.
The inputs, outputs, and time step $Ts$ were set accordingly to the tasks and the sensitivity of the devices.

Figure 7.2 displays the simulation results of topological reservoirs on different tasks. The horizontal blue dashed line is the baseline, which is based on the performances of the ESNs, for comparisons. The horizontal green dashed line is the power reference line that is based on the power performances of SWPL reservoirs. The simulation results of the Biolek memcapacitive reservoirs and the Chang memristive reservoirs show that the performances of the reservoirs are independent of device implementation.
7.2. INFLUENCE OF TOPOLOGICAL STRUCTURE ON POWER CONSUMPTION

Figure 7.2: Performance of mem-device reservoirs for different tasks and network topologies. The performance of the ESNs (blue dashed line) and the power consumption of SWPL reservoirs (green dashed line) are the baselines for comparisons. The results suggest that selecting an appropriate topology for the reservoirs, such as regular or SWPL topology, will reduce power consumption without sacrificing their performance.
7.3 PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

In Figure 7.2, the regular and SWPL reservoirs achieved comparable performances to ESN reservoirs for all three tasks (the blue dashed lines). Furthermore, one can observe that both the regular and SWPL reservoirs maintain their comparable performances with few mem-devices. As a result, compared to the other reservoirs (green dashed green lines), both the regular and SWPL reservoirs reduced power consumption by an average factor of $28\times$, and $31\times$, respectively. Compared to the regular reservoirs, the SWPL reservoirs, on average, consumed less power by a factor of $1.2\times$.

7.3 PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

The results in Section 7.2 showed that the SWPL reservoirs consumed less power than the regular reservoirs by a factor of $1.2\times$. Is it possible to find a network topology that lies between a regular to a random topology to further reduce the power consumption without sacrificing reservoir performance? This is the goal of research question 1.3 (See Section 2.1).

As shown in Section 5.3, the characteristics of SWPL structures emerge with different settings for $D, \alpha, \beta, \delta$, and $\gamma$, particularly the probability of rewiring or the randomness $\beta$ and the locality $\alpha$. In this section, I investigated how the dynamical response of mem-device reservoirs changes when the randomness $\beta$ and the locality $\alpha$.

Due to the stochastic process of forming SWPL network reservoirs from grid networks, for each combination of $\alpha$ and $\beta$, twenty instances of a mem-device reservoir with the similar settings of network parameters were simulated and then the results were averaged. The cost of different topological networks was considered by determining the wiring cost of each network topology that is associated with the nanowire resistance $R_{nw}$ and the dissipated power $P_{nw}$. 
7.3. PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

Table 7.7: Input Scaling Factors for Simulation Sweeps

<table>
<thead>
<tr>
<th>Device Model</th>
<th>Reservoir Type</th>
<th>( W_{\text{total}} ) (O)</th>
<th>Scaling Input Factor ( \nu ) ‡</th>
<th>Spoken Digits</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek [22]</td>
<td>Memcapacitive</td>
<td>69</td>
<td>0.118</td>
<td>1.774</td>
<td></td>
</tr>
<tr>
<td>Mohamed [131]</td>
<td>Memcapacitive</td>
<td>69</td>
<td>1.382</td>
<td>1.262</td>
<td></td>
</tr>
<tr>
<td>Chang [94]</td>
<td>Memristive</td>
<td>69</td>
<td>1.630</td>
<td>0.935</td>
<td></td>
</tr>
<tr>
<td>Oblea [138]</td>
<td>Memristive</td>
<td>69</td>
<td>0.466</td>
<td>0.257</td>
<td></td>
</tr>
<tr>
<td>SPESN †</td>
<td>Software</td>
<td>( N - \text{Inputs} - 3 )</td>
<td>0.118</td>
<td>1.774</td>
<td></td>
</tr>
</tbody>
</table>

† The number of trained weights for the output layer was set accordingly to the reservoir nodes \( N \) for SPESN reservoirs in order to avoid overfitting [116].
‡ Scaling input signals prevents the over-stimulated or under-stimulated conditions of the reservoirs due to device input sensitivity.

From the isolated spoken digits and MNIST datasets, ten classes of digits, zero to nine, were selected as input signals to measure the dissimilar class distance \( C_d(t) \) of mem-device reservoirs. Each class had 145 inputs of the same digits to measure the similar class variance \( C_v(t) \). The separation property of the reservoirs was calculated from the dissimilar class distance and the similar class variance defined in Eq. 6.5. The estimation of Lyapunov exponents (Eq. 6.6) was based on 8 input signals whose sequence was composed of fifty randomly selected digits. The digit wave files and images were converted into input vectors. The input vectors, in turn, were translated into electrical voltages with a scaling range of \([-\nu, \nu]\) to provide stimuli to the reservoirs. The reservoirs were simulated with three sweep types: reservoir-node sweep, randomness \( \beta \) sweep, and locality \( \alpha \) sweep. Since the primary focus was the dynamical responses of reservoirs under the influence of randomness, \( \beta \), and locality, \( \alpha \), the removed links \( \gamma \) and the added links \( \delta \) were set to zero.

The range of the locality \( \alpha \) is [0.0, 2.4] and the range of the randomness \( \beta \) is [0.0, 1.0]. Both \( \alpha \) and \( \beta \) were set to 1.2 and 0.5, the midpoints of the intervals, for the reservoir node sweep. The scaling factor \( \nu \) of input signals, specified in Table 7.7, was set accordingly to
the input sensitivity of the devices to prevent an overstimulated behavior of the reservoirs. In order to avoid overfitting, the number of trained weights for the SPESN reservoirs was set accordingly to the reservoir nodes $N$: $O = N - Inputs - 3$. As Lukoševičius pointed out, for an ESN a sufficient number of trained weights with the regularization approach prevents the output layer from over-fitting the input data during the training phase [116].

All reservoirs were trained and tested for their classification performances. In addition, the reservoir states were collected for the measurements of the separation property and the estimation of Lyapunov exponents.

The simulation results of the node sweep are shown in Fig. 7.3. The classification performance of the SPESN reservoir on the MNIST dataset (Fig. 7.3f) declined with an increasing number of reservoir nodes. Study showed that a large number of trained weights for the readout layer of an ESN could escalate a slight deviation of the output from the expected value in subsequent time steps whereas an insufficient number of trained weights could prevent the readout layer to characterize the dynamics of reservoirs [116]. In my simulations, I selected experimentally the trained weights for the SPESN readout layer as specified in Table 7.7. The trained weights were not sufficient, which resulted in a decline in the performance. Finding the optimal number of trained weights for large-scale SPESN reservoirs is subject of further research.

As shown in Figs. 7.3c and 7.3d, the mem-device reservoirs display a similar pattern of the Lyapunov estimations where the estimations increased with a larger reservoir. For the spoken digit task, however, the SPESN reservoirs exhibited a different response in which the Lyapunov exponent estimations started to decline when the reservoir nodes were more than 400 nodes (Fig. 7.3c). This phenomenon was reflected in the decline of the performance when the performance of the reservoirs reached a plateau at 400 nodes and slightly decreased after that point (Fig. 7.3e). Our simulation results for the SPESN
Figure 7.3: Dynamic measurements and classifications of reservoirs ($\alpha = 1.2$ and $\beta = 0.5$). Reservoirs were simulated using Spoken digit dataset (a, c, and e) and MNIST dataset (b, d, and f). The Lyapunov exponent estimations show a consistent pattern and provide a useful metric to predict the mem-device reservoir performance.
reservoirs with the spoken digit task coincided with the findings in another study where increasing the size of an ESN did not improve its performance [211].

The separation measurements on both tasks did not produce a consistent pattern that can be correlated with the reservoir performance (Figs. 7.3a and 7.3e, 7.3b and 7.3f). The Lyapunov exponent estimations, on the other hand, illustrate a positive correlation with the reservoir performance, in which the reservoirs with higher estimations of Lyapunov exponent achieved higher performance. This phenomenon is also observed in the work of Chrol-Cannon and Jin where the performance was positively correlated to the Lyapunov exponent estimations of an ESN [42]. In fact, the Lyapunov exponent has been used as a metric to quantify the computational dynamics of a recurrent neural network [24]. I will use the estimations of Lyapunov exponents as a metric to measure the dynamic response of mem-device reservoirs.

Figure 7.4 shows the simulation results of the randomness sweep ($\beta$) from a regular topology ($\beta = 0.0$) to a completely random structure ($\beta = 1.0$). Three reservoir types were simulated: highly global-connection type ($\alpha = 0.0$), SWPL connection type ($\alpha = 1.2$), and highly local-connection type ($\alpha = 2.4$). Both reservoirs, the Biolek memcapacitive and the Chang memristive reservoirs, had a similar response for the estimations of Lyapunov exponents (Figs. 7.4a and 7.4b) where the estimations declined with a higher level of randomness $\beta$. It means that the chaotic nature of the reservoirs decreased with a higher number of random connections. This phenomenon was also observed in the formation of SW networks from ring networks in which the chaotic nature of the SW networks decreased with a high level of randomness [208].

The classification performances (Figs. 7.4c and 7.4d) were about 88% for the Biolek reservoirs and 90% for the Chang reservoirs. The Chang reservoirs, however, experienced a higher level of variations between 0.2 and 1.0 of the randomness $\beta$. The highly local
7.3. PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

Figure 7.4: Lyapunov exponent estimation, classifications, and wire dissipated powers of reservoirs for the $\beta$ sweep. The Biolek reservoirs were a memcapacitive type whereas the Chang reservoirs were a memristive type. Reservoirs were simulated with three topological types: highly global ($\alpha = 0.0$), small-world power-law ($\alpha = 1.2$), and highly local ($\alpha = 2.4$). The change in $\beta$ does not affect the performance of the reservoirs but contributes linearly to the dissipated power in nanowires.
7.3. PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

reservoirs ($\alpha = 2.4$) in both cases, moreover, had more performance variations compared to the highly-global SWPL reservoirs. Since the results were the average results of twenty reservoir instances of similar settings simulated at each value of $\beta$, I suspect that having more reservoir instances (e.g., 100 reservoir instances) will reduce the variations in the reservoir performances. On average, the performances of each reservoir remained the same across for all $\beta$.

As expected, the dissipated power in the nanowire resistors grew with an increasing $\beta$ (Figs. 7.4e and 7.4f). As shown in Section 5.3, the randomness $\beta$ denotes the probability of local connections being replaced by global links in the formation of SWPL networks. With a higher level of randomness, more local connections of nearby nodes were replaced by global ones between distant nodes, which, in turn, increased the nanowire lengths and resistance. As a result, the total dissipated power in the nanowires extended with a higher value for the randomness $\beta$. The randomness $\beta$ did not affect the performance of the reservoir networks but increased the cost of the systems in terms of the nanowire lengths, the nanowire resistance, and the dissipated power.

Figure 7.5 illustrates the simulation results of the locality sweep ($\alpha$) as the reservoir networks evolved from a highly global-connection type ($\alpha = 0.0$) to a highly local-connection type ($\alpha = 2.4$). Three topologies of reservoir networks were simulated: regular topology ($\beta = 0.0$), SWPL topology ($\beta = 0.5$) and totally random topology ($\beta = 1.0$). The estimations of Lyapunov exponents (Figs. 7.5a and 7.5b) illustrate an almost linear increase with respect to the locality ($\alpha$) for the SWPL and random reservoir topologies. Once again, the results portray a similar pattern that was observed in small-world networks where the chaotic nature of the networks decreases with higher global connections [208]. In Figs. 7.5a and 7.5b, the reservoirs with low values of $\alpha$ were highly global networks, whereas the highly local networks were associated with high values of $\alpha$. The
Figure 7.5: Lyapunov exponent estimation, classifications, and wire dissipated powers of reservoirs for the $\alpha$ sweep. The Biolek reservoirs were a memcapacitive type whereas the Chang reservoirs were a memristive type. Reservoirs were simulated with three topological structures: regular ($\beta = 0.0$), small-world power-law ($\beta = 0.5$), and totally random ($\beta = 1.0$). Similar to the $\beta$ sweep, the change in $\alpha$ only contributes to the dissipated power in the nanowires but not the performance.
7.3. PROPERTY OF SMALL-WORLD POWER-LAW NETWORKS

chaotic nature of the reservoirs decreased as the reservoirs evolved from a local to a global topological structure.

Figures 7.4c and 7.4d show the classification performances. Similar to the randomness sweep, the reservoir performances fluctuated between 88% and 90%. Having more instances of reservoirs (e.g., 100 instances) would minimize the level of variation in reservoir performances. On average, the Biolek and Chang reservoirs achieved about 88% and 91%, regardless of the change in the locality ($\alpha$).

The dissipated power of the nanowires, however, decreased when the reservoir networks had more local connections (high value of $\alpha$), shown in Figs. 7.5e and 7.5f. The locality $\alpha$ determines the probability of establishing connections between two nodes of a Euclidean distance $l$. The probability is governed by the power-law function: $p = l^{-\alpha}$. With a higher value of $\alpha$, remote nodes had less chance to establish connections than nearby nodes. Reservoirs with a higher value of $\alpha$ were more likely to have more local connections (short connections) and reservoirs with a lower value of $\alpha$ tended to have more global connections (long connections). As a result, global-connection reservoirs dissipated more power in the nanowires than the local-connection ones. The locality $\alpha$ did not influence the performance of the reservoirs but affected the cost of the systems, expressed in terms of the nanowire lengths and the dissipated power.

Simulation results indicate that both the locality $\alpha$ and the randomness $\beta$ affected the dissipated power in the nanowires of the reservoirs. Appropriate values of $\alpha$ and $\beta$ need to be selected in order to reduce wiring cost and the dissipated power without compromising the performance of the reservoirs. The heat map plots in Figure 7.6 are the summary results of the dissipated power in the nanowires, based on simulation results of the $\alpha$ and $\beta$ sweeps. Reservoirs with a high level of locality $\alpha$ and a low level of randomness $\beta$ (the lower right corner areas of the plot in Figs. 7.6a and 7.6b) will ensure low wiring
7.4. OPTIMIZING PARAMETERS FOR SPECIFIC TASKS

Figure 7.6: Dissipated power in nanowire resistance of Biolek reservoirs (a) and Chang reservoirs (b) for the spoken digit task. The optimal settings of $\alpha$ and $\beta$ is the blue region with high values of $\alpha$ and low values of $\beta$ that will minimize the dissipated power in the nanowires, and, therefore, the cost of the systems.

...cost and low dissipated power in the nanowires. Such settings of $\alpha$ and $\beta$, however, do not guarantee that such reservoirs will achieve optimal performance in terms of the classification, the number of devices within reservoirs, and the overall power consumption of systems. Finding reservoir parameters (not only $\alpha$ and $\beta$ but also other parameters) for optimal performance is the aim of the next section.

7.4 OPTIMIZING PARAMETERS FOR SPECIFIC TASKS

In previous experiments, $\alpha$ and $\beta$ were chosen with the mid-range values in order to illustrate the dynamic response and the nanowire dissipated power as the reservoirs underwent topological changes. Combinations of $\alpha$ and $\beta$ do not provide a full picture of reservoir dynamics since the reservoir dynamic response also depends on other parameters such as $\delta$, $\gamma$, $\nu$, $\theta$, and $T_s$ ($\nu$ is the scaling factor for input vectors, $\theta$ is the decay factor, and $T_s$ is the time step for input vectors). Finding an optimal setting for the reservoir param-
7.4. OPTIMIZING PARAMETERS FOR SPECIFIC TASKS

Parameters is an iterative process of fine-tuning experimental values, which is time-consuming and task-dependent. Here, an optimization algorithm, namely the particle swarm algorithm (Section 6.3), is selected to carry out the optimization task. The particle swarm algorithm is robust and capable of addressing various large-scale science and engineering optimization problems compared to the traditional genetic technique [38]. Due to the stochastic process in the optimization task, fifty instances of reservoirs with a similar setting were simulated to obtain average results for each generation. The target of the optimization task is where reservoirs reach optimal performance in terms of high classification correction and low power consumption. The results are in Tables 7.9, 7.11, and 7.13.

Table 7.9: Optimized Parameters for NARMA-10 Task

<table>
<thead>
<tr>
<th>Device</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$\gamma$</th>
<th>$\delta$</th>
<th>$\nu$</th>
<th>$\theta^{\ddagger}$</th>
<th>$T_s$ ($\mu$s)</th>
<th>NRMSE</th>
<th>$P_{\text{total}}$ (nW)</th>
<th>$P_{\text{nw}}$ (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek</td>
<td>2.2</td>
<td>0.62</td>
<td>48</td>
<td>0</td>
<td>0.10</td>
<td>0.67</td>
<td>10.8</td>
<td>0.45</td>
<td>0.04</td>
<td>3.91E-15</td>
</tr>
<tr>
<td>Mohamed</td>
<td>2.4</td>
<td>0.49</td>
<td>49</td>
<td>0</td>
<td>0.11</td>
<td>0.51</td>
<td>96.3</td>
<td>0.39</td>
<td>35.76</td>
<td>1.22E-10</td>
</tr>
<tr>
<td>Chang</td>
<td>2.5</td>
<td>0.47</td>
<td>35</td>
<td>0</td>
<td>0.15</td>
<td>-</td>
<td>500.0</td>
<td>0.38</td>
<td>316.31</td>
<td>2.08E-13</td>
</tr>
<tr>
<td>Oblea</td>
<td>2.7</td>
<td>0.57</td>
<td>50</td>
<td>0</td>
<td>0.12</td>
<td>-</td>
<td>0.5</td>
<td>0.43</td>
<td>172.27</td>
<td>1.80E-08</td>
</tr>
</tbody>
</table>

$\dagger$ MC denotes memcapacitive reservoirs and MR is memristive reservoirs.
$\ddagger$ Neither of the memristive models (Chang and Oblea) has the decay factor, $\theta$.

The reservoirs had 252 nodes with 20 inputs and 15 outputs; $\nu$ is the scaling factor for input vectors, $\theta$ is the decay factor, and $T_s$ is the time step for input vectors.

In the tables, $I$ is the number of input nodes, $O$ is the number of trained weights for the output layer, $N$ is the number of reservoir nodes, $\theta$ is the decay factor, and $T_s$ is the time step for input vectors. The decay factor $\theta$ is only applied to the memcapacitive models (Biolek and Mohamed) and controls the state volatility of the devices. The optimized results address the research questions 1.6 and 1.7 (See Section 2.1).

Figure 7.7 illustrates the simulation results of mem-device reservoirs for NARMA-10, Isolated Spoken Digits, and MNIST tasks with optimized parameters, as shown in Tables

101
7.4. OPTIMIZING PARAMETERS FOR SPECIFIC TASKS

Table 7.11: Optimized Parameters for Spoken Digit Recognition

<table>
<thead>
<tr>
<th>Device</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$\gamma$</th>
<th>$\delta$</th>
<th>$\nu$</th>
<th>$\theta^\dagger$</th>
<th>$T_s$ (ns)</th>
<th>Perf (%)</th>
<th>$P_{total}$ ($\mu W$)</th>
<th>$P_{nw}$ (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek</td>
<td>0.93</td>
<td>0.38</td>
<td>23</td>
<td>0</td>
<td>0.001</td>
<td>0.18</td>
<td>30.4</td>
<td>90.6</td>
<td>3.36E-7</td>
<td>8.13E-15</td>
</tr>
<tr>
<td>Mohamed</td>
<td>1.23</td>
<td>0.51</td>
<td>14</td>
<td>0</td>
<td>0.113</td>
<td>0.36</td>
<td>872.0</td>
<td>92.0</td>
<td>0.60</td>
<td>2.08E-8</td>
</tr>
<tr>
<td>Chang</td>
<td>1.19</td>
<td>0.58</td>
<td>20</td>
<td>0</td>
<td>0.465</td>
<td>-</td>
<td>5.0E5</td>
<td>91.0</td>
<td>6.34</td>
<td>7.43E-11</td>
</tr>
<tr>
<td>Oblea</td>
<td>1.64</td>
<td>0.55</td>
<td>49</td>
<td>0</td>
<td>0.611</td>
<td>-</td>
<td>203.8</td>
<td>89.3</td>
<td>2.97</td>
<td>2.37E-6</td>
</tr>
</tbody>
</table>

† MC denotes memcapacitive reservoirs and MR is memristive reservoirs.
‡ Neither of the memristive models (Chang and Oblea) has the decay factor, $\theta$.
The reservoirs had 422 nodes with 69 outputs; $\nu$ is the scaling factor for input vectors, $\theta$ is the decay factor, and $T_s$ is the time step for input vectors.

Table 7.13: Optimized Parameters for Classifying MNIST Images

<table>
<thead>
<tr>
<th>Device</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$\gamma$</th>
<th>$\delta$</th>
<th>$\nu$</th>
<th>$\theta^\dagger$</th>
<th>$T_s$ ($\mu s$)</th>
<th>Perf (%)</th>
<th>$P_{total}$ ($\mu W$)</th>
<th>$P_{nw}$ (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek</td>
<td>2.37</td>
<td>0.64</td>
<td>26</td>
<td>0</td>
<td>1.785</td>
<td>0.90</td>
<td>0.40</td>
<td>73.56</td>
<td>33.06</td>
<td>6.28E-7</td>
</tr>
<tr>
<td>Mohamed</td>
<td>2.15</td>
<td>0.27</td>
<td>43</td>
<td>0</td>
<td>0.325</td>
<td>0.14</td>
<td>0.96</td>
<td>77.97</td>
<td>94.16</td>
<td>8.06E-6</td>
</tr>
<tr>
<td>Chang</td>
<td>2.18</td>
<td>0.86</td>
<td>49</td>
<td>0</td>
<td>0.458</td>
<td>-</td>
<td>500.00</td>
<td>79.06</td>
<td>186.19</td>
<td>7.701E-9</td>
</tr>
<tr>
<td>Oblea</td>
<td>2.65</td>
<td>0.69</td>
<td>46</td>
<td>0</td>
<td>0.539</td>
<td>-</td>
<td>0.06</td>
<td>80.03</td>
<td>203.41</td>
<td>2.00E-4</td>
</tr>
</tbody>
</table>

† MC denotes memcapacitive reservoirs and MR is memristive reservoirs.
‡ Neither of the memristive models (Chang and Oblea) has the decay factor, $\theta$.
The reservoirs had 2067 nodes with 166 outputs; $\nu$ is the scaling factor for input vectors, $\theta$ is the decay factor, and $T_s$ is the time step for input vectors.
7.4. OPTIMIZING PARAMETERS FOR SPECIFIC TASKS

Figure 7.7: Performance and performance-power ratios of regular, random, and SWPL reservoirs on three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; in general, the SWPL reservoirs were able to achieve similar performances (a), but their performance-power ratios were higher (b) compared to regular and random reservoirs. SWPL reservoirs with optimized parameters yield the highest performance in term of the performance-power ratio.

7.9, 7.11, and 7.13. The reservoirs were simulated for three different topological structures: regular, SPWL, and random. The performances of the ESN were the reference for comparison. The results in Fig. 7.7a show that the SWPL reservoirs performed similarly to other reservoirs. Compared to the regular and random reservoirs, the SWPL reservoirs, on average, achieved higher the performance-power ratios (Fig. 7.7b). In fact, compared to the regular reservoirs, on average, the performance-power ratios of the SWPL reservoirs were better by a factor of $1.4\times$. Similarly, compared to the random reservoirs, the SWPL obtained better the performance-power ratio by a factor of $2.9\times$.

The SWPL reservoirs had better performance-power ratios due to the fact that they used fewer devices without losing their performances for the three tasks (Fig. 7.8). The optimized parameters $\alpha, \beta, \gamma, \nu, \theta$, and $T_s$ allowed the SWPL reservoirs to achieve optimal performances with a minimum number of devices. Compared to the regular and random
### 7.4. Optimizing Parameters for Specific Tasks

Figure 7.8: Number of devices and nanowires utilized by regular, random, and SWPL reservoirs for three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; the SWPL reservoirs used fewer devices (a) and a fewer number of nanowires (b) while maintaining similar performances compared to the regular and random reservoirs. SWPL reservoirs with optimized parameters utilize the least number of devices for the same tasks.

reservoirs, the SWPL reservoirs employed fewer devices by a factor of $1.6 \times$.

Figure 7.9: Total nanowire length and power consumption of regular, random, and SWPL reservoirs for three tasks: NARMA-10, Isolated Spoken Digits, and MNIST; although the SWPL topology required a longer nanowire length (a), the SWPL reservoirs consumed less the total power (b). The results suggest that the number of devices within a reservoir significantly contributes to the total power consumption of the systems.
Further observations indicate that the regular topology offered the least cost to systems in terms of the total nanowire length $L_{nw}$ and, therefore, the power dissipated in the nanowires, compared to the SWPL and random topologies (Fig. 7.9a). However, when considering the total power consumption, the SWPL reservoirs were the optimal choice for the reservoir topology (Fig. 7.9b). The results of the total power consumption suggest that the number of devices with reservoirs significantly contributes to the total power consumption of the reservoirs. Compared to the regular topology, the reservoirs with the SWPL topology consumed less power, by a factor of $1.4\times$. Similarly, compared to the random topology, the reservoirs with the SWPL topology reduced the power consumption by a factor of $14\times$. The trade-off between the cost to a system, illustrated by the total nanowire length, and its total power consumption is beyond the scope of my current work and is a subject of further research.

7.5 MIXED MEM-DEVICE RESERVOIRS

The fundamental characteristic of a reservoir is its dynamical structure that translates temporal-spatial input signals into a higher dimensional space. Any physical system with enough dynamics for processing information can function as a reservoir. A bucket of water, for instance, is capable of performing nonlinear computation [134]. Both memristive and memcapacitive devices are nonlinear and are adaptable in RC. So far, only reservoirs with one device type have been studied for their dynamic responses and performances. A network of combining memristive and memcapacitive devices can potentially offer a higher dynamical substrate for RC. For a specific task, what is the combination of memristive and memcapacitive devices that leads to high performance and low power reservoirs? This is the aim of the research question 1.5 (See Section 2.1).
7.5. MIXED MEM-DEVICE RESERVOIRS

Table 7.15: Setting Parameters for Mixed-device Reservoirs

<table>
<thead>
<tr>
<th>Task</th>
<th>Mixed Reservoir †</th>
<th>α</th>
<th>β</th>
<th>γ</th>
<th>ν</th>
<th>I</th>
<th>O</th>
<th>N</th>
<th>θ</th>
<th>Ts (µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARMA 10</td>
<td>Biolek &amp; Chang</td>
<td>1.3</td>
<td>0.1</td>
<td>46</td>
<td>0.11</td>
<td>19</td>
<td>18</td>
<td>289</td>
<td>0.69</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>0.5</td>
<td>0.5</td>
<td>8</td>
<td>0.11</td>
<td>10</td>
<td>11</td>
<td>212</td>
<td>0.26</td>
<td>50.8</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>1.6</td>
<td>0.5</td>
<td>12</td>
<td>0.11</td>
<td>11</td>
<td>17</td>
<td>374</td>
<td>0.37</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>0.3</td>
<td>0.6</td>
<td>5</td>
<td>0.12</td>
<td>3</td>
<td>5</td>
<td>55</td>
<td>0.79</td>
<td>71.2</td>
</tr>
<tr>
<td>Spoken Digits</td>
<td>Biolek &amp; Chang</td>
<td>0.4</td>
<td>0.6</td>
<td>2</td>
<td>1.11</td>
<td>117</td>
<td>55</td>
<td>1156</td>
<td>1.15</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>0.7</td>
<td>0.3</td>
<td>0</td>
<td>0.86</td>
<td>117</td>
<td>76</td>
<td>695</td>
<td>1.16</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>2.9</td>
<td>0.0</td>
<td>2</td>
<td>1.27</td>
<td>117</td>
<td>89</td>
<td>738</td>
<td>0.18</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>0.5</td>
<td>0.0</td>
<td>5</td>
<td>0.12</td>
<td>117</td>
<td>103</td>
<td>689</td>
<td>0.06</td>
<td>92.2</td>
</tr>
<tr>
<td>MNIST</td>
<td>Biolek &amp; Chang</td>
<td>0.4</td>
<td>0.3</td>
<td>5</td>
<td>0.95</td>
<td>784</td>
<td>128</td>
<td>1584</td>
<td>0.23</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>1.6</td>
<td>0.4</td>
<td>5</td>
<td>0.26</td>
<td>784</td>
<td>150</td>
<td>1675</td>
<td>0.90</td>
<td>56.6</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>2.3</td>
<td>0.3</td>
<td>3</td>
<td>1.00</td>
<td>784</td>
<td>142</td>
<td>1862</td>
<td>0.24</td>
<td>500.0</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>1.6</td>
<td>25.4</td>
<td>0</td>
<td>0.17</td>
<td>784</td>
<td>172</td>
<td>2073</td>
<td>0.52</td>
<td>24.0</td>
</tr>
</tbody>
</table>

† The settings of the network parameters are dependent on the sensitivity of memristive and memcapacitive devices. ν is the scaling factor for input vectors, I is the number of input nodes, O is the number of trained weights for the output layer, N is the number of reservoir nodes, θ is the decay factor, and Ts is the time step for input vectors.
7.5. MIXED MEM-DEVICE RESERVOIRS

Figure 7.10: Performance and power consumption of mixed mem-device Biolek-Chang and Mohamed-Chang reservoirs with respect to the percentage of the memristive Chang devices in the reservoirs. The performances of the reservoirs were based on the three benchmark tasks: NARMA-10, Isolated Spoken Digits, and MNIST.
7.5. MIXED MEM-DEVICE RESERVOIRS

Simulation sweeps of various combinations of memristive and memcapacitive devices (a sweep from 0% to 100%) were performed for different mem-device reservoirs. The performances and the power measurements are based on three tasks: NARMA-10, Isolated Spoken Digits, and MNIST. Due to the distinctive physical nature of each device that is receptive to an input stimulus, combining mem-device reservoirs required specific settings of network parameters. Table 7.15 illustrates the parameter settings of mem-device reservoirs for different tasks. In this Table, the time step (equivalent to the pulse width of a write pulse) of the Chang device is 500\(\mu s\), which is based on the physical nature of the device [51]. Other settings of the network parameters were derived from the findings of the optimal settings of the reservoirs using the particle swarm optimization, described in Section 7.4.

Figure 7.10 illustrates the simulation results of the Biolek-Chang and the Mohamed-Chang reservoirs with the three tasks. In general, the performances of the mem-device reservoirs (the blue lines) were improved with a higher percentage of memristive devices for the reason that the connections of memristive and memcapacitive devices formed RC filters, which further enhanced the dynamic response of the reservoirs. However, it is not the case for the NARMA-10 task. Further research will provide more data to understand this phenomenon. A higher percentage of memristive devices is associated with higher performance but also leads to more power consumption (green line plots).

The simulation results of the Biolek-Oblea and the Mohamed-Oblea reservoirs are shown in Figure 7.11. The responses of the Biolek-Oblea reservoirs were similar to those obtained from the Biolek-Chang reservoirs. For the Isolated Spoken Digit and MNIST tasks, the Mohamed-Oblea reservoirs, nonetheless, expressed unexpected responses. Since the response of the reservoirs is dependent on the physical behavior of combining devices, I suspect that the parameter settings were not sufficient to reflect the
7.5. MIXED MEM-DEVICE RESERVOIRS

Figure 7.11: Performance and power consumption of mixed mem-device Biolek-Oblea and Mohamed-Oblea reservoirs with respect to the percentage of the memristive Oblea devices in the reservoirs. The performances of the reservoirs were based on the three benchmark tasks: NARMA-10, Isolated Spoken Digits, and MNIST.
7.6 SPIKING INPUT SIGNALS

physical nature of the Mohamed-Oblea reservoirs. This will be part of further research.

Table 7.17: Optimal Percentage of Memristive Devices

<table>
<thead>
<tr>
<th>Task</th>
<th>Mixed Reservoir</th>
<th>Memristor Percentage (%)</th>
<th>RMSE † Performance (%)</th>
<th>Power (μW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARMA-10</td>
<td>Biolek &amp; Chang</td>
<td>45.18</td>
<td>0.0195</td>
<td>0.042</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>41.42</td>
<td>0.0204</td>
<td>0.064</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>27.33</td>
<td>0.0169</td>
<td>0.131</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>43.52</td>
<td>0.0196</td>
<td>0.105</td>
</tr>
<tr>
<td>Spoken Digits</td>
<td>Biolek &amp; Chang</td>
<td>77.23</td>
<td>86.2160</td>
<td>33.283</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>67.45</td>
<td>85.3320</td>
<td>8.479</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>49.33</td>
<td>85.2000</td>
<td>46.839</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>9.34</td>
<td>92.1280</td>
<td>0.277</td>
</tr>
<tr>
<td>MNIST</td>
<td>Biolek &amp; Chang</td>
<td>47.56</td>
<td>73.1450</td>
<td>570.754</td>
</tr>
<tr>
<td></td>
<td>Biolek &amp; Oblea</td>
<td>49.23</td>
<td>77.6800</td>
<td>10.722</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Chang</td>
<td>39.04</td>
<td>78.4250</td>
<td>1050.456</td>
</tr>
<tr>
<td></td>
<td>Mohamed &amp; Oblea</td>
<td>48.19</td>
<td>80.4800</td>
<td>9.861</td>
</tr>
</tbody>
</table>

† The measurements are in root-mean-square errors for NARMA-10 and in percentage corrections (%) for Isolated Spoken Digits and MNIST.

Table 7.17 summarizes the results where the mem-device reservoirs achieved their highest performances with minimum power consumption at the optimal settings of the memristive percentages. The optimal settings of the memristive percentages were determined from the particle swarm algorithm, as described in Section 6.3.

7.6 SPIKING INPUT SIGNALS

It has been shown that various models of spiking neural networks illustrate their capability of learning and achieving high classification performance with power efficiency [207]. The goal of this section is to investigate the additional complexity of training and testing spiking memcapacitive reservoirs in order to improve performance and power consumption (research question 1.6 in Section 2.1).
Figure 7.12: An example of spiking reservoir computing architecture. The Spike Converter translates spatial stimuli, such as MNIST, into spike trains as inputs to the reservoir. The Spike Accumulator converts the spike train outputs of the reservoir within a time interval $\Delta t$ to analog values for the Readout Layer. The Readout Layer utilizes a simple learning algorithm, such as ridge regression technique, during the training phase.

In the current RC architecture (Figure 3.11), the reservoirs can accept both analog and spiking inputs. However, the Readout Layer is trained with ridge regression algorithm and its inputs are analog values. In order to minimize the training complexity, additional functional units were added to the RC architecture for converting spatial information into spike trains and vice versa. A modified architecture is shown in Figure 7.12. The Spike Converter was added to translate spatial information into spike train inputs. In the experiments, MNIST was used as the common benchmark to compare the performances of analog and spiking memcapacitive reservoirs. Pixel values of a digit image were translated as the probability of firing a single spike within $\Delta T$ and spike trains were generated using the Poisson Spike algorithm [59]. It was proven that the maximum of 100 spikes is sufficient to encode digit images [59]. The Poisson Spike algorithm was chosen so that both analog and spiking reservoirs were trained and tested with the same images for com-
7.6. SPIKING INPUT SIGNALS

parison. The Spike Accumulator integrated the spike train outputs of the reservoir into the analog inputs for the Readout layer.

Table 7.19: Spike Settings

<table>
<thead>
<tr>
<th>Spiking Reservoir</th>
<th>Spike Amplitude (V)</th>
<th>Spike Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biolek</td>
<td>0.125</td>
<td>100ps</td>
</tr>
<tr>
<td>Mohamed</td>
<td>0.125</td>
<td>100ps</td>
</tr>
<tr>
<td>Chang</td>
<td>1.0</td>
<td>1ps</td>
</tr>
<tr>
<td>Oblea</td>
<td>0.5</td>
<td>1ps</td>
</tr>
</tbody>
</table>

For each mem-device model, analog and spiking reservoirs were simulated for the image classification using the same subset of the MNIST dataset: 1,000 trained images and 400 tested images. Both reservoirs had similar topology structures. The spike settings, which depended on the sensitivity of the devices, are shown in Table 7.19.

As shown in Figure 7.13, the mem-device reservoirs achieved similar performances, indicated by dashed blue lines. With spiking inputs, however, the mem-device reservoirs consumed much less power, represented by dashed green lines, compared to analog inputs. The low power consumption of the mem-device reservoirs was due to the fact that it took less power to produce spikes, which sufficiently transferred signals from one node to another within the reservoirs. The spiking train is the most efficient way for a biological neuron to conserve its power consumption in processing information while maintaining its small membrane area to house ion channels and receptors [137].

Figure 7.14 displays the power-times products of spiking and analog reservoirs. Despite the saving power feature, the power-time products of spiking reservoirs were higher than those of analog reservoirs, by a factor of $131\times$, $99\times$, $98\times$, and $100\times$ for BiolekC4, Mohamed, Chang, and Oblea, respectively. Often, the simulation time is not considered
Figure 7.13: Performances, power consumption, and simulation times of analog and spiking reservoirs. The horizontal dashed and green lines mark the lowest measurements of power consumption. The spiking reservoirs perform better than analog reservoirs in terms of power consumption but also require longer simulation time.
7.7 DEVICE FAULTS AND VARIATIONS

Device faults and variations are inevitable issues in any fabrication process. The fluctuation of device parameters is an uncontrollable factor that has a great impact on nano-CMOS transistors and circuits [37, 48, 188]. Various techniques and models have been implemented in each step of fabrication with an attempt to eliminate faults early in the process so that a higher yield is possible [124].

Although memristive devices were fabricated in a qualitatively identical process, they...
still exhibited significant variations in switching behavior [7, 164]. These variations not only affected the computational stage but also the learning stage when memristive devices were used as synapses for a neural network [164]. Memcapacitive devices are also susceptible to faults and variations. A small variation in the functionality of a device can contribute significantly to the unpredictable behavior of a reservoir, composed of a large number of memcapacitive devices. How do memcapacitive reservoirs react to device faults and variations? The exploration of research question 1.8 will shed some light on the reservoir behavior under the effect of faults and variations.

7.7.1 Fault and Variation Model

Stuck open and stuck short are a common fault in an electrical circuit [98]. An open or short fault is often modeled as an open circuit ($R = \infty$) or a short circuit ($R = 0 \Omega$). Simulating a short fault in a mem-device reservoir may lead to a convergent problem due to the numerical calculations.

![Fault and Variation Model](image)

Figure 7.15: Fault and Variation Model; for the open fault, the device is removed (a) whereas the device is replaced with a 0V source (b) to prevent convergent errors; (c) A Gaussian white noise with a standard deviation $\sigma$ is injected into the reservoir as a way to simulate device variations [68].

The fault models are shown in Figures 7.15a and 7.15b. The device is removed for
7.7. DEVICE FAULTS AND VARIATIONS

a stuck-open fault and is replaced by a 0V source for a stuck-short fault. The 0V source prevents convergent errors during simulations. Since both the memcapacitive devices (Biolek and Mohamed) did not specify coefficient variations of the devices, the simulations of device variations were introduced indirectly by injecting white noise (or Gaussian noise) with a standard deviation of $\sigma$ into the reservoirs, as shown in Figure 7.15c. It has been shown that the white noise sufficiently represented thermal noise and device variations [68]. At time step $t$, Gaussian white noise with a standard deviation of $\sigma$ is added to the weights of $n$ randomly selected devices to simulate the device variation for the time step $t + 1$ [68]. Adding Gaussian white noise to the weights of the devices is a common technique to simulate the device-to-device and the cycle-to-cycle variations that affect the overall performance of a network [68, 107].

7.7.2 Open-fault Simulation Results

Figure 7.16: The performance of mem-device reservoirs under the effect of the open-fault. The results indicate that the reservoirs were tolerant to a high percentage of open faults.

Figure 7.16 illustrates the performances of the mem-device reservoirs under the effect of open faults. In general, the mem-device reservoirs were able to accommodate up to
40% of device faults within the reservoir networks without losing their performances for both tasks. In fact, when the level of device faults was at 40%, the mem-device reservoirs still maintained their reasonable classification performances of 86.6% and 75.2% for the isolated spoken digits and the MNIST task, respectively. It has been shown that a feed-forward network with one hidden layer using the back-propagation algorithm suffered a loss in its XOR performance for open faults, about an increase of 60.14% error at the 22.22% of faults [176]. In contrast, at 20% of faults, the mem-device reservoirs only suffered a loss of 1.0% and 1.1% in their performance for the isolated spoken digit and the MNIST tasks, respectively. The fault-tolerant nature of mem-device reservoirs is due to the fact that the recurrent connections within the reservoirs allowed alternative paths where electrical signals could travel between nodes even in the presence of open faults.

### 7.7.3 Short-fault Simulation Results

![Graph](image-url)

(a) Isolated Spoken Digits.  
(b) MNIST Recognition.

Figure 7.17: The performance of mem-device reservoirs under the effect of the open-fault. The impact of short faults is much more severe to the performance of the reservoirs than the open faults.

The simulation results of short faults are shown in in Figure 7.17. For the isolated
spoken digit task, the reservoirs completely lost their performance at 10% of short faults (Figure 7.17a). In a similar way, the reservoirs totally failed at only 2% of short faults for the MNIST task (Figure 7.17b). From an electrical point of view, the mem-device reservoirs were electrical circuits where mem-devices were randomly connected between nodes. When a short fault occurred, similar to a short circuit between two nodes, other mem-devices connecting between two nodes became ineffective and the short circuit bypassed all electrical current. As a result, the impact of the short faults was much more severe to the performance of the reservoirs than the open faults.

**7.7.4 Device Variation Simulation Results**

It has been shown that it is possible to represent thermal noise and the device variations by adding white noise (or Gaussian noise) to a reservoir. Here, an additive Gaussian noise technique is adopted to simulate the effect of the variations of the devices, a common approach to processing images using neural networks [68, 107, 192]. The Gaussian white noise of zero mean and 0.01 of the standard deviation ($\sigma = 0.01$) was added to the weights of $n$ randomly selected mem-devices at a time step $t$ [68]. The Gaussian noise represented the device variation that contributed to the device weights for the time step $t + 1$. The simulation sweep for $n$ was from 0% to 100%.

Figure 7.18 summarizes the simulation results. The effect of the device variations was more severe for the isolated spoken digit task than the MNIST task with respect to the percentage of the device variation. For example, at 20% of the variation, the mem-device reservoirs lost almost 10% of their performance for the isolated spoken digit task but only 2% of their performance for the MNIST tasks. The simulation results suggest that the effect of the device variation on the performance of the reservoirs is task-dependent. The simulations of reservoirs with added Gaussian white noise did not accurately represent
Figure 7.18: The performance of mem-device reservoirs under the effect of device variations.

the device variation but reflected how the device variation might influence the overall performance of reservoirs.
In this chapter, I will try to answer the RC question Q3 by studying network dynamics of hierarchical and deep reservoirs. For the hierarchical reservoirs, I will present a hierarchical model, measure their kernel quality, obtain their performance and power consumption on complex tasks, and illustrate the dynamic behavior of mix-device reservoirs. For the deep reservoirs, I will adopt the model of deepESN [63] for deep memcapacitive reservoirs and verify their functionality on polyphonic music task.

8.1 HIERARCHICAL RESERVOIR COMPUTING

Conventional computing systems are based on a modular principle in which fundamental logic gates provide a building block for a circuit, and circuits are modules for building larger architectures. Most reservoir networks are monolithic and they are not easily scalable. Increasing the number of nodes within a reservoir does not necessarily increase the reservoir dynamics since the voltages at nodes (or the reservoir state) are correlated through mem-devices. Unlike the monolithic structure, each cluster node (each cluster node is a network) in hierarchical reservoirs is an isolated and uncorrelated unit and, therefore, characterizes better input signals. Triefenbach et al., for example, showed that hierarchical reservoir systems had better dynamics than single-layer reservoir net-
works in representing an acoustic model for speech recognition [185]. In another study, hierarchical reservoirs had better dynamics and outperformed monolithic reservoirs by at least 20% for certain tasks [30]. Hierarchical random Boolean network reservoirs surpassed the performance of random Boolean counterparts by at least 34% for binary image recognition, food-foraging trails, and memory recall tasks [39]. Recently, Dale observed that a hierarchical structure of ESNs improved the learning capability of the network to perform general tasks [45]. Can a hierarchical structure of memcapacitive reservoirs improve its computing capacities for complex tasks? What is the learning complexity for such hierarchical reservoirs? Research questions 2.1, 2.2, and 2.3 try to answer the role of hierarchical structure in RC (See Section 2.2).

### 8.1.1 Hierarchical Structure

Studies have shown that simplified structure reservoirs achieved equivalent performance compared to random reservoirs [67, 153, 179]. This section will introduce the general architectures of hierarchical mem-device reservoirs.

Figure 8.1 illustrates an example of hierarchical reservoirs. Each reservoir node (cluster node) is a memcapacitive network. Input signals $u(t)$ are applied to all cluster nodes through a scaling input matrix $W^{in}$, whose values are chosen from a set $\{-\kappa, \kappa\}$ according to a Bernoulli distribution. The cluster nodes are connected with random weights $w$ specified by the spectral radius $\lambda$ of the reservoir weight matrix $W^{res}$. It was suggested by Jaeger that $|\lambda| \leq 1$ is an essential condition to maintain the behavior of a single-layer reservoir [87]. When $|\lambda| > 1$, signals circulating within a network are amplified indefinitely, which leads to chaotic behavior. The condition of $\lambda$ is also applied to the cluster nodes within the hierarchical reservoirs. It has been shown that $\lambda$ could be greater than 1 for hierarchical memristive reservoirs due to signal loss caused by the resistance of
8.1. HIERARCHICAL RESERVOIR COMPUTING

Figure 8.1: Hierarchical topologies. (a) Hierarchical reservoir using delay-line topology; (b) hierarchical reservoir using delay-line with feedback topology; (c) hierarchical reservoir using simple cycle topology; (d) hierarchical reservoir using random topology. Each cluster is a SWPL network. (a), (b), and (c) are reprinted from [153].
memristive devices [30]. The output voltages at each reservoir node are forwarded to a readout layer for training with a linear regression algorithm. Only the readout layer of the hierarchical reservoirs is trained.

In the following sections, I will investigate the computing capacity of four topological structures of hierarchical memcapacitive reservoirs: Delay-Line (DL), Delay-Line with Feedback (DLFB), Simple Cycle (SC), and Random (R). The first three topological structures of hierarchical reservoirs are defined in [153]. For the random structure, cluster nodes are randomly connected within a hierarchical reservoir.

### 8.1.2 Hierarchical Reservoir Dynamics

The goal of this section is to quantify the enhanced dynamic response of hierarchical reservoirs by measuring the kernel quality of the reservoirs with an increasing number of devices within the reservoirs. Fifty different digits were chosen from the isolated spoken digit dataset as the inputs to reservoirs. Mem-device reservoirs were simulated with monolithic and hierarchical structures (Delay-line, Delay-line with feedback, Simple Circle, and Random). At each setting, fifty instances of similar reservoir settings were simulated to obtain averaged results.

Figure 8.2 summarized the simulation results. Four mem-device reservoirs (Biolek, Mohamed, Chang, and Oblea) showed a common trend in which the hierarchical reservoirs had a high kernel quality compared to those of the monolithic reservoirs with respect to the number of devices. For example, with a number of devices of 3,000, the kernel quality of the hierarchical reservoirs, on average, is better than the monolithic reservoirs by a factor of \(1.4 \times\). As Maass pointed out, the computing power of a reservoir is determined by its kernel quality to separate input patterns [120]. The simulation results confirmed that the hierarchical structures, indeed, enhanced the dynamic responses of the reservoirs.
Figure 8.2: Kernel quality of reservoirs (monolithic and hierarchical) with respect to the number of devices. Given the same number of devices, the hierarchical reservoirs have a better dynamics than the monolithic ones.
8.1.3 Performance and Power Consumption of Hierarchical Reservoirs

Monolithic and hierarchical reservoirs were trained and tested on three classification tasks: isolated spoken digits, MNIST images, and CIFAR-10 images. The color images from the CIFAR-10 dataset are represented in red-green-blue (RGB) formats, which are high-dimensional images. In order to reduce the image dimension, the training and testing color images were converted into grayscale images. Preprocessing color images into grayscale images is one of the common techniques for dimension reduction in training and testing neural networks [151]. Four topological structures – delay-line, delay-line with feedback, simply cycle, and random – were selected for hierarchical reservoirs. Due to the stochastic nature of the hierarchical reservoirs, fifty instances of reservoirs with similar network parameters were simulated to obtain average results. The chosen network parameters for reservoirs were obtained from optimizing processes using the particle swarm algorithm, as described in Section 6.3. The performance of the hierarchical reservoirs is based on the benchmarks (Isolated Spoken Digits, MNIST, and CIFAR-10) and compared to those of monolithic reservoirs.

Figure 8.3 summarizes the average results of four mem-device reservoirs (Biolek, Mohamed, Chang, and Oblea) on the three tasks. The low performance of the reservoirs on the CIFAR-10 task, compared to the state-of-the-art neural networks (about 94.03% [4]), was due to the dimension reduction of the grayscale images from the color images. Maintaining the original dimension of the color image would increase the performance of the reservoirs but also expands the size of the reservoirs that leads to longer simulation times.

In general, the hierarchical reservoirs were able to achieve comparable performance to those of the monolithic reservoirs for the three tasks. For the MNIST task, however, the hierarchical reservoirs gained similar performance to one of the monolithic reservoirs
Figure 8.3: Performance and power consumption of monolithic and hierarchical reservoirs on different tasks. The power measurements are normalized to the power measurements of the monolithic reservoirs for comparisons. For the CIFAR-10 image task (c), the training and testing color images from the dataset were converted into grayscale images to reduce their dimensions. The hierarchical reservoirs, on average, consumed less power compared to monolithic reservoirs for both isolated spoken digit and CIFAR-10 tasks while maintaining comparable performance. For the MNIST task, the hierarchical reservoirs were not able to compete with the monolithic reservoirs.
with much higher power consumption. I suspect that since the input data from the MNIST images are quite simple and sparse, 1’s and 0’s, the monolithic reservoirs had sufficient dynamics to translate the input data into their internal states whereas the hierarchical reservoirs did not improve the translation of the input data but required more devices, resulting in higher power consumption. For the other two complex tasks, isolated spoken digits and CIFAR-10 images, the hierarchical reservoirs yielded a large improvement in power reduction. In fact, for the isolated spoken digits, compared to the monolithic reservoirs, the hierarchical reservoirs, on average, reduced the power consumption without a loss in their performance by a factor of $1.4\times$, $8.8\times$, $9.5\times$, and $6.3\times$ with delay-line, delay-line feedback, simple cycle, and random structures, respectively. Similarly, for the CIFAR-10 image task, the hierarchical reservoirs reached to higher performance with less power, by a factor of $5.6\times$, $4.2\times$, $4.8\times$, and $1.9\times$ in comparison to the monolithic reservoirs. The results suggest that the hierarchical reservoirs had better dynamics than the monolithic reservoirs to solve sufficiently complex tasks.

### 8.1.4 Optimizing Parameters for Hierarchical Reservoirs

As Goudarzi and Teuscher pointed out, optimal network parameters depended on specific tasks and device characteristics [70]. In addition to four different topological structures, hierarchical reservoirs have a large number of network parameters, which is a challenge for finding optimal settings using a manual process. A genetic algorithm, namely the particle swarm algorithm, was employed for the optimization process. The particle swarm algorithm is described in Section 6.3. The results of this section will address research question 2.2 (See Section 2.2).

Since the initial states of memcapacitive devices within clusters were randomly selected, fifty instances of hierarchical reservoirs with similar settings were simulated to
### Table 8.1: Optimized Parameters for Spoken Digit Recognition Task

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>1.811</td>
<td>0.661</td>
<td>1.260</td>
<td>1.363</td>
<td>1.030</td>
<td>1.584</td>
</tr>
<tr>
<td>β</td>
<td>0.560</td>
<td>0.367</td>
<td>0.285</td>
<td>0.479</td>
<td>0.347</td>
<td>0.875</td>
</tr>
<tr>
<td>δ</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ν</td>
<td>0.051</td>
<td>0.216</td>
<td>0.016</td>
<td>0.121</td>
<td>0.047</td>
<td>0.116</td>
</tr>
<tr>
<td>Outputs †</td>
<td>154</td>
<td>107</td>
<td>170</td>
<td>75</td>
<td>148</td>
<td>106</td>
</tr>
<tr>
<td>Nodes ‡</td>
<td>114</td>
<td>20</td>
<td>51</td>
<td>103</td>
<td>34</td>
<td>41</td>
</tr>
<tr>
<td>θ</td>
<td>0.962</td>
<td>0.032</td>
<td>0.414</td>
<td>0.083</td>
<td>0.935</td>
<td>0.929</td>
</tr>
<tr>
<td>Ts (µs)</td>
<td>6.058</td>
<td>4.462</td>
<td>8.703</td>
<td>9.157</td>
<td>2.043</td>
<td>6.558</td>
</tr>
<tr>
<td>Clusters</td>
<td>114</td>
<td>80</td>
<td>132</td>
<td>123</td>
<td>139</td>
<td>118</td>
</tr>
<tr>
<td>κ</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>λ</td>
<td>0.428</td>
<td>0.932</td>
<td>0.876</td>
<td>1.484</td>
<td>1.205</td>
<td>1.169</td>
</tr>
</tbody>
</table>

† This is the number of outputs from the hierarchical reservoir to the readout layer.
‡ This is the number of nodes within each cluster.

The hierarchical reservoirs had 117 inputs; ν is the scaling factor for input vectors, θ is the decay factor, Ts is the time step for input vectors, κ is the scaling factor of the hierarchical reservoirs, and λ is the spectral radius.
obtain average results for each fine-tuning step. Four topological structures of hierarchical reservoirs were chosen for the experiments: delay-line (DL), delay-line with feedback (DLFB), simple cycle (SC), and random (R). The hierarchical reservoirs were composed of cluster nodes and each cluster node was an SWPL memcapacitive network. The connections between cluster nodes were determined by the spectral radius $\lambda$. The lower and upper limits of parameters were based on experimental results.

Table 8.3: Optimized Parameters for CIFAR-10 Image Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Delay-line</th>
<th>Delay-line Fb</th>
<th>Simple Cycle</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>1.254</td>
<td>1.994</td>
<td>2.118</td>
<td>2.186</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.692</td>
<td>0.713</td>
<td>0.788</td>
<td>0.542</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>2.708</td>
<td>38.074</td>
<td>24.538</td>
<td>41.635</td>
</tr>
<tr>
<td>$\delta$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\nu$</td>
<td>0.040</td>
<td>0.128</td>
<td>0.075</td>
<td>0.115</td>
</tr>
<tr>
<td>Outputs †</td>
<td>82</td>
<td>165</td>
<td>135</td>
<td>239</td>
</tr>
<tr>
<td>Nodes ‡</td>
<td>119</td>
<td>157</td>
<td>109</td>
<td>147</td>
</tr>
<tr>
<td>$\theta$</td>
<td>0.600</td>
<td>0.730</td>
<td>0.337</td>
<td>0.458</td>
</tr>
<tr>
<td>$Ts$ ($\mu$s)</td>
<td>1.754</td>
<td>3.000</td>
<td>5.226</td>
<td>5.543</td>
</tr>
<tr>
<td>Clusters</td>
<td>1212</td>
<td>530</td>
<td>1225</td>
<td>1276</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>1.269</td>
<td>0.447</td>
<td>0.863</td>
<td>1.021</td>
</tr>
</tbody>
</table>

† This is the number of outputs from the hierarchical reservoir to the readout layer.
‡ This is the number of nodes within each cluster.

The hierarchical reservoirs had 1024 inputs; $\nu$ is the scaling factor for input vectors, $\theta$ is the decay factor, $Ts$ is the time step for input vectors, $\kappa$ is the scaling factor of the hierarchical reservoirs, and $\lambda$ is the spectral radius.

For the isolated spoken digit task, the hierarchical reservoirs were trained with 1,000 digits and tested with 500 digits. For the CIFAR-10 image task, however, due to the high dimension of color images, $32 \times 32$ pixel images expressed in the color intensity format (R, G, B), only a subset of color images (1,000 images for training and 400 images for testing) was adopted for training and testing hierarchical reservoirs. Furthermore,
the color images were converted into grayscale images for dimension reduction. The implication is that the hierarchical reservoirs may yield a low performance, compared to one of the state-of-the-art neural networks. The results, however, provide evidence that hierarchical reservoirs can solve more power-sufficiently the same subset of complex tasks compared to monolithic reservoirs.

The optimizing parameters are shown in Tables 8.1 and 8.3. In the Tables, "Outputs" denotes the number of signals from the hierarchical reservoirs to their readout layers for training. "Nodes" are the electrical nodes of an SWPL memcapacitive network within a cluster node. The reservoirs had 117 inputs for the isolated spoken digit task and 1,024 inputs for the CIFAR-10 image task. The scaling input factor, \( \kappa \), was set to 1.0 since the input signals were already scaled by \( \nu \) to prevent an over-stimulated condition in the reservoirs. The spectral radius, \( \lambda \), which determined the connection weights between cluster nodes, had a fixed value during the training and testing processes. Tables 8.1 and 8.3 do not include the classification performance and the power consumption of the hierarchical reservoirs on the tasks since these measurements were already shown in Figure 8.3.

### 8.1.5 Mixed Mem-device Hierarchical Reservoirs

Since both memristor and memcapacitor are nonlinear devices, the networks of memristive and memcapacitive devices, in theory, have better dynamics as hierarchical reservoirs to characterize input streams. This section will try to evaluate how the combinations of memristive and memcapacitive devices enhance the reservoir response by measuring the kernel quality of the reservoirs. A set \( k \) of input sequences \( u^i(n) \), where \( i = 1, \cdots, k \), and \( n = 1, \cdots, T_k \), generates reservoir states \( x^i(n) \). The reservoir states are collected into a matrix \( M (M \in \mathbb{R}^{k \times N}) \). Each column of the matrix \( M \) is a vector of reservoir states \( x^i(n) \) corresponding to a particular input \( u(n) \). The rank \( r \) of the matrix \( M \) is the measurement of
8.1. HIERARCHICAL RESERVOIR COMPUTING

reservoirs’ computing power. A reservoir with a higher rank \( r \) has a better kernel quality for separating input patterns. The results of this section will address research question 2.3 (See Section 2.2).

Figure 8.4: Kernel quality measurements of mixed reservoirs with respect to the percentage of memristive devices. The dynamic response of the hierarchical reservoirs increases when the percentage of the memristive devices is above 50%.

Fifty different digits from the isolated spoken digit dataset were chosen as the input signals to measure the kernel quality of mixed-device reservoirs. The digit wave files were converted into 117-input vectors with their amplitudes scaled to a range \([-\nu, +\nu]\) to avoid an overstimulated condition in the reservoirs. Four combinations of memristive and mem-
8.1. HIERARCHICAL RESERVOIR COMPUTING

capacitive models were selected to form mixed-device reservoirs, namely Biolek-Chang, Biolek-Oblea, Mohamed-Chang, and Mohamed-Oblea. Hierarchical mixed-device reservoirs were constructed with four topological hierarchies: Delay-line (DL), Delay-line with feedback (DLFB), Simple cycle (SC), and Random (R). One hundred instances of the same reservoirs were simulated for a percentage sweep of memristive devices to obtain average measurements. The reservoirs were also trained and tested for image classifications. The network parameters are specified in Tables 8.5 and 8.7. At each setting of the memristive percentage, the measurements of classification, power consumption, and kernel quality were recorded.

Table 8.5: Parameters for Mixed-device Reservoirs on Spoken Digit Recognition Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Biolek-Chang</th>
<th></th>
<th></th>
<th></th>
<th>Biolek-Oblea</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DL</td>
<td>DLFB</td>
<td>SC</td>
<td>R</td>
<td>DL</td>
</tr>
<tr>
<td>α</td>
<td>0.255</td>
<td>1.965</td>
<td>2.979</td>
<td>2.555</td>
<td>1.578</td>
</tr>
<tr>
<td>β</td>
<td>0.233</td>
<td>0.503</td>
<td>0.353</td>
<td>0.044</td>
<td>0.703</td>
</tr>
<tr>
<td>γ</td>
<td>26.971</td>
<td>16.017</td>
<td>41.439</td>
<td>0.155</td>
<td>3.773</td>
</tr>
<tr>
<td>δ</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ν</td>
<td>0.533</td>
<td>0.012</td>
<td>0.068</td>
<td>0.441</td>
<td>0.023</td>
</tr>
<tr>
<td>Outputs †</td>
<td>177</td>
<td>196</td>
<td>180</td>
<td>93</td>
<td>182</td>
</tr>
<tr>
<td>Nodes ‡</td>
<td>41</td>
<td>60</td>
<td>46</td>
<td>93</td>
<td>20</td>
</tr>
<tr>
<td>θ</td>
<td>0.859</td>
<td>0.057</td>
<td>0.403</td>
<td>0.313</td>
<td>0.477</td>
</tr>
<tr>
<td>Ts (µs)</td>
<td>500.0</td>
<td>500.0</td>
<td>500.0</td>
<td>500.0</td>
<td>3.445</td>
</tr>
<tr>
<td>Clusters</td>
<td>112</td>
<td>160</td>
<td>161</td>
<td>141</td>
<td>107</td>
</tr>
<tr>
<td>κ</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>λ</td>
<td>1.397</td>
<td>1.169</td>
<td>1.236</td>
<td>0.659</td>
<td>0.635</td>
</tr>
</tbody>
</table>

† This is the number of outputs from the hierarchical reservoir to the readout layer.
‡ This is the number of nodes within each cluster.

The hierarchical reservoirs had 117 inputs; ν is the scaling factor for input vectors, θ is the decay factor, Ts is the time step for input vectors, κ is the scaling factor of the hierarchical reservoirs, and λ is the spectral radius.

As expected, the computing power of the Biolek-Chang and the Biolek-Olea topological reservoirs increased with a higher percentage of memristive devices present in the
reservoirs, specified by the increase of the kernel quality in Figures 8.4a and 8.4b. The increase of kernel quality, or computing power, is due to the fact that memristive and memcapacitive devices within the reservoirs formed RC filters, which, in turn, further improve the nonlinear dynamics of the reservoirs. The Mohamed-Chang and Mohamed-Oblea topological reservoirs, on the other hand, did not show a significant improvement in the computing power of the reservoirs, even with the presence of memristive devices within the reservoirs, shown in Figures 8.4c and 8.4d. A study in material computing, such as poly-butyl methacrylate and poly-methyl methacrylate composites, has shown that matching the response of the material to the processing time for a particular task would significantly improve the computing capacity of material-computing reservoirs [47]. In other words, the closeness between the time step processing input images and the response time of RC filters (or the time constant) would significantly increase the computing power of the hierarchical reservoirs. The time steps for the Mohamed-Chang and Mohamed-Oblea reservoirs were experimentally chosen so that the reservoirs were able to exhibit dynamic responses. These time-step parameters, however, may not be close to the response time of the memristive and memcapacitive devices. Consequently, the computing power of the Mohamed-Chang and Mohamed-Oblea reservoirs did not improve much with the level of memristive devices in the systems. Finding optimal settings for the time-steps is a subject of further research.

Table 8.9 summarizes the simulation results of the topological mixed-device reservoirs on the isolated spoken digit task. The reservoirs achieved their highest performance where their kernel quality measurements were at maximum. It is interesting to notice that the topological mixed-device reservoirs, in general, accomplished their highest computing powers with a high level of memristive percentage within the reservoirs. This phenomenon suggests that the nonlinear characteristics of memristive devices significantly
Table 8.7: Parameters for Mixed-device Reservoirs on Spoken Digit Recognition Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mohamed-Chang</th>
<th>Mohamed-Oblea</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DL</td>
<td>DLFB</td>
</tr>
<tr>
<td>α</td>
<td>2.838</td>
<td>2.838</td>
</tr>
<tr>
<td>β</td>
<td>0.149</td>
<td>0.149</td>
</tr>
<tr>
<td>γ</td>
<td>38.075</td>
<td>38.075</td>
</tr>
<tr>
<td>δ</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ν</td>
<td>0.119</td>
<td>0.119</td>
</tr>
<tr>
<td>Outputs †</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td>Nodes ‡</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>θ</td>
<td>0.869</td>
<td>0.869</td>
</tr>
<tr>
<td>Ts (µs)</td>
<td>500.0</td>
<td>500.0</td>
</tr>
<tr>
<td>Clusters</td>
<td>111</td>
<td>111</td>
</tr>
<tr>
<td>κ</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>λ</td>
<td>1.483</td>
<td>1.483</td>
</tr>
</tbody>
</table>

† This is the number of outputs from the hierarchical reservoir to the readout layer.
‡ This is the number of nodes within each cluster.

The hierarchical reservoirs had 117 inputs; ν is the scaling factor for input vectors, θ is the decay factor, Ts is the time step for input vectors, κ is the scaling factor of the hierarchical reservoirs, and λ is the spectral radius.

Table 8.9: Summary for Mixed-device Reservoirs on Spoken Digit Recognition Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mohamed-Chang Reservoirs</th>
<th>Mohamed-Oblea Reservoirs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DL</td>
<td>DLFB</td>
</tr>
<tr>
<td>Biolek</td>
<td>MR (%)</td>
<td>94.00</td>
</tr>
<tr>
<td></td>
<td>Perf. (%)</td>
<td>90.82</td>
</tr>
<tr>
<td></td>
<td>Power (µW)</td>
<td>2.83</td>
</tr>
<tr>
<td>Mohamed</td>
<td>MR (%)</td>
<td>98.00</td>
</tr>
<tr>
<td></td>
<td>Kernel</td>
<td>24.44</td>
</tr>
<tr>
<td></td>
<td>Perf. (%)</td>
<td>92.49</td>
</tr>
<tr>
<td></td>
<td>Power (µW)</td>
<td>0.22</td>
</tr>
</tbody>
</table>
8.2. DEEP RESERVOIR COMPUTING

contributed to the overall dynamic responses of the reservoirs. Having a high percentage of memristive devices with the reservoirs would potentially increase the power consumption of the system. Extended experiments of fine-tuning further the parameters of the hierarchical structures to match the physical property of mem-devices would improve both performance and power consumption of the systems. This is a subject of a further research.

8.2 DEEP RESERVOIR COMPUTING

Deep learning is a subset of machine learning that has networks capable of learning unsupervised from data that is unstructured or unlabeled [25]. In deep learning, the hierarchical structure of neural networks is based on the structure and feature representations of input data. Deep learning architectures such as deep neural networks [79], deep belief networks [130], and deep recurrent neural networks [72] have been applied to computer vision, speech recognition, natural language processing, and audio recognition.

Deep learning is a learning model of data representations at different levels of abstraction [61, 63]. One of the approaches to deep learning is to use each layer in a hierarchical structure to represent a particular structure of inputs such as object edges, object category, and color intensity in an image task. A deep learning network is often implemented as a feedforward hierarchy of multiple hidden layers of nonlinear units. Training such a network is a time-consuming operation due to the fact that the synaptic weights of each hidden layer must be updated. Here, a deep RC can be beneficial since synaptic weights of reservoirs in cluster nodes do not need to train. Can a deep memcapacitive RC architecture harness the temporal dynamics differentiation at the different layers for a computational task? Research question 2.4 explores a possible answer to the question
8.2. DEEP RESERVOIR COMPUTING

(See Section 2.2).

8.2.1 Deep Memcapacitive Reservoir Architecture

Although various network topologies have been proposed for deep learning networks [63, 72, 79, 130], none of these topologies are designed for mem-devices. Here, I want to introduce a deep mem-device reservoir using the proposed architecture in [63]. Figure 8.5 shows a general architecture of a deep mem-device RC.

Figure 8.5: Deep memcapacitive reservoir computing architecture. The architecture is organized in a stacked structure of \( N \) layers. Each layer is a small-world power-law reservoir. At each time step \( t \), the input signal \( u(t) \) stimulates the first reservoir layer and the output of each successive layer \( x(t)^n \), scaled by a fixed input vector \( W^{(n+1)} \), excites a higher one in the stack. The outputs of all layers \( \{y_1(t), y_2(t), \ldots, y_N(t)\} \) are collected as the input signal to the Readout node. \( W^{res(n)} \) is the internal weight of a memcapacitive reservoir layer. Only the Readout node is trained. Reprinted from [63].

The architecture is organized in a stacked structure of \( N \) memcapacitive reservoir
layers. Each layer is a small-world power-law reservoir. The input signal \( u(t) \) stimulates the first layer while the output \( x(t)^n \) of a successive layer \( n \), scaled by an input vector \( W^{(n+1)} \), provides an input signal to a higher layer in the stack. The input vectors \( (W^{in}, W^2, W^3, \ldots, W^N) \) are fixed vectors whose values are randomly selected from the set \([-\nu, +\nu]\). The outputs of all layers \( \{y_1(t), y_2(t), \ldots, y_N(t)\} \) are collected as the input signal to the Readout node. \( W^{res(n)} \) is the internal weight of the memcapacitive reservoir layer. Similar to a hierarchical RC, only the Readout node of the deep RC is trained by a simple algorithm such as the linear ridge regression technique.

At a time step \( t \), the outputs of a layer \( n \) \( (x(t)^n \) and \( y_n(t) \)) and the output of deep RC \( y(t) \) are defined as:

\[
\begin{align*}
    x(t)^1 &= f \left( W^{res(1)} \ast W^{in} \ast u(t) \right), \\
    y_1(t) &= f \left( W^{res(1)} \ast W^{in} \ast u(t) \right), \\
    x(t)^n &= f \left( W^{res(n)} \ast W^{(n)} \ast x(t)^{n-1} \right), \\
    y_n(t) &= f \left( W^{res(n)} \ast W^{(n)} \ast x(t)^{n-1} \right), \\
    y(t) &= W^{out} \cdot (y_1(t), y_2(t), y_3(t), \ldots, y_N(t)),
\end{align*}
\]

where \( f() \) is a transformation function, determined by node voltages of the memcapacitive reservoir layer \( n \).

### 8.2.2 Polyphonic Music Task

The polyphonic music task is a challenging benchmark in identifying next music notes from a history of music notes being played in the past. The task is complex, due to high-dimensional dependencies of concurrent music notes being played at different time intervals, and requires the ability of a system to retain previous timestep data in order to make a prediction. The polyphonic music task, defined in [26], is a common benchmark for
DeepESN [63]. Two music datasets, Piano-midi.de and MuseData, were considered as the polyphonic music task for deep memcapacitive reservoirs. These datasets are composed of high dimensionality and complex temporal dependencies of music notes at different time scales. They are a series of musical compositions, such as complex piano and orchestral combinations, with simultaneous music notes. The datasets are represented as piano-rolls that were preprocessed from MIDI files [26]. Each piano-rolls are divided into three subsets for training, validation, and testing. The characteristics of piano-roll samples of datasets are listed in Table 8.11.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Rolls</th>
<th>Samples</th>
<th>AvgLen</th>
<th>MinLen</th>
<th>MaxLen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Piano-midi.de</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td>87</td>
<td>872.5</td>
<td>111</td>
<td>3857</td>
<td></td>
</tr>
<tr>
<td>Validation</td>
<td>12</td>
<td>711.7</td>
<td>209</td>
<td>1637</td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>25</td>
<td>761.4</td>
<td>65</td>
<td>2645</td>
<td></td>
</tr>
<tr>
<td>MuseData</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td>524</td>
<td>476.9</td>
<td>9</td>
<td>3457</td>
<td></td>
</tr>
<tr>
<td>Validation</td>
<td>135</td>
<td>613.0</td>
<td>63</td>
<td>3722</td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>124</td>
<td>518.9</td>
<td>45</td>
<td>4273</td>
<td></td>
</tr>
</tbody>
</table>

AvgLen, MinLen, and MaxLen refer to the average, minimal, and maximal time frames in the piano rolls for training, validation, and testing.

As explained in [26], each dataset is a list of time steps, and each time step is a list of the non-zero elements in the piano-roll at this instant (in MIDI note numbers, between 21 and 108 inclusive). Both datasets were converted into sequences of 88-dimensional and 82-dimensional vectors of 1’s and 0’s as inputs for deep memcapacitive reservoirs. At each instant of a time step, the value of the dimensional vectors is set to 1 if the music note is played and to 0 otherwise.

The polyphonic music task is a prediction of the next step on high-dimensional vectors. In particular, it aims to predict the music notes played at time step $t + 1$ given the music notes played at time step $t$. The performance of deep memcapacitive reservoirs
was measured with the expected frame-level accuracy (FL-ACC), which is the evaluation model developed by Bay et al. to estimate the fundamental frequencies of all concurrent music notes within a polyphonic piece [14]. The FL-ACC is calculated as follows:

\[
FL - ACC = \frac{\sum_{t=1}^{T} TP(t)}{\sum_{t=1}^{T} TP(t) + \sum_{t=1}^{T} FP(t) + \sum_{t=1}^{T} FN(t)},
\]

where \( T \) is the total number of time steps for testing sequences, \( TP(t) \) is the number of true positive notes, \( FP(t) \) is the number of false positive notes, \( FN(t) \) is the number of false negative notes, and \( t \) is the current predicting time step.

### 8.2.3 Simulation Results

Four deep mem-device reservoirs – Biolek, Mohamed, Chang, and Oblea – were simulated for the polyphonic music task using two datasets: Piano-midi.de and MuseData. The performance of two additional networks, deepESN [63] and recurrent neural network restricted Boltzmann machine (RNN-RBM) [14], was considered as the reference to compare the performance of the deep mem-device reservoirs.

The genetic algorithm (Section 6.3) was employed to find optimal parameters for the highest measurement of FL-ACC. At each setting, fifty instances of deep mem-device reservoirs were trained and tested for the frame accuracy. The results were averaged for the measurements of FL-ACC and power consumption.

Table 8.13 contains the network parameters of the deep-mem reservoirs on the two music datasets. In Table 8.13, \( O \) refers to the number of outputs from the reservoirs to the Readout node for training, \( L \) is the number of reservoir layers, \( N \) is the number of nodes within each layer, and \( \lambda_r \) is the ridge parameter for the Readout node. The value of \( \lambda_r \) was from the simulation results in [63]. Each reservoir layer was an SWPL network with the
Figure 8.6: Performance and power consumption of deep mem-device reservoirs for the music note prediction task on two datasets: (a) piano-midi.de and (b) MuseData. In general, deep mem-device reservoirs achieved comparable performance to that of the RNN-RBM, except the Biolek deep reservoirs.

Table 8.13: Network Parameters of Deep-mem Reservoirs for the Music Task

<table>
<thead>
<tr>
<th></th>
<th>Piano-midi.de</th>
<th>MuseData</th>
<th></th>
<th>Piano-midi.de</th>
<th>MuseData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Biolek</td>
<td>Mohamed</td>
<td>Chang</td>
<td>Oblea</td>
<td>Biolek</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>2.534</td>
<td>0.288</td>
<td>2.153</td>
<td>2.692</td>
<td>0.200</td>
</tr>
<tr>
<td>(\beta)</td>
<td>0.692</td>
<td>0.297</td>
<td>0.323</td>
<td>0.093</td>
<td>0.174</td>
</tr>
<tr>
<td>(\delta)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(\nu)</td>
<td>1.430</td>
<td>1.522</td>
<td>1.155</td>
<td>0.758</td>
<td>1.908</td>
</tr>
<tr>
<td>O(^\dagger)</td>
<td>324</td>
<td>333</td>
<td>265</td>
<td>281</td>
<td>312</td>
</tr>
<tr>
<td>(\theta)</td>
<td>0.149</td>
<td>0.038</td>
<td>0.529</td>
<td>0.196</td>
<td>0.043</td>
</tr>
<tr>
<td>Ts</td>
<td>9.630</td>
<td>3.610</td>
<td>500.000</td>
<td>0.012</td>
<td>0.488</td>
</tr>
<tr>
<td>((\mu s))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L(^\ddagger)</td>
<td>11</td>
<td>14</td>
<td>14</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>N(\star)</td>
<td>225</td>
<td>219</td>
<td>216</td>
<td>217</td>
<td>219</td>
</tr>
<tr>
<td>(\lambda_r)</td>
<td>0.100</td>
<td>0.100</td>
<td>0.100</td>
<td>0.100</td>
<td>0.100</td>
</tr>
</tbody>
</table>

\(^\dagger\) This is the number of outputs from reservoir layers to the Readout node.

\(^\ddagger\) is the number of reservoir layers.

\(*\) This is the number of nodes within each reservoir layer.

The deep reservoirs had 88 inputs; \(\nu\) is the scaling factor for input vectors, \(\theta\) is the decay factor, Ts is the time step for input vectors, and \(\lambda_r\) is the ridge parameter for the Readout node. The ridge value was referred from the simulation results in [63].
same structure, specified by $\alpha, \beta, \gamma,$ and $\delta$.

Figure 8.6 illustrates the performance of the deep-mem reservoirs on the music datasets. Compared to the performance of the state-of-the-art networks (DeepESN and RNN-RBM), the performance of the deep-mem reservoirs was lower for the Piano-midi.de (Figure 8.6a) and MuseData datasets (Figure 8.6a). The results suggested that the performance of the deep-mem reservoirs strongly depends on the characteristics of the devices and the settings of the network parameters. The selected ranges of the network parameters were based on experimental results and may not be accurate to describe the physical properties of the mem-devices. According to the simulation results in [63], the state-of-the-art DeepESN achieved its optimal performance with thirty-five layers and thirty-six layers for the Piano-midi.de and MuseData datasets, respectively. Increasing the number of layers and finding sufficient settings of the network parameters tailoring to the physical properties of the devices in the deep-mem reservoirs could potentially increase their performance. This is a subject of further research. Nevertheless, the initial results provide evidence that the deep-mem reservoirs are capable of performing high-dimensional and complex tasks.
9

CONCLUSIONS AND FUTURE WORK

9.1 CONCLUSION

The motivation for my research stemmed from the growing possibilities of realizing emerging nanoelectronics, such as memcapacitive devices in neuromorphic hardware. A memcapacitor is a two terminal device with memory. It has similar behavior as a biological synapse. Being a storage device with low switching energy, memcapacitor is suitable for low power networks. The nonlinear property of the memcapacitor produces a rich dynamic response of random networks that are suitable for reservoir computing. My research offered new reservoir computing architectures that harnessed the computing nature of memcapacitive devices.

My research addressed three inherent challenges of reservoir computing (RC), as described in Chapter 2: the role structure of RC (Q1), the reservoir dynamics by a mixture of nonlinear devices (Q2), and the capability of solving complex tasks using hierarchical RCs (Q3). The extensive study of network dynamics in memcapacitive RCs (Chapter 7) addressed the first two challenges of RC whereas the results of network dynamics in hierarchical structures (Chapter 8) responded to the last question.
Network Dynamics in Memcapacitive Reservoir Computing

Fading memory is one of the essential properties of reservoirs in which similar input streams are mapped to the same reservoir state. The analysis of fading memory effect on Section 7.1 illustrated that memcapacitive reservoirs experienced the fading effect at the device level and at the system level. The simulation results of reservoir topological structures in Section 7.2 indicated that SWPL mem-device reservoirs accomplished equivalent performance, compared to the baseline ESN, but consumed much less power compared to the random and SW reservoirs (Figure 7.2) for the three tasks: the time series NARMA-10, the isolated spoken digits, and the MNIST image classification. On average, with the power consumption of the SWPL reservoirs as a reference, the SWPL reservoirs reduced significantly the power consumption by a factor of $1.21\times, 31\times$, and $31.2\times$ compared to the regular, the random, and the SW reservoirs, respectively.

Further analysis of SWPL structures (Section 7.3) showed that the combination of the locality $\alpha$ and the rewire probability or the randomness $\beta$ allowed different SWPL structures to emerge. The setting of high locality $\alpha$ and low randomness $\beta$ decreased the cost to the systems in terms of wiring cost and nanowire dissipated power but did not guarantee the optimal performance in respect of the correct classification, the number of devices within reservoirs, and the overall power consumption of systems.

Finding the appropriate settings of network parameters to ensure optimal performance of SWPL reservoirs is an important task. A genetic algorithm was employed to refine the network parameters (Section 7.4). The results illustrated that SWPL reservoirs with optimal network parameters consumed less power while maintaining equivalent performance. Compared to the regular reservoirs, the SWPL reservoirs consumed less power, by a factor of $1.3\times, 1.4\times$, and $1.5\times$. Similarly, compared to the random topology, the SWPL
reservoirs saved the power consumption by a factor of $4.8\times$, $1.6\times$, and $2.1\times$, respectively.

Since both memristor and memcapacitor are nonlinear devices, the combination of memristive and memcapacitive devices could potentially enhance the dynamic response of SWPL reservoirs (Section 8.1.5). Table 7.17 summarized the results of mixed-device reservoirs on NARMA-10, Isolated Spoken Digit, and MNIST tasks.

Device faults and variations are the inevitable problems in any system, including mem-device reservoir networks. Both faults and variations affected the performance of mem-device reservoirs. Section 7.7 studied how mem-device reservoirs responded to stuck-open faults, stuck-short faults, and device variations. The mem-device reservoirs were able to accommodate up to 40% of open faults without losing their performance. However, for the short faults, the mem-device reservoirs suffered tremendously the effect of the faults in their performance, even at a small percentage (2%) of the short faults. Adding Gaussian white noise is a common technique to analyze the effect of device variations on the performance of a system. The simulation results on the isolated spoken digit and the MNIST image tasks inferred that the performance of the mem-device reservoirs was task-dependent. The technique of adding Gaussian white noise did not quantify accurately the effect of the device variations but demonstrated how the mem-device functioned under the presence of the device variations.

**Network Dynamics in Hierarchical Structures**

The results in Section 8.1.3 underlined that the hierarchical reservoirs have better dynamics than the monolithic reservoirs and, therefore, were able to solve sufficiently the two complex tasks: the isolated spoken digits and the CIFAR-10 images. For the isolated spoken digits, compared to the monolithic reservoirs, the hierarchical reservoirs, on average, reduced the power consumption without a loss in their performance by a factor of
9.2. LIST OF CONTRIBUTIONS

\[ 1.4 \times, 8.8 \times, 9.5 \times, \text{ and } 6.3 \times \] with delay-line, delay-line feedback, simple cycle, and random structures, respectively. Similarly, the hierarchical reservoirs reached to higher performance for the CIFAR-10 image tasks with less power, by a factor of \[ 5.6 \times, 4.2 \times, 4.8 \times, \text{ and } 1.9 \times. \] The results suggest that the hierarchical reservoirs have better dynamics than the monolithic reservoirs to solve sufficiently complex tasks.

The measurements of kernel quality of hierarchical reservoirs in Section 8.1.5 provided evidence that combining nonlinear devices, such as memristive and memcapacitive devices, increased the dynamic response of the reservoirs. The reservoirs accomplished their highest performance in which the measurement of their kernel quality was at maximum.

Although the performance of the deep mem-device reservoirs was low compared to the state-of-the-art networks (DeepESN and RNN-RBM), it offered initial results that deep reservoirs using mem-devices were able to solve a high-dimensional and complex task such as polyphonic music task. The initial results of deep mem-device reservoirs in this research serve as a starting point where the performance of deep mem-device reservoirs can be improved further with better settings of network parameters and architectures.

9.2 LIST OF CONTRIBUTIONS

General

- My research has addressed three inherent challenges of reservoir computing (RC): the role reservoir structure of RC, the reservoir dynamics by a mixture of nonlinear devices, and the capability of solving complex tasks using hierarchical RCs.

- Although applications of memcapacitive devices have been presented from a theoretical perspective, my research is an example of realizing the computational nature
of memcapacitive devices in architectural and algorithmic considerations for neuromorphic tasks, such as NARMA-10, Isolated Spoken Digits, MNIST, and CIFAR-10.

- My research has explored hardware architectures (random and hierarchical) to perform tasks efficiently by exploring inherent device properties of low-power computing nature of memcapacitors.

- Utilizing the computational ability of memcapacitive devices has allowed my research to shed light on unconventional hardware architectures with power efficiency beyond that of conventional architectures.

**Memcapacitive Network Dynamics in Reservoir Computing**

- My research shows how the topological structure of reservoirs reduces the power consumption of the systems without sacrificing their performance.

- My research studies the dynamic response of memcapacitive small-world power-law reservoirs using separation matrix, kernel quality, and Lyapunov exponent as the reservoirs undergo topological changes.

- My research presents memcapacitive small-world power-law reservoir with optimal settings of network parameters to achieve high classification performance with low power consumption.

- My research demonstrates reservoir topologies and neuromorphic tasks for which a particular reservoir topology is suitable.
• My research demonstrates that a model of combining nonlinear devices (memristive and memcapacitive) improves reservoir dynamics and therefore reservoir performance.

• My research illustrates how device faults and variations influence the performance of memcapacitive reservoirs.

**Dynamics in Hierarchical Reservoir Computing**

• My research quantifies how hierarchical reservoirs have better dynamic behavior than monolithic reservoirs with the measurement of kernel quality.

• My research demonstrates that hierarchical reservoir computing is beneficial for solving more complex tasks.

• My research shows that mixed-device reservoirs (memristive and memcapacitive) allow performance improvements of a hierarchical system.

• My research exhibits that deep memcapacitive reservoirs are able to achieve comparable performance to the performance of the recurrent neural network restricted Boltzmann machine in polyphonic music task.

**Broader Impacts and Applications**

• My research is relevant for building emerging, adaptive, and power-efficient neuromorphic hardware with direct applications in embedded systems, the Internet-of-Things, and robotics.
• My research can be extended for applications in therapeutic systems such as the
diagnosis of neuromuscular disease, analyses of ECG/EKG signals, and prediction
of epileptic seizures.

9.3 FUTURE WORK

The spiking input signals in Section 7.6 required a Spike Converter to translate spatial
stimuli into spike train inputs to reservoirs and a Spike Accumulator to integrate spike
outputs to analog values for training and testing the Readout layer. In order to compare
the same images for both analog and spiking reservoirs, a Poisson Spike algorithm was
used to convert spatial stimuli into spike inputs. However, it is possible to implement
other algorithm or technique such as spike-time dependent plasticity (STDP) mechanism
for memcapacitive reservoirs. The complexity of implementing STDP for memcapacitive
reservoirs is a subject of further research.

In Section 8.2.3, the performance of the deep mem-device reservoirs was low com-
pared to the performance of the state-of-the-art networks. Finding sufficient settings of
network parameters matching the physical properties of the devices in question has the
potential to improve further the performance of the deep mem-device reservoirs.

Although the applications for the proposed architecture in my research are limited
to NARMA-10, Isolated Spoken Digits, MNIST, and CIFAR-10, extending the proposed
architecture of memcapacitive devices is possible for applications in therapeutic systems
such as the diagnosis of neuromuscular disease, analyses of ECG/EKG signals, and pre-
diction of epileptic seizures.
9.4 LIST OF PUBLICATIONS


Bibliography


[155] S. Sarma, B. M. Mothudi, and M. S. Dhlamini. Observed coexistence of memristive, memcapacitive and meminductive characteristics in polyvinyl alco-


