Memristive Silicon Oxide Devices for Biorealistic Learning

Viet Cuong Vu

Principal Supervisor: Anthony J. Kenyon

Subsidiary Supervisor: Adnan Mehonic

Department of Electronics and Electrical Engineering
University College London

This dissertation is submitted in fulfilment for the degree of
Doctor of Philosophy

April 2025

Declaration

I, Viet Cuong Vu, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Viet Cuong Vu
April 2025

Abstract

latex.

Impact Statement

latex.

Contributions

List of Publications

- V. C. Vu, A. Kenyon, D. Joksas, A. Mehonic, D. J. Mannion, and W. H. Ng, “Spiking Neural Networks with Nonidealities from Memristive Silicon Oxide Devices,” in 2024 IEEE 24th International Conference on Nanotechnology (NANO), Jul. 2024, pp. 46–50.
- D. J. Mannion, V. C. Vu, W. H. Ng, A. Mehonic, and A. J. Kenyon, “Unipolar Potentiation and Depression in Memristive Devices Utilizing the Subthreshold Regime,” IEEE Transactions on Nanotechnology, vol. 22, pp. 313–320, 2023.

Conference Presentations

- "Unipolar Potentiation and Depression within Optically Active Memristive Devices Subthreshold Regime", EMRS Spring 2025.
- "Circuit-Based Modelling of Current Transients within the Memristive Devices Subthreshold Regime", MEMRISYS 2024.
- "A Compact SPICE Model for Current Transients within the Subthreshold Regime of Memristors", IEEE MetroXRAINE 2023.

Acknowledgements

latex.

Table of contents

List of figures

List of tables

Chapter 1

Background

1.1 Research Hypothesis

This chapter presents an introduction to neuromorphic engineering and its fundamental concepts. These concepts are categorised into different groups and analysed with more detail in relation to the overall study. In the context of this thesis, each topic's fundamental ideas are concisely explained.

1.2 Device Material Characteristics

Extensive research has been conducted in device physics and material science to explore innovative materials and techniques for memories and prolonged retention objectives [137]. The term "neuromorphic" was created by researchers to describe new technologies and systems that, in addition to being essential for the construction of massive AI computer networks, exhibit certain behaviours that can be compared to those of real synapses [62].

Soon after, the notion of using these novel nanoscale components as "memristors" gained popularity, with the underlying notion being that they could be utilised to produce synapses in deep neural networks and sustain their synaptic weights locally [144]. The hardware and technology described could enable neural networks to perform "in-memory computing" and exhibit advanced non-linear properties, mimicking the physics of biological synapses [299].

The research in this field aims to develop various types of volatile and non-volatile memristive electronics. Additionally, spike or pulse-based control systems are being created to elicit biologically realistic learning behaviours in memristive cross-bar arrays. The challenging task

is to find the perfect artificial synapse, which requires investigation into different materials, tools, and techniques.

1.2.1 Theoretical Foundations

The presence of symmetry in nature, which is believed to arise from a common origin, is remarkable. However, the traditional electromagnetic passive circuit components of resistor, capacitor, and inductor are inadequate for describing the characteristics connected by the symmetry of circuit theory. Leon Chua addressed this issue by introducing the concept of a memristor in 1971 [49], which couples flux linkage and charge as a circuit device. However, proof of resistive switching in the memristor model was not established until 2008 [327].

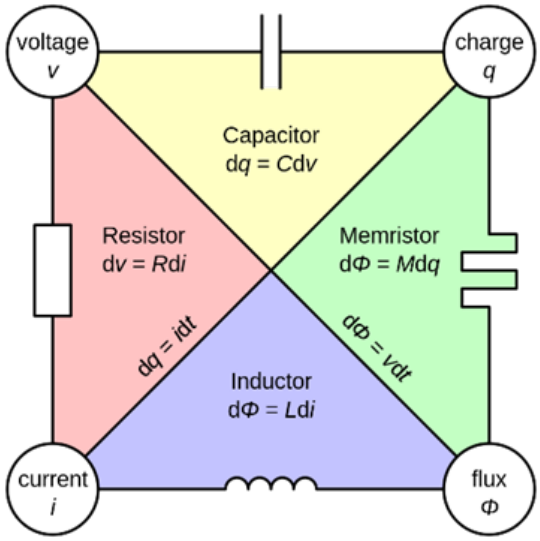

Fig. 1.1 Conceptual symmetries of resistor, capacitor, inductor, and memristor [66]. These four fundamental variables in circuit theory are depicted with their relationships. Each variable can be related to another via either a passive component or a well-known equation.

There are three fundamental circuit elements and four essential circuit variables in basic electrical circuit theory. It is evident that one component is absent to achieve symmetry. This device ought to function in such a way that charge and magnetic flux are interconnected, as illustrated in Figure 1.1. The link between the mathematical memristive model and a two-terminal resistive switching device is pivotal in this instance.

The concept of memristance differs from that of resistance in that it is dependent on charge, rather than being a constant value. As current is the amount of charge flowing per unit

time, we can express it as $q = \int_{-\infty}^{t_0} i(t)dt$, where charge is the sum of current at a given time t_0. This means that the memristance, being dependent on charge, is determined by the historical currents that have passed through the device. When the current flowing through the device is interrupted, the memory state persists until current flow is restored. The device is undoubtedly equipped with a type of memory known as a "memristor".

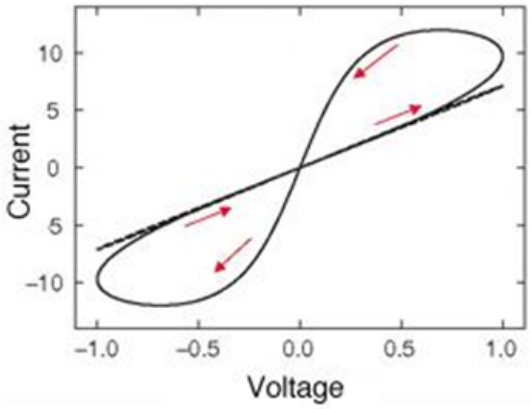

Fig. 1.2 The typical I-V characteristic of a memristor displays a pinched hysteresis loop resulting from the nonlinear relationship between current and voltage in memristance [375].

The pinched hysteresis loop, which is characteristic and dependent on frequency, distinguishes these memristor devices from other components [50], as shown in Figure 1.2. This loop is a common, natural phenomenon. As the voltage input frequency increases, the loop decreases in size. When the frequency is close to infinity, the memristor can be approximated as a resistor.

1.2.2 Material Properties

Among the range of new non-volatile memory devices, the primary focus of this study is on memristor devices, including MRAM, PRAM, FeRAM, and RRAM [368]. Resistive switching, a reversible phenomenon of two-terminal elements, characterises the devices. Through electrical signalling, they change resistance in a non-volatile manner, with the process driving the resistive switching defined by the device's materials [243].

Resistive random-access memory (RRAM) is a device that uses resistance switching, where reversibility is attained through repeated application of appropriate stimuli, according to [211]. Repeated application of suitable stimuli ensures reversibility. An RRAM cell comprises an insulating thin film (usually a metal oxide), sandwiched between two electrodes,

within which resistance switching occurs.

The term "memristance" is favoured to express the general characteristics of these RRAM devices. The central hypothesis of this model is that memristance is a function of the total charge that has been passed through the device or that the integral of the applied voltage is consistent with certain experimental data. This can be used to toggle between different resistance levels. Although this ideal memristor model is often used in RRAM cells, it may not satisfy practical requirements.

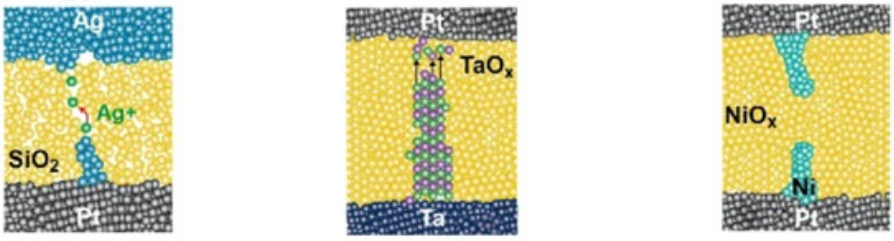

Fig. 1.3 The main RRAM types, from left to right: electrochemical metallization memory (ECM), vacancy change memory (VCM), and thermochemical memory (TCM). [100]

In normal operation, the state of the memristor can be objectively designated as having high resistance in the "OFF" state and low resistance in the "ON" state, with a substantial difference in resistance levels. The shift from the high resistance state (HRS) to the low resistance state (LRS) is referred to as "Set," while the reverse is called "Reset." An electroforming step is generally necessary to convert the device from pristine to switchable, whereby the former tends to exhibit higher resistance.

Although all RRAM devices operate on a metal-insulator-metal (MIM) architecture, their categorisation and analysis remain challenging. RRAM devices are loosely classified into two types based on their functional mechanisms: oxide-RAM (OxRAM) and conductive bridge RAM (CBRAM) [209]. However, the internal physical behaviour of RRAM devices greatly varies, making it difficult to obtain a unified picture of them. RRAM cells may be classified according to their switching mechanisms which are electrochemical metallisation (ECM), valence change mechanism (VCM), or thermochemical mechanism (TCM) [100]. The operation of the device is explained by one of the mechanisms depicted in Figure 1.3.

ECM: The process of electrochemical metallisation (ECM) refers to the formation of a metallic conducting bridge in the electrolyte that connects two metallic electrodes. Generally, these electrodes comprise a conducting metal, like *Ag* or *Cu*, and the other is a chemically inert metal, like *Pt* or *TiN* [370]. The active layer typically consists of an ion conductor, solid electrolyte, or bulk isolator. The key factor driving these devices is the formation and breakage of *Ag* or *Cu* filaments in the active layer. During electroforming and "Set" processes, a positive bias is applied to the device, inducing anodic oxidation. Metallic cations originate from the active electrode and propagate through the insulating layer in the presence of an electric field. When the positive ions arrive at the inactive electrode, they undergo chemical reduction, leading to the production of filaments. If the electric field is strong enough to cause temperature-assisted atom diffusion and significant current-induced Joule heating, the filament is broken during the "Reset" phase.

VCM: The valence change mechanism (VCM) describes the migration of oxygen vacancies, allowing carriers to tunnel between electrodes. VCM cells accommodate metallic electrodes that can be symmetrical or asymmetrical, with a metal oxide used as the insulating layer. VCM cells accommodate metallic electrodes that can be symmetrical or asymmetrical, with a metal oxide used as the insulating layer. Examples of such metal oxides include $TiO_x, SrTiO_x, TaO_x$, and HfO_x [188]. In this instance, the active electrode does not provide diffused metallic ions to create conducting filaments in the insulating layer. Instead, it is likely that oxygen vacancy migration is the primary switching mechanism for VCM devices. The device's symmetry is established following the initial forming step to form an n-type conducting channel located near the active electrode. Often, a tunnelling barrier exists between the electrode and the conductive channel. Fluctuations in the barrier height due to oxygen ion flow result in the production of different resistance states. During the "Set" procedure, oxygen ions are removed from this region. In contrast to the increased thermionic and tunnelling currents, the barrier height is decreased. These ions return to the vacant positions during the "Reset" process.

TCM: Thermochemical memory (TCM) theory proposes that metal oxides with lower valencies are energetically preferred at higher temperatures. This energy difference induces the creation of a metal filament. At higher temperatures, the oxide states with lower valencies have lower energy and are therefore favored. TCM devices have an insulating layer that can be made of NiO or TiO_x[373]. TCM devices typically operate with local stoichiometry that is dependent on the temperature gradient. This results in the movement of oxygen ions and vacancies along the temperature gradient, altering the switching position along the

conductive pathway. The creation of conductive filaments can be altered irreversibly due to an increase in local temperature, leading to thermionic breakdown at high current levels. The formation of the filament is determined by the metal oxide. The injection of electrons at the cathode causes filament growth in n-type oxides, which subsequently progresses to the anode. Meanwhile, p-type oxides depend on anode contact hole injection, resulting in filament development in the opposite direction [167].

This study's devices and samples will employ silicon dioxide SiO_x [240] as their primary insulating layer material, in addition to electrode materials like Ag or Cu. Since silicon-rich silica is commonly utilised as an insulating layer, it holds significant promise for CMOS-compatible processing. To gain a better understanding of silicon oxide's characteristics, an electron injection model was developed [92].

Within the amorphous structure of silicon oxide, $O-Si-O$ bonds are present, some of which possess wide-angle bonds that can function as profound electron traps, able to capture two electrons. The $Si-O$ bond is subsequently weakened once the broad bonds have captured both electrons, which lessens the energy requirement to break the connection and produce the Frenkel defect. The creation of these imperfections leads to the formation of a group of voids, which can subsequently aid in the creation of the conductive filament.

In bulk silicon oxide, a conductive filament can form within the insulating layer during electroforming. The switching process is typically controlled by a single conductive filament, which is not affected by the electrode size. The switching action leads to minor modifications to the filament. Since the resistance change usually takes place within a limited area, it is unrelated to the thickness of the insulating layer.

Contrastly, the filamentary switching, also called surface switching mechanism, is relatively unfamiliar. The electrode size significantly determines its conductivity. This mechanism's operation primarily depends on Schottky tunnel barrier formation over the entire electrode contact and insulating layer, which produces a switching layer at the interface.

1.2.3 Conduction Mechanisms

How well a substance resists the flow of electrical charge is determined by its electrical conductivity, an inherent characteristic. An ideal insulator would have infinite resistance and zero conductivity. However, a thin layer of silicon oxide that is just a few hundred

nanometres thick has limited conductance and fall short of ideal electrical resistivity.

A range of external factors have the potential to affect the conductivity of the semiconductor material. These include the electric field applied, sensitivity to temperature and light frequencies. The electric field strength, E, may be modelled by utilising the applied voltage, V, and distance across which the voltage is applied, d, giving the equation $E = \frac{V}{d}$. It should be noted, however, that this basic approximation may not be applicable to actual electronics.

Table 1.1 Basic conduction mechanisms for insulators [332].

Process	Expression	Voltage & Temperature
Conventional Tunnelling	$J \propto E^2 exp\left[-\frac{4\sqrt{2m^*}(q\phi_B)^{3/2}}{3q\hbar E_i}\right]$	$J \propto V^2 exp\left(-\frac{b}{V}\right)$
Fowler-Nordheim Tunnelling	$J = \frac{q^2E^2}{8\pi\hbar\phi_B} exp\left[-\frac{4\sqrt{2m^*}(q\phi_B)^{3/2}}{3q\hbar E_i}\right]$	$J \propto \frac{4\pi q m^* kT}{\hbar^3}$
Thermionic Emission	$J = A^{**}T^2 exp\left[-\frac{q(\phi_B - \sqrt{qE_i/4\pi\varepsilon_i})}{kT}\right]$	$J \propto exp\left[\frac{q}{Kt}\left(a\sqrt{V} - \phi_B\right)\right]$
Poole-Frenkel Emission	$J \propto E_i exp\left[-\frac{q(\phi_B - \sqrt{qE_i/\pi\varepsilon_i})}{kT}\right]$	$J \propto V exp\left[\frac{q}{kT}\left(2a\sqrt{V} - \phi_B\right)\right]$
Ohmic	$J \propto E_i exp\left(-\frac{\Delta E_{ac}}{kT}\right)$	$J \propto exp\left(-\frac{c}{T}\right)$
Ionic Conductance	$J \propto \frac{E_i}{T} exp\left(-\frac{\Delta E_{ac}}{kT}\right)$	$J \propto \frac{V}{T} exp\left(-\frac{d'}{T}\right)$
Space-Charge-Limited-Current	$J = \frac{9\varepsilon_i \mu V^2}{8d^3}$	$J \propto V^2$

Tunnelling: Conventional tunnelling is the primary mode of conduction for insulating materials in high electric fields. Tunnelling, a quantum mechanical phenomenon, occurs due to the finite probability of the electron wave function passing through a potential barrier with limited height. Conventional quantum tunnelling, on the other hand, refers to the instantaneous tunnelling of electrons across the entire width of the barrier. Table 1.1 illustrates the relationship between tunnelling current density and voltage as well as the dependence on electric field, while remaining independent of temperature. The variables used in the formula are ϕ_B for tunnelling barrier height, E for the insulator electric field, $m^* = 0.42m$ for the carrier effective mass for silicon oxide, $\hbar$ for the reduced Planck constant, q for electric charge, and b for a constant factor of proportionality. Where technical terms are first used, their abbreviations are explained.

Fowler-Nordheim Tunnelling: Electron tunnelling through only a portion of the barrier height is recognised as Fowler-Nordheim tunnelling. The mechanism of Fowler-Nordheim tunnelling is determined by the trapezoidal shape of the potential barrier. The application

of a strong electric field results in more band-bending. As a result, the effective breadth through which carriers must tunnel is exponentially reduced. Fowler-Nordheim tunnelling is the primary conduction mechanism for metal oxide systems with thick oxide layers. After tunnelling through, the carriers are able to move freely between the conduction and valence bands. The straightforward requirement for tunnelling in this instance is that the electric field multiplied by the thickness of the layer exceeds the barrier height.

Thermionic Emission: In general, thermionic emission is formed using a metal-to-insulator junction instead of a P-N semiconductor junction. This often leads to a negligible forward voltage drop and a swift switching action. It is important to ensure a clean surface for close contact between the metal and semiconductor during production. The fundamental relationships for thermally induced current are presented in Table 1.1. Where A^{**} represents the effective Richardson constant, ε denotes the permittivity of the insulator, and a serves as a proportionality constant, the current is generated by the movement of charge carriers, which may be either electrons or ions, across a potential barrier that has been thermally excited. In order for such movement to occur, the carriers' thermal energy must exceed the material's work function. The current density's magnitude is directly proportional to the temperature.

Poole-Frenkel Emission: Conduction may take place regardless of explicit quantum tunnelling through the insulator. In materials with a high density of structural defects, carriers are unable to travel as they would in tunnelling mechanisms. Owing to the presence of these structural flaws, additional energy states, or traps, emerge around the energy band margins. These traps limit the flow of electricity through a capture and release process. Electrons trapped in these states are dealt with by the Poole-Frenkel conduction mechanism. These trapped electrons can eventually accumulate sufficient energy to escape from isolated trap states due to thermal fluctuations in the material. If the electrons are not captured in another trap state, they will eventually reach the conduction band. Therefore, this mechanism is affected by both the applied electric field and the thermal energy. The electron's total energy arises from a combination of electric fields and temperature fluctuations. The electron's conduction process is primarily propelled by the electron drift current, $J = qn\mu E$, in which q refers to the electric charge, n represents the carrier density, μ denotes the carrier mobility, and E represents the electric field. The influence of trap depth ϕ_B, insulator permittivity ε, and temperature T on this current can be demonstrated. To accommodate the fabrication approach and semiconductor materials utilized, an non-ideal factor m that ranges between 1 and 2 may be incorporated in the equation.

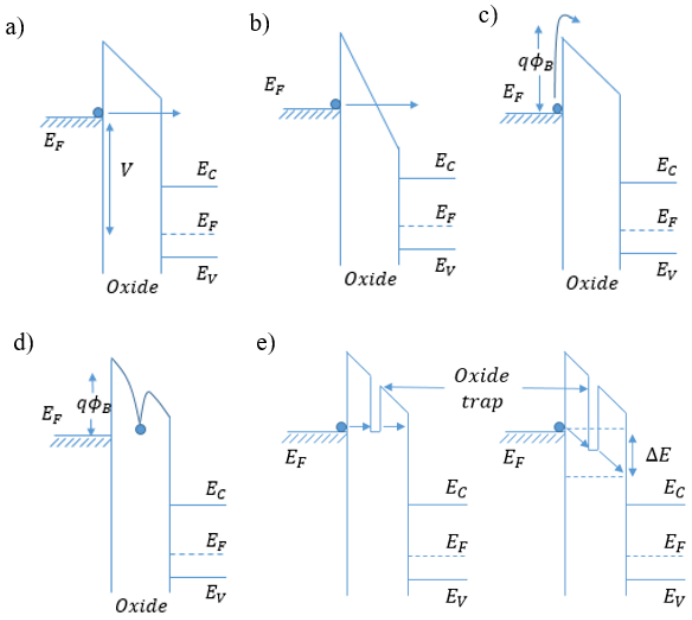

Fig. 1.4 Energy-band diagrams showing different conduction mechanisms: (a) direct tunnelling, (b) Fowler-Nordheim tunnelling, (c) thermionic emission, (d) Poole-Frenkel emission, (e) trap-assisted tunnelling [332]

Other conduction mechanisms comprise Ohmic conduction, where the current density depends on the electric field, voltage, and temperature. In Ohmic conduction, ΔE_{ac} represents the activation energy, k signifies the Boltzmann constant, T denotes the temperature in Kelvin, and c is a constant of proportionality. This type of conduction dominates at high temperatures and low fields since carriers change between conductive states due to thermal energy. Although ionic conduction has its own activation energy and proportionality constant, it shows comparable behaviour to ohmic conduction. This phenomenon can be described as ions moving through a substance via imperfections in the crystal lattice of a solid.

A flawless insulator hinders the mere ingress and egress of ions. Nonetheless, at the interfaces of metal-insulator, ionic carriers accumulate and alter the voltage distribution of the region under the influence of an electric field. Upon removing the applied electric field, a significant internal field remains, permitting the flow of an ionic current until an equilibrium state is reached. Finally, space-charge-limited current may arise from injected charge from the electrodes extending deeper into the insulator without compensating charges.

Charges are introduced to the dielectric material from one electrode and are subsequently removed by the other. The Mott-Gurney law is defined as $J = \frac{9\varepsilon\mu V^2}{8L^3}$ and $J = \frac{2\varepsilon vV}{L^2}$ for space charge limited current in solid and in the velocity-saturation regime, respectively. In this

context, ε represents the dielectric permittivity, μ indicates the carrier mobility, L denotes the material thickness, and $v = \mu E$ represents the electron drift velocity. With no inherent conductivity and a zero electric field at the cathode injecting charge, this mode of conduction necessitates the presence of a single type of charge carrier.

The tunneling processes described in the table above are based on a single-step process. However, defects within the insulating layer may allow for two or more tunneling phases, as shown in Figure 1.4 (e). These structural flaws or traps may be generated during production or under stress from a high electric field. Consequently, the energy barrier may be fragmented in multiple ways if traps exist, thus increasing the likelihood of carriers tunneling through progressively thinner barriers.

Elastic or inelastic trap-assisted tunnelling may occur, either with or without energy dissipation in the carriers. In materials with a high density of structural imperfections, conduction is more probable in the presence of multiple oxide traps during the tunnelling process. A simpler equation linking tunnelling current density with trap barrier height has been validated at high electric fields for the case of trap-assisted tunnelling [125].

1.2.4 Device Behaviours

After examining the physics that underlie the materials and mechanisms composing new RRAM devices, it is helpful to evaluate the general device behaviours concerning $I-V$ characteristics, volatility, polarity dependence, and power consumption to enhance application design. RRAM devices can respond in three ways [201].

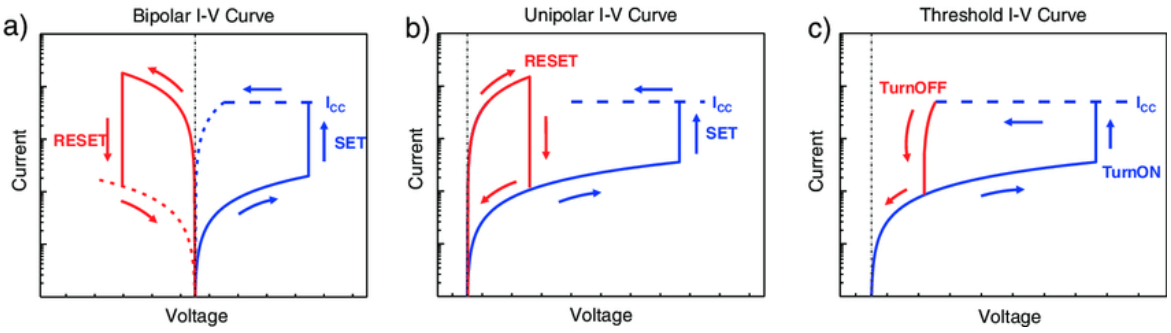

Fig. 1.5 Schematic I-V curves in: a) non-volatile bipolar memory switching mode, b) non-volatile unipolar memory switching mode; and c) volatile threshold switching mode. In the SET process, a compliance current is required to avoid hard breakdown [201].

They are classified as unipolar if they are set and reset with the same polarity. When set and reset are accomplished with opposing polarities, the device is said to be bipolar. Moreover, threshold switching occurs when a device transitions to its low resistance state within a certain voltage range. Additionally, the term nonpolar designates devices that are polarity-independent, and thus, may perform both unipolar and bipolar operations. Figure 1.5 summarises these three categories.

Bipolar Switching: Bipolar switching entails configuring and resetting devices in polarities that oppose one another. The said behaviour has been noted in several oxide materials [374]. Current compliance is necessary for the setup of the switching mechanism, but the same applied current compliance can be used to reset it. The compliance system is notably simpler than that of the unipolar switching strategy. The switching voltage magnitude for bipolar devices falls within typical limits for digital chip integration. For instance, operational voltages can be below $\pm 1V$ in either polarity [245].

Unipolar Switching: Unipolar switching, similar to bipolar switching, has been shown to occur in various oxide materials [141]. In unipolar operation, electroforming and setting are driven by the electric field, whereas resetting is usually driven by current. To avoid competition between the two operating systems, a current compliance must be enforced during the setting process, as resetting is initiated by excessive currents. Once the device is configured, the current compliance can be augmented or eliminated entirely to enable adequate current flow and reset the device. The unipolar device's ability to operate with a single supply rail makes it compatible with digital integrated circuits. A significant drawback of these devices is the complex compliance system that is required, as the high reset current can generate heating and power usage issues, especially when scaling large memory arrays [402].

Threshold Switching: Threshold switching happens when a device changes to a low resistance state once the applied voltage reaches a threshold and quickly returns to the previous state when the voltage falls below a lower threshold, displaying hysteresis [2]. This occurrence was suggested as a consequence of thermal dissipation after assessing the diverse thicknesses of the bottom electrode [37]. This suggests that altering the electrode dimensions in the manufacturing process may result in different intended threshold switching behaviours.

Aside from the processes of switching, multiple aspects must be considered during the development of RRAM. Firstly, OxRAM samples typically exhibit a lower on/off conductance

ratio within the range of 10s-100s and provide strong retention for up to 10^{12} cycles [135]. On the other hand, CBRAM offers a relatively high on/off conductance ratio within the range of 10^3-10^6, however, it has limited endurance to less than 10^4 cycles [4].

The stochastic behaviour of oxygen or metal ions during ionic migration and the variability in filament form from device to device and cycle to cycle within a single device are unpredictable parameters that pose a significant challenge for the design of RRAM cells [5]. Due to these variations, the density of RRAM prototypes ready for commercialisation is quite low at approximately 4 Mb.

Recent research has made significant progress in showcasing the capabilities of high-density devices with impressive features and low power consumption of approximately 0.1 pJ [394]. Additionally, these devices have better reliability, with endurance exceeding 10^12 cycles and retention of over 10 years at 150°C [127]. They also exhibit higher density of 32 Gb through simpler fabrication steps and stronger thermal characteristics [35]. This, coupled with the considerable resistance fluctuation not only among different devices but also within and between programming cycles on the same device [255], has engendered apprehension about the reproducibility of their electrical characteristics.

These issues have prevented RRAM from commercialisation, despite its many appealing characteristics. Thankfully, the primary deep learning-based neural processing methods, such as regression, pattern recognition, and speech recognition, are random in nature and require less accurate computing than deterministic traditional computing. As a result of fewer impact from device variations during manufacture, compared to memory applications, on computation outcomes [218], RRAM devices have generated additional possibilities to become suitable candidates for neural technologies. This can be attributed to the more tolerated necessity for variation in neural applications, alongside recent technical advancements.

1.2.5 Compute Hardware Basics

When discussing computer hardware, three key factors are essential: architecture, data representation, and method of computation. For instance, modern digital computers use the von Neumann architecture, represent data discretely, and perform computations using logic gates. The history of traditional computer architecture can be traced back to the 19th century when the concept of mathematical computing devices was first introduced [28]. The primary reason for this separation was complexity. Memory and computation units are challenging to design individually, so it made sense to separate them while still allowing communication

between them.

John von Neumann popularised this approach, which became the norm with the emergence of the first electronic computers in the 1940s [356]. However, this architecture can lead to the so-called von Neumann bottleneck. In many applications, data must be retrieved from memory and moved to computing units for the execution of mathematical operations, such as multiplication and addition, before being stored back in memory. If the application is data-intensive, a significant amount of time and energy is spent on transferring data between memory and compute units, rather than on the actual computations.

Data representation is a crucial aspect of modern computing, and it has been largely defined by the concept of the Turing machine. Alan Turing's work laid the foundation for programming, which is the idea that a machine can implement arbitrary algorithms [346]. However, this relies on machines operating on precisely identifiable states, which in practice means discrete data representation. The approach of digital computation also determined how calculations are performed in hardware.

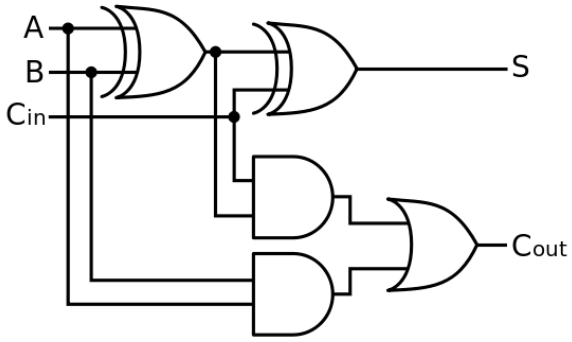

Fig. 1.6 In the full adder circuit, A is the operand to be added, and B is the operand being carried forward. C_{in} is the carry forward bit of the previous addition, and S is the total. The carry bit of the next addition is C_{out}.

Logic gates are the fundamental components of digital computers. The theory of logic circuits was developed in parallel with the conceptualisation of the Turing machine. Building on the work of George Boole, Claude Shannon and others [310], switching circuits were designed to solve arbitrary problems in Boolean algebra. The basic elements of these circuits, known as logic gates, were initially implemented using mechanical relays, vacuum tubes, and later transistors. However, even simple operations can require a large number of devices. For instance, a full adder circuit in figure 1.6, adds two binary numbers and a carry bit,

requiring 26 transistors [290]. Adding more bits necessitates more transistors, increasing energy consumption and circuit area.

There are several levels to compute optimisations in a digital computer. At the most basic level, the number of transistors in a computer can be reduced to allow for the implementation of more complex circuits in the same space. Different decisions can be made regarding the operations that those transistors are required to carry out; this is commonly known as instruction set architecture (ITS). The broadest instruction sets result in the most general purpose processors, such as the central processing unit (CPUs). Conversely, the narrower instruction set which results in implementations such as the graphics processing unit (GPUs), can lead to better performance in specific applications.

At a structural level, the conventional von Neumann architecture remains difficult to phase out, but there have been efforts to unify memory and computation even within the digital paradigm. For example, microcontrollers have a single chip that consolidates both computation and memory. Similarly, many cutting-edge AI chip designs aim to pack the maximum number of computational elements and memory units onto a single chip.

1.3 In-memory Computing Paradigms

As improvements at the device level in digital computers become increasingly difficult, alternative computing paradigms offer a more promising direction. Specifically, targeting the memory bottleneck mentioned earlier could bring significant benefits. The Von Neumann architecture results in slow and energy-inefficient data movement, which is even more pronounced today, as data-intensive applications like machine learning gain popularity. Bringing memory and compute units closer together, ideally achieving in-memory computing (IMC), could alleviate the problem.

Proposed solutions for addressing the von Neumann bottleneck vary. These include digital and analogue approaches, which can be differentiated based on the integration level between memory and computing units. The conventional von Neumann architecture, which completely separates memory and compute units, represents one end of the spectrum. In the pursuit of IMC, there are techniques that involve placing memory and compute units on the same chip.

One such technique is near-memory computing, which integrates embedded non-volatile memories (NVMs) into microcontrollers. In this approach, NVM is used to store model parameters, while volatile memory, such as SRAM, is sandwiched between NVM and compute units to hold intermediate input/output data.

In a true IMC approach, memory serves a dual purpose: storing data and performing computations. To achieve general-purpose computing in memory is incredibly challenging; therefore, a more realistic approach is to focus on specific applications. For instance, in the field of machine learning, the objective may be to expedite linear algebra computations.

One commonly used approach for IMC is the utilization of resistive crossbar arrays. In this approach, neural network weights are represented using analogue properties, such as device conductance. The underlying physics, together with the circuit structure, can then perform the necessary computations. This is achieved without the need to move the weights from memory to compute units, as the weights (in the form of conductances) are directly operated on by applying physical stimuli, such as voltage, to them.

1.3.1 Recent Development

Computers based on the von Neumann architecture and the complementary metal-oxide semiconductor (CMOS) technology are now ubiquitous due to their scalability, which has resulted in multi-decade improvements in speed and space efficiency. However, the slowing down of Moore's law and the increasing costs in data-intensive applications such as machine learning are becoming prohibitive. For instance, it has been reported that programs such as ChatGPT have high training and inference costs [251]. This could be a factor in their reduced performance [43], as minimising expenses may be prioritised.

Modern computer designs reflect the well-known issue of memory bottlenecks. Large volumes of data are stored in off-chip memory, such as dynamic random-access memory (DRAM), while frequently accessed data are stored in on-chip memory, such as static random-access memory (SRAM), which is located closer to the compute units.

On-chip memory can be more than 100 times more energy efficient than off-chip memory, but it is also more costly and requires more space [104]. That is difficult in the case of machine learning since the models being trained are so huge. Without compression, even models from ten years ago cannot fit into ordinary SRAM units, necessitating the use of

more energy-intensive DRAM [105].

Additional enhancements to the compute units can be realised following memory optimisation. CPUs are the most general-purpose calculation units, supporting a wide range of instructions and operating in a sequential order. They can be made more parallel by adding more cores, but whether this is effective depends on the application.

GPUs are more specialised, with a concentration on graphics processing, although they can perform a wide range of matrix operations. They are also more parallel than CPUs, but they can only execute a limited number of instructions and control flow. Tensor processing units (TPUs) are compute units created expressly with machine learning in mind; optimisations are made for matrix multiplication, data retrieval, and even instruction fetching [152].

Finally, device developments have tended to focus on lowering transistor sizes, resulting in higher density and speed. This general tendency (known as Moore's law) has resulted in transistors that are just a few nanometers in size. Commercial manufacturing of 5 nm transistors began in 2020, with deployments in Apple, Qualcomm, Huawei, Marvell, and Nvidia consumer products [359].

3nm transistor manufacture began in 2022, although widespread commercialization has yet to occur. Continuous transistor size decreases are extremely difficult and expensive to produce, with diminishing advantages. Moore's law (in its classic version) is not likely to continue indefinitely due to physical limitations—even now, the tiniest transistors may be just a few tens of atoms thick in certain directions.

1.3.2 Original Implementation

Linear algebra and vector-matrix products rely heavily on multiplication and addition. These procedures can be carried out by using fundamental circuit laws. Consider a resistive element with conductance G (the reciprocal of resistance). If a voltage V is supplied to it, the current I flowing through it will be equal to $V \times G$. This indicates that conductance G functions as a multiplicative factor, as per Ohm's law. For a circuit with several branches, each carrying a current I_i. At the intersection of these branches, the total current flowing through it will be $I = \sum_i I_i$. This indicates that currents are combined together, as per Kirchhoff's current law.

Once multiplication and addition are possible, higher-level operations may be performed using specialist circuits. For vector-matrix products, a resistive crossbar array can be used,

which is a two-dimensional grid of conductive wires with resistive components at each intersection. A crossbar array's output currents are essentially the product of a voltage vector and a conductance matrix. Consider a vector-matrix product, $\mathbf{y} = \mathbf{x}^{\intercal}\mathbf{W}$. Where $\mathbf{x}$ can be translated to voltages $\mathbf{V} = k_V\mathbf{x}$, $\mathbf{W}$ to conductances $\mathbf{G} = k_G\mathbf{W}$, and generate outputs $\mathbf{y}$ from currents $\mathbf{I} = \mathbf{y}k_Vk_G$, where k_V and k_G are positive constants.

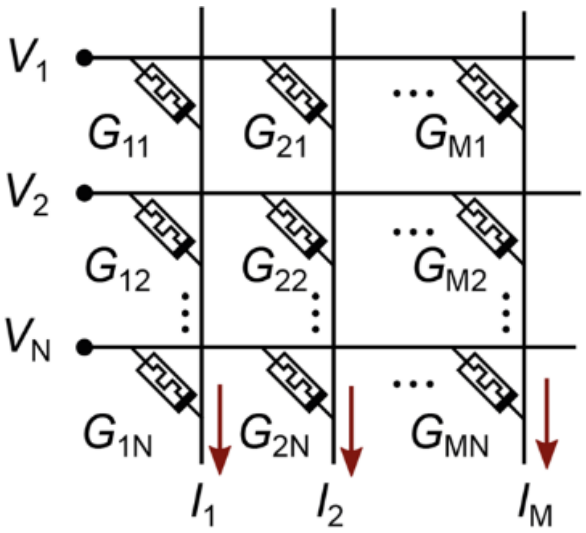

Fig. 1.7 The memristor-based crossbar architecture with a single memristor array and a constant-term circuit [343]. The resistive components are located at the connections of the word and bit lines. When voltages $\mathbf{V}$ are applied to the word line, the resistive element at the junction of the i^{th} word line and the j^{th} bit line generates $V_i \times G_{i,j}$ units of current, assuming zero wire resistance, as per Ohm's law. The currents created by each individual element are then aggregated along the bit lines using Kirchhoff's current law.

Crossbars can compute products of voltage vectors and conductance matrices due to their structural design. This is because the structure controls which voltage-conductance pairs are multiplied and which consequent currents are combined together. These circuits have two sets of wires: word lines and bit lines. Voltages $\mathbf{V}$ are applied to the word lines, and currents $\mathbf{I}$ are measured along the bit lines. A resistive element located at the junction of the i^{th} word line and the j^{th} bit line has a conductance of $G_{i,j}$.

When V_i is applied to the i^{th} word line, the device generates a current of $V_i \times G_{i,j}$ (assuming no wire resistance). The currents generated in the jth bit line are added together to provide a current of I_j. This current is calculated by taking the dot product of voltage $\mathbf{V}$ and the j^{th} column of the conductance matrix $\mathbf{G}$. Given that the j^{th} element of a vector-matrix product is just the dot product of the vector and the j^{th} column of the matrix, the vector containing

all output currents may be concisely expressed as $\mathbf{I}^{\intercal} = \mathbf{V}^{\intercal}\mathbf{G}$.

The individual application determines which resistive devices are used in the crossbar array. Weights $\mathbf{W}$ are repeatedly updated during neural network training, necessitating the ability to alter the conductances in the crossbar array numerous times. In contrast, during inference, the weights are fixed, allowing the conductances to be set after initial programming. Regardless of the conditions, the conductances will be unique to the network, requiring the ability to change them at least once.

Memristive devices, or memrisitors, are differentiated by their ability to change their conductance in response to electrical inputs. As a result, they make an excellent choice for crossbar-based linear algebra accelerators. The choice of memristor depends on whether the crossbar array is used for training or inference. The former is far more difficult and would demand memristors that can be repeatedly programmed in a linear fashion. Given these complications, much research on memristive crossbars has been on inference.

Even in the absence of nonidealities, any memristor will have a restricted range of conductance values that it can be configured to. This is a hurdle when attempting to represent real numbers with solely positive conductances G. To demonstrate, if the range of attainable conductances is $G \in [G_{off}, G_{on}]$, the crossbar array can only represent matrix values up to $w \in \left[\frac{G_{off}}{k_G}, \frac{G_{on}}{k_G}\right]$. Since G_{off} is a positive number, hence only positive w may be expressed.

One potential option is to employ differential pairs, in which the matrix element w is represented as the difference between two conductances, $G+$ and $G-$ [149]. The two conductances can be chosen symmetrically around the 'average' value $G\pm = G_{avg} \pm \frac{k_G w}{2}$, where $G_{avg} = \frac{G_{off}+G_{on}}{2}$. The two sets of conductances can be represented by independent conductance matrices $\mathbf{G}+$ and $\mathbf{G}-$, which are assigned to different bit lines of the crossbar array [166].

The bit lines will then generate independent sets of currents, which may be represented as vectors $\mathbf{I}+$ and $\mathbf{I}-$. Vector-matrix products are linear, thus the result may be calculated by subtracting $\mathbf{I}-$ from $\mathbf{I}+$. In reality, the 'positive' and 'negative' bit lines are frequently arranged near to one another, which helps to mitigate the detrimental effects of line resistance, a significant non-ideality [148].

1.3.3 Memory Cells

It is critical to understand the viability of various technologies for IMC systems. When volatile memory is used for computing, the computational parameters are often acquired from nonvolatile off-chip memory, which adds cost to overall system performance. The goal is to install IMC directly on NVMs, which would further minimise data travel, increasing energy efficiency and lowering latency.

One prospective path to explore is the use of mature Flash technology as a computational NVM. However, developing memory technologies have several advantages, including interoperability with increasingly powerful processor nodes. This is mostly owing to the lower operating voltages compared to Flash technology. Furthermore, several of these emerging technologies provide multi-bit or even fully analogue programmability [223]. This may be used to conduct multiply-accumulate operations in the analogue domain, resulting in additional performance advantages.

In a broader context, irrespective of the IMC concept, it is crucial to recognise that contemporary computing systems utilise a range of memory technologies, which are typically organised in a hierarchical structure. These memory technologies, which operate within the digital paradigm, are based on mature complementary metal–oxide–semiconductor (CMOS) fabrication and technology.

Table 1.2 Benchmark of emerging memory technologies for various applications. [252].

	STT MRAM SCM/DRAM	Embedded MRAM	SOT Cache	Standalone PCM	Embedded PCM	Standalone RRAM	Embedded RRAM	FeRAM	FeFET
Capacity	>1Gb	10-100Mb	>1Mb	Gb	10-100Mb	∼1Gb	1-10Mb	Poor	Small
Scalability	Medium	Medium	Poor	Good	Good	Medium	Good	Medium	Poor
MLC	No	No	No	Possible	Possible	Theoretical	Theoretical	Theoretical	Theoretical
3D Integration	No	No	No	Yes	Yes	Yes	Yes	No	No
Architecture	Crossbar	Crossbar	3 Terminals	Crossbar	1T1R	Crossbar	1T1R	1T1R	3 Terminals
Retention	>1yr 100°C	10yrs 150°C	85-100°C	85-100°C	Automotive	10yrs 85°C	10yrs >85°C	85-100°C	SMT Compliant
Latency	10ns	10ns	<ns	100ns	100ns	100ns	100ns	<20ns	5ns
Power	pJ/bit	pJ/bit	fJ/bit	10pJ/bit	10pJ/bit >200uA	1-10pJ/bit	1-10pJ/bit ∼100uA	10fJ/bit	10fJ/bit
Endurance	10^{10}	$> 10^6$	$> 10^{10}$	10^7	10^6	10^7	10^6	$> 10^{11}$ destructive	$10^4 - 10^5$
Issues	N/A	N/A	N/A	Drift	Drift	Variability, Noise	Variability, Noise	Small Size Variability	Small Size Variability
Space	DRAM	NVM	Cache	SCM	MPU,MCU	SCM	MPU,MCU	DRAM	Flash
Maturity	Everspin, Avalanche	Avalanche, TSMC	No Products	Intel, Micro	ST Microelectronic	No Products	Panasonics, Dialog, TSMC	Texas Instruments, Fujitsu, Cypress	Good

In the majority of cases, electrical charge is employed as a surrogate for data, and these are designated as charged-based memories. Fast volatile memory, such as static random-access memory (SRAM), represents the pinnacle of this hierarchical structure. It has the shortest

access time, but is also characterised by low area density and typically the highest cost.

The middle of the hierarchy is typically occupied by off-chip DRAM, which serves as the main memory. It exhibits superior area density but experiences a slight extension in access time. At the lowest point of the hierarchy, there are NVM technologies such as flash memory (in solid-state drives, or SSDs) and hard disk drives (HDDs), which possess the highest area density but are significantly slower. In this memory hierarchy of computing systems, emerging NVM technologies are seen as a means of bridging the performance gap, particularly in terms of speed, between the extremes of the hierarchy.

The most promising emerging memory technologies include resistive random-access memory (RRAM), phase-change memory (PCM), ferroelectric random-access memory (FeRAM), magnetic random-access memory (MRAM), spin-transfer torque magnetic random-access memory (STT-MRAM), and ferroelectric field-effect transistor (FeFET). It should be noted, however, that this is not an exhaustive list. There are other promising technologies under exploration and rapid development, such as electrochemical random-access memory (ECRAM) [223], which could offer further computational performance benefits.

In order to provide a comprehensive overview, we will briefly address the physical mechanisms that underpin the operation of these memory technologies. Regardless of the specific technology under consideration, in addition to the typical requirements of NVM, the crucial enabling factor for executing linear algebra operations using, for instance, crossbar arrays is the ability to effectively configure a single memory cell into multiple stable, non-volatile memory states.

A resistive random-access memory (RRAM) cell is typically structured on a metal – insulator – metal basis, with the central insulating layer serving as a resistance-switching layer. This layer is typically composed of metal oxides, although other materials, including chalcogenides, organic substances, nitrides, and two-dimensional (2D) materials, are also being investigated.

Furthermore, a distinction can be made between filamentary and interface-based switching. This differentiation depends on whether the write operation results in the formation of conductive filaments within the insulating environment, or if the resistance switching is based on modulating barrier heights (for example, Schottky barriers) between the switching layer and

the electrodes (metals).

A PCM cell shares certain similarities with an RRAM cell; however, the switching layer in PCM is composed of phase-change materials. These materials possess the capacity to reversibly transition from a crystalline to an amorphous phase. A representative example of a phase-change material is chalcogenide (such as $Ge_2Sb_2Te_5$), wherein the phase transition is driven by Joule heating generated by voltage pulses applied across the cell.

In addition to the two separate states, RRAM and PCM may also store intermediate states. This can be accomplished by either changing the shape of the conductive filaments (as in RRAMs) or adjusting the fraction of material that transitions between the two separate phases. This functionality is particularly useful in the context of analogue or multi-bit computing. However, other obstacles remain, including the accuracy and feasibility of programming several intermediate states, as well as their preservation.

Ferroelectric random-access memory (FeRAM) and ferroelectric field-effect transistors (FeFET) are both based on the use of ferroelectric effects. Typical examples of ferroelectric materials for FeRAM are perovskites. However, recent research has demonstrated significant interest in employing ferroelectricity from HfOx-based materials [257] (either doped or undoped) due to their superior compatibility with complementary metal-oxide semiconductor (CMOS) technology.

In the case of FeRAM, the state is read by measuring the displacement current during switching in the ferroelectric material, which is a destructive process and not ideal for computing applications. Alternative strategies include ferroelectric tunnel junctions (FTJs) and three-terminal FeFETs. In these cases, the states are read either as resistance or a threshold voltage. As a result, the read process is non-destructive and better suited for computing systems.

MRAM uses the magnetoresistive phenomenon to store data. The cell structure is similar to that of RRAM and PCM, but MRAM has two ferromagnetic layers, each capable of holding a magnetic field and separated by a thin tunnel layer. One of the ferromagnetic layers has a constant magnetic orientation, whilst the other layer may change its magnetic orientation to be parallel (aligned) or anti-parallel (anti-aligned) relative to the fixed layer. The first represents a logical '0', and the second represents a logical '1'. The state is determined by measuring resistance, as the two configurations produce different resistances due to magne-

toresistance effects.

Currently, the two most common MRAM implementations are STT-MRAM and spin-orbit torque magnetic random-access memory (SOT-MRAM). The cell architecture of both implementations are identical, with the key variation being the approaches used to program the cell state. In the case of STT-MRAM, the magnetic orientation of the 'free layer' may be changed using spin torque, which transmits current pulses across the tunnel junction. In contrast, in SOT-MRAM, the transition to a new state is triggered by applying a current pulse down the heavy metal line (such as platinum or tantalum) on which the tunnel junction is formed. The current pulse causes the buildup of spin-polarized electrons, which switches the magnetisation of the free layer.

As previously noted, the notion of ECRAM has received significant attention in recent years. The ECRAM structure is similar to a transistor, with an extra vertical stack of a reservoir layer and a solid-state electrolyte. The insertion of ionised defects into the reservoir layer has the potential to change the channel conductivity. These flaws are often oxygen vacancies or lithium ions, although they may also comprise other species, such as protons. ECRAM has the potential to outperform other NVMs in terms of cycling endurance and linear conductance modulation control. This linear conductance modulation is frequently used in machine learning applications, particularly during training.

Memtransistors are a more contemporary idea that combines a three-terminal transistor shape with memristor-like functionality. This makes it easier to modify the channel's conductance by applying significant source-drain voltages. These devices often utilise two-dimensional semiconductor materials, such as MoS2, for their channel. Compared to the other NVM technologies discussed, memtransistors are still in the early phases of development.

1.3.4 Neural Processors

Advances in ANN implementation and system speed, which prompted modifications in computer architectures, have resulted in improvements not just in software but also in hardware [138]. To run neural networks, many forms of hardware were used, including standard computers, neural accelerators, and neuromorphic computing [217].

In reality, neuromorphic computers are becoming the most effective means of running neural networks, which necessitate whole other kinds of compute equipment, hence the names "neuromorphic chips" or "neural processing units" (NPUs). The number and structure of

computational cores distinguishes central processing units (CPUs), graphics processing units (GPUs), and neural processing units (NPUs) [334].

In comparison to traditional CPUs, which are now sold with up to tens of cores, NPUs and GPUs have been claimed to contain over several thousand cores [102]. Despite major variations in operations and organizations, there are numerous commonalities between NPUs and GPUs since they both use many cores for parallelizing neural network workloads, which may be found in image processing jobs. GPU cores, like CPU cores, serve primarily as processing units and lack memory capabilities.

This already takes into consideration the existence of SRAMs, which mostly served as caches. This suggests that GPUs require external memory to work, indicating that the GPU processor and memory are fundamentally independent functions. The primary difference between the core architectures of both CPUs and GPUs versus those of NPUs is that the latter are built with cores that have their own memory units, or synapses, that can perform neural network operations [287].

Thus, NPUs are capable of working with memory and processing concurrently under the same calculation, which is inspired by and extremely comparable to that of the human brain. Many contemporary NPUs use SRAM and DRAM as synapses, which are volatile components. This suggests that extra storage devices, such as flash drives or storage devices with non-volatile characteristics, are still required to maintain their synaptic weights, even though their processing and memory operations are connected [46].

The implementation of memristive devices as non-volatile synapses can reduce the need for external storage in order to have concurrent processing and memory during computation [165]. In order to dramatically reduce energy consumption and information congestion during data transfer across memory devices and processing components, NPU designs have continued to investigate the interconnected capabilities between processing and memory [189].

The ideal neural chip should have massively parallel processing capacity for bit-wise, fixed-point, and floating-point calculations of many different data sizes; low memory latency; memory bandwidth hundreds of times greater than what is obtainable in today's desktop devices; a remarkable memory size capable of handling big data; and a novel architecture

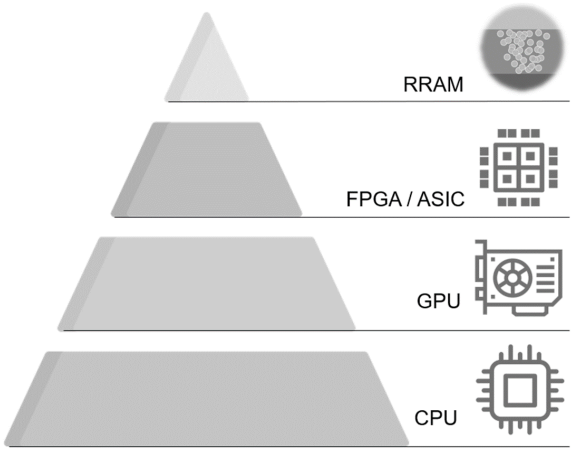

Fig. 1.8 Typical hardware technologies for DNN acceleration. Specialized hardware for high-performance DNN training and inference is included in the pyramid's top layer. RRAM is the name of the apex, although the term is meant to include all programmed non-volatile memristive devices.

that allows flexible and rich interconnection between memory and computing.

Each of the aforementioned requirements should be fulfilled while also taking into account reduced power consumption and increased energy efficiency. This makes it possible to create neural network units in circuit designs within NPU cores. According to the following general operating theory, the input neuron in the first layer can take a data input from the pre-synaptic neurons, perform matrix vector multiplication (MVM) with synaptic weights, and then produce another value for the hidden neurons.

These hidden neurons will act as input data for the subsequent neuron layer and then output neurons. Multiplication, addition, and activation have traditionally been the roles of the input and output neuron layers. While the weight storage capability of synapses may be implemented in memristors to retain the updated weights, these processes can be accomplished using CMOS electronic or analogue circuitry. In this view, weight storage becomes the main purpose of conventional synaptic structures [302].

Up to this point, only the use of memristive devices for weight storage has been considered, which remains the most prevalent option for practical NPUs. The most recent study is presently focusing on ways of more effectively and innovatively utilising memristors by leveraging the particular nonlinear $I-V$ characteristics of memristive devices.

It has been investigated whether the traditional method of matrix-vector multiplication, in which matrix components store analogue attributes or multi-level device values, can improve the effectiveness of ANNs by expanding synaptic devices' capabilities beyond their use in basic weight storage in ANN. [386]. The memristive crossbar array can effortlessly perform matrix-vector multiplication and naturally convert the weighted mixture of input signals to output signals, reducing the computational cost to $O(1)$ from $O(n^2)$.

To create a memory-sensitive crossbar that can conduct analogue multiply-accumulate (MAC) actions in a single time interval, a variety of device fabrication techniques may be used [291]. In a non-von Neumann architecture, doing multiplication at the memory cross-point can result in an immediate decrease in temporal complexity. Under this widely used strategy [391], MVM may be parallelized with the vector denoted with N as the size of the input voltage signals $[V_1, V_2, ...V_N]$.

The crossbar rows are subjected to these voltages, and the size of the matrix is (NxM). Its weights are kept in the memristive element at each cross point and are expressed as conductances. By using the fundamental Ohm's equation $I = V.G$, the current accumulated at each horizontal column represents one component of the resulting multiplication vector of length N.

Extra logic electronics, such as multipliers and adders, are not necessary for this matrix-vector multiplication approach because of the special characteristics of memristive devices, which enable them to automatically conduct multiplication and addition concurrently. The memristor may efficiently perform the functions of weight storage, multiplier, and adder throughout the process [277].

The memristive device has to have appropriate conductance to voltage proportionality to support such operation [246]. This methodology may be extended to do matrix-vector multiplication and train neural networks [314]. This variable resistance technology, which enables neural networks to be controlled via MVM in neural processors, is poised to make memristive devices particularly ideal for enabling it. [390].

Memristors for MVM are much easier to construct than weight-storing methods [111]. It has frequently been claimed that using NPUs for computation is much more efficient than using conventional processing elements. This more intense MVM technique produces incredibly

effective processing. [54].

The MVM method is, in essence, the most effective technique to employ memristors as neural network synapses. However, there are still issues with real-world applications that need analogue properties or enough multi-levels of memristive devices to support machine learning for efficiency.

1.3.5 Hardware Accelerators

Meristive devices have been used as weight components in many studies' neural network designs [315] in order to meet the two aforementioned main requirements for deep neural network acceleration: huge parallelism and decreased memory access [11]. This adaptive characteristic is an essential component of the brain's in-memory computing capabilities, which is also lacking in today's general-purpose compute resources' traditional design.

This in-situ processing power may be used to conduct concurrent operations within memory, allowing for major advancements in deep neural network learning and inference. This is primarily achieved through the creation of memristive crossbar neuromorphic designs, which are predicted to result in a 2,500-fold reduction in power and a 25-fold increase in acceleration when contrasted to advanced specialised hardware such as GPUs [77].

Fully linked deep neural network layers may be created by simply transferring the weights to crossbar memristive points and mirroring the voltages in the inputs. In order to convert the extra convolutionary operations to standard matrix operations for the more complicated convolutional neural network implementation, mapping techniques are needed. The input feature mappings and convolutional filterings are convolutioned into matrix operations using a unfolding technique as one method for this conversion [183].

Aside from the memristive devices utilised as programmable components in memristive deep neural network (MDNN) designs, additional peripheral circuitry is often required to implement feed-forward with back propagation learning rules in MDNNs [173]. The additional circuitry may include any of the following: a conversion circuit for converting input feature maps to input signals in the form of pulse width modulated (PWM) circuits for programmable memristive devices; current aggregators or amplifiers that move the reading current over each column of the memristive crossbar; analogue to digital converters that pass the voltage passing through the crossbar; activation function circuits and their derivatives.

The network weights must also be modified using an update module, which requires additional software code and uses a technique like stochastic gradient descent (SGD). After that, the updated memristive weights must be transferred to the memristor crossbar, which necessitates the usage of Bit-Line (BL) and Word-Line (WL) switch arrays to address the memristors for update and a circuit to do so. There are multiple ways to construct MDNN accelerators using the various recommended circuits. Ex-situ learning, in which the new weight levels are produced and delivered to the real memristors by auxiliary electronics, has been employed in other experiments.

The ideal memristive crossbar is anticipated to significantly speed up DNN learning and inference while drastically reducing power consumption, but when crossbar dimensions are scaled up for implementation in actual DNN architectures, particularly ones needed for smart healthcare and biosignal processing, device non-idealities noted in experimentally manufactured memristors impose major performance degradation. Examples of non-idealities include limited on/off ratios, nonlinear asymmetry and stochastic conductivity variations, unit yield, and temporal and spatial unit fluctuations.

To lessen the impact of these faults, specific peripheral circuitry and system-level mitigation techniques were also created [399]. Naturally, these techniques significantly increase system complexity and time consumption. Therefore, it is crucial to consider how these nonidealities would affect MDNN performance prior to actually using them in biosignal processing applications where accuracy is crucial. Additionally, a universal tool is essential that can reliably mimic the transformation of a pre-trained DNN into an MDNN while taking into consideration the defects of the devices that have been experimentally described.

Training of these complicated algorithms is often carried out in external data centres due to the tremendous amount of time and energy required to develop a brand-new version of a massive DNN with the aim of accomplishing challenging cognitive tasks, particularly in biosignal processing [23]. The weighted pre-trained DNN may then be translated for use on memristive crossbars using a variety of current frameworks and tools that can mimic and ease this transition [8].

There are currently no massive scale MDNN systems that have realised any useful biosignal processing applications, even at the emulation level. However, some lower scale MDNNs have been emulated for biosignal processing applications such as cardiac arrhythmia categorization [112] or after being used to detect breast cancer on a physical programmable

memristive array [32].

Comparable to recent developments in CMOS-based DNN accelerating chips, hardware implementations of MDNNs, which have repeatedly shown to yield significant energy savings compared to cutting-edge GPUs, are projected to be partially or completely realised. These implementations can currently only do basic tasks like MNIST and CIFAR classification, in contrast to their CMOS equivalents. The construction of large-scale neural networks needed for biosignal processing tasks involving sequential or temporal data sources is by no means optimal in this case.

1.3.6 Signal Processing

Case studies of neural networks built using the back-propagation learning method in healthcare and biosignal processing, like cancer detection [269] or monitoring ECG [177], date back to the early 90's. Because they were very shallow and had few parameters, there was little demand for high-performance computer resources. There was an increased need for specialised high-speed accelerators as a result of the return of CNNs in the early 2010s, which was followed by the rapid spread of DNNs and enormous data sources [6].

As a result, there are new demands for GPU repurposing, and research into alternative types of electronics and implementation methods, such as CMOS processors and ASIC platforms, [180], FPGA systems for DNN inference [182, 181], as well as memristive crossbar and in-memory computing [395] has grown. Despite significant advances in the deployment of non-GPU systems for deep learning acceleration [179], healthcare workloads and biosignal processing have depended mostly on traditional technology such as GPUs, as do other data processing activities.

Depending on the complexity of the neural network needed, the number of parameters, and the size of the available training data, biosignal processing tasks are frequently trained on high-end computing resources with large clusters, on customised proprietary processing units like Google TPU, or on a variety of infrastructure-as-a-Service provider platforms like Microsoft Azure, GCP, and AWS, among others [236].

A few of the benefits of outsourcing the compute service include the ease with which these platforms provide the same programming languages, such as Python, the accessibility of well-known open source deep learning repositories, such as Keras and Torch, and the presence of a sizable community of service providers who support the use of GPUs and Infrastructure-as-

a-Service for various deep neural network implementations.

Due to all of these factors, deep learning inference can still benefit from additional research and development with regards to both cutting-edge and established hardware design technologies, paving the way for low-power and inexpensive deep learning hardware for point-of-care devices and the Internet of Things in the healthcare industry. Despite their appeal, medical inference systems and edge processors that can handle biosignals are still not widely used in hardware.

1.3.7 Edge Computing

On-chip learning and adaptive mechanisms, which allow the selected system to adjust to each patient's own biosignature and drift in time, may be very helpful for biosignal processors and customised therapy. Given this, it is currently difficult to implement effective on-chip online learning in these circumstances using neuromorphic architecture. The two key elements influencing this challenge are the placement of the weight update and storage.

To avoid wasting a considerable amount of chip area for wiring that needs extra routes to update this information, the synapses must be able to obtain learning information for each on-chip network's weight changes locally [85]. Because the Hebbian learning rule may meet this criteria, the majority of existing on-chip learning algorithms today focus on implementing this algorithm in various versions with unsupervised or semi-supervised learning [86].

This has several downsides since local Hebbian-based learning rules are constrained to static patterns or use a very shallow network [12]. More and more researchers are using on-chip gradient-descent approaches, which aim to create error-based training techniques with the lowest mean square error for the objective functions of the neural network.

In the great majority of modern multi-layer machine learning applications, the spike-based delta algorithm is the most frequently used weight update for single-layer networks. It is based on the back-propagation algorithm [273]. Particularly noteworthy is the development and application of a single-layer mixed-signal memristive system based on the delta rule for the categorization of EMGs [153].

Non-local weight changes with limited on-chip deployment become more apparent when used with multi-layer deep learning [22], making localization of the back-propagation algorithm a popular area of ongoing study [297]. The ideal weight storage system for continuous

on-chip training is a memory with non-volatile characteristics that allows for linear analogue state changes [349]. Particularly non-volatile memristors offer a lot of possibilities for such uses.

There exists a significant research on both deep learning accelerators, with the possibility of obtaining complementing benefits like real-time data processing and power consumption reductions of many orders of magnitude. These neuromorphic processors are excellent for end-to-end use cases such as wearable devices with streaming inputs that are continually checked in an always-on approach. A number of studies have been published that use combined analog-digital neuromorphic systems for biosginal processing tasks. This is summarised in Table 1.3 with some of today's most promising neuromorphic processors.

Table 1.3 Neuromorphic platforms used for biosignal processing applications [10].

Neuromorphic Chip	**DYNAP-SE**	**SpiNNaker**	**TrueNorth**	**Loihi**	**ODIN**
CMOS Technology	180nm	ARM968, 13nm	28nm	14nm FinFET	28nm FDSOI
Implementation	Mixed-signal	Digital	Digital ASIC	Digital ASIC	Digital ASIC
Neurons per core	256	1000 (1M cores)	256	Max 1k	256
Synapses per core	16k	1M	64k	114k-1M	64k
Energy per SOP	17pJ @ 1.8V	Max 1W per chip	26pJ @ 0.775V	23.6pJ @ 0.75V	12.7pJ @ 0.55V
Size	38.5mm2	102mm2	–	60mm2	0.086mm2
Biosignal Considered	EMG [65], ECG [19], HFO [311]	EMG and EEG [21]	EEG and LFP [267]	EMG [34]	EMG [34]

Commercially, it has already been proved that CMOS technology can be combined with developing technologies, notably non-volatile filamentary switching [113]. Other efforts to combine CMOS with memristor technology are in progress, with the goal of creating supervised local error-based training circuitry with just one network layer by utilising the special features of memristive devices [272].

There are currently relatively few large-scale biosignal processing MDNN implementations that use generic programmable CMOS and memristor integration or otherwise, with just one such architecture programmed for an MLP with cancer detection. Despite the significant advantages that memristors provide in terms of parallel processing and in-memory programming paradigms, there is still a scarcity [57], while simultaneously working with CMOS integration [48].

Memristive devices are excellent candidates for deep learning processors in particular because to the strict restrictions on device size and power consumption, especially for mobile and edge-based smart healthcare. Memristors have limitations in terms of durability, misalignment, and analog noise buildup, which must be addressed before employing them for

biosignal processing.

This creates a new need for more research into this emerging technology's material, device, and platform design aspects while also creating an infrastructure for open-source software that supports MDNNs. Additionally, research into methods similar to those employed in the creation of CMOS-based deep learning processors, like approximated computation algorithms and low-precision data modeling, can advance the creation of MDNNs and make it easier to use them for edge biosignal processing.

1.4 Neural Computing Nomenclatures

The field of machine learning (ML) is a sub-branch of artificial intelligence (AI) concerned with teaching computers to perform various tasks. Initially, AI focused on programming machines to perform intelligent actions, drawing from fields such as optimization and control theory. One limitation of this initial approach is that the programmer must provide precise instructions to the machine on how to behave in various situations.

Rule-based systems have been developed to guide behaviour, but they are only successful for certain tasks. For instance, it is challenging to determine an explicit set of rules to differentiate between visual stimuli such as a cat and a dog. One approach is to create rules based on characteristic features, such as differences in their ears, noses, and mouths. However, creating rules that cover the majority of situations is challenging, and even promising rules may prove fragile to changes in viewing angle, distance, or environment.

Recognising objects is a subconscious process, making it difficult to establish explicit rules. Unlike mental math, for which we have established rules, object recognition lacks such explicit guidelines. Our visual system has been exposed to millions of seconds of ever-changing stimuli, and has learned to process them in a way that enables us to distinguish different types of objects and judge the depths of obstacles in our environment.

ML was developed when researchers began to explore the possibility of machines learning implicit methods of operation. Instead of programming a set of rules or commands, the machine can learn its own program through a data-driven approach. No changes in content have been made. This is often described as a data-driven approach: give the machine a general structure for the problem, and a lot of data, and let it learn a way to solve the problem. However, there is no universal structure that performs well across all types of problems.

Therefore, much of the research in ML focuses on identifying the most effective structures and learning procedures for specific problems or groups of problems.

Artificial Neural Networks (ANNs) are a popular sub-field of machine learning. They are inspired by the brain, where a neuron (or node) is the basic unit of calculation. In ANNs, a neuron takes a set of inputs that are linearly weighted and passes them through a statistical function to produce the output. ANNs can contain numerous neurons. Machine learning often involves adjusting the model parameters, typically the weights on each neuron's inputs, to achieve the desired output.

Another complicating factor is that the system should perform well not only on the trained tasks but also on unseen ones. While it is easy to remember input-output pairs, it is much harder to generalize that knowledge to new inputs in a 'smart' way. This is the crux of the machine learning problem. This section outlines the specialized terminology concerning the functioning of artificial neural networks. This is then translated into actualized memristive neural networks in this work.

1.4.1 Historical Considerations

Since the invention of the first electro-mechanical computers, attempts have been made to use them for cognitive tasks beyond precise mathematical calculations, such as image classification. Machine learning has a rich history of exploring various approaches, but statistical methods, particularly artificial neural networks and their many variants, have become the most popular and widely used for many machine learning tasks.

In the previous computer age, there were numerous efforts at different computing techniques, such as thinking machines and Turing machines [84]. The von Neumann [304] architecture has kept up with semiconductor technology developments as stipulated by Moore's law [345]. Unfortunately, this pace is still insufficient in the face of the emergence of the big data age, which involves massive volumes of data being processed swiftly under numerous uncertainties [233].

This emerging tendency suggests a significant urge to shift the computing paradigm away from speed and toward efficiency. It is commonly understood that the human brain is 500,000 times more efficient than a normal computer [316]. As a result, research has shifted toward imitating brain activities for more efficient processing, which is also a major driver of AI technology. Significant developments in AI technology and applications have been made in

recent years.

Nowadays, the area of artificial intelligence is flooded with numerous technologies, some of which are more advanced than others, causing some misunderstanding between the terms AI, machine learning, and deep learning. Many different opinions exist in discussing the links between stated topics. The general consensus is that AI is an umbrella concept that encompasses everything related to intelligent computing, with machine learning being a type of computation and deep learning being a specific type of technology that can train computing systems to be smarter in some ways through deep neural network technologies [14].

The overall notion of AI remains fascinatingly basic yet broad, with the fundamental goal of creating intelligent computers capable of making their own judgments [377]. Machine learning was developed as a subset of AI that empowers computers with the tools they require to increase their learning skills without being explicitly programmed by exposing them to enormous volumes of data. The capacity to learn from provided data sets prior to decreasing mistakes or enhancing the chance of their predictions being true is the motivating premise underlying machine learning [139].

Following on from this, deep learning is a branch of machine learning that is primarily concerned with algorithms that draw inspiration from the structure and function of the brain. Another way to think of deep learning is as the capacity to automatically extract key characteristics from data that are essential to the objective of classification or inference, as opposed to other more conventional machine learning scenarios where these features must be explicitly established. [411].

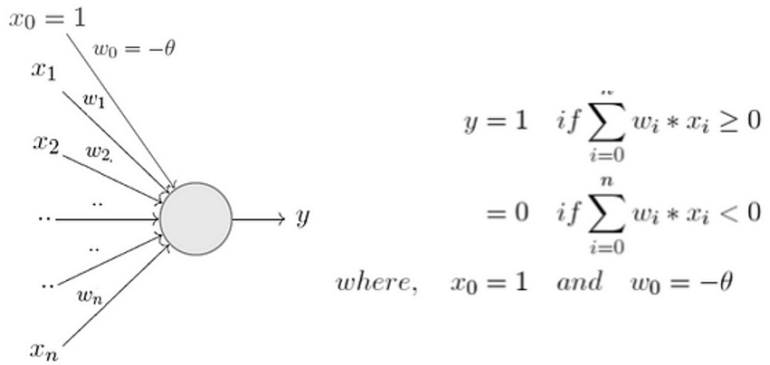

Fig. 1.9 The structure and operation of a perceptron [231].

The perceptron, one of the most primitive versions of artificial neural networks, was conceived in the 1940s [234] and built in the 1950s [294]. It functions as a simple binary classifier, making decisions based on a weighted sum of inputs x. During training, the weights w and bias θ are adjusted to minimize the error on the training set. For instance, a perceptron can be used to distinguish between puppies and kittens based on the intensity of image pixels.

For a period of time, criticisms of the perceptron resulted in reduced funding and research in the field of connectionism [231]. During this time, there was a shift towards rules- and heuristics-based systems [191]. However, these methods often lacked robustness and had to be redesigned for individual tasks. In the 1990s, connectionist approaches resurged due to advances in training algorithms, such as backpropagation [295], and the inception of more advanced architectures.

Connectionist approaches resurged in the 1990s, thanks in large part to the improvement of computing devices (Moore's law) and the development of GPUs. The parallelisation of computations, crucial for graphics processing (e.g. in computer games), proved to be useful in statistical machine learning as well, since the same underlying mathematical methods based on linear algebra were used.

After offering solutions for a wide range of AI applications, including but not limited to pattern recognition, natural language processing, modeling and forecasting, voice recognition, bio-informatics, driverless cars, drones, and many more, deep neural networks have attracted increasing attention [117]. Many changes have been made to various ANN designs in order to improve accuracy and efficiency in diverse applications.

Beginning with Rosenblatt's development of the perceptron in 1958 [124], neural networks have been created by overcoming restrictions such as self-contradiction [119]. A multi-layered feed-forward neural network that can be adequately trained one individual layer at a time and fine-tuned by back-propagation using unsupervised Boltzmann machines (RBMs) was only created in 2006 [329].

Most modern artificial neural networks include a generalisation of the perceptron known as fully synaptic layers [324]. These layers can have multiple postsynaptic neurons and include a nonlinear activation function σ. Each synaptic layer consists of pre-synaptic neurons x and post-synaptic neurons y, with the synaptic weights stored in a matrix w. The activation of

each postsynaptic neuron is calculated by taking the dot product of the presynaptic neurons and the corresponding column of the weight matrix (plus bias), followed by the activation function.

To calculate the activation of all postsynaptic neurons, combine them into a vector y and express it as a vector-matrix product of the inputs $x^\intercal$ and weights w (plus bias vector), followed by the activation function applied element-wise. Multiple fully connected layers can be combined to form a multilayer perceptron. For example, a multilayer perceptron can be used to recognise handwritten digits [271]. Whether the network used is a convolutional neural network, a transformer, or some other architecture, fully connected layers are typically included.

However, the use of fully connected layers can be computationally and memory-intensive. In a synaptic layer with m presynaptic neurons and n postsynaptic neurons, the number of weights required is $(m+1) \times n$, due to the connection of each presynaptic neuron to each postsynaptic neuron, and the presence of one bias weight per postsynaptic neuron. Although computations can be expensive, the majority of the time and energy is spent on transferring data between the memory and compute units of the von Neumann architecture.

Today, neural networks are being emphasised as technology that may be used not only for deep learning challenges, but also in a variety of other AI applications [406]. ANNs are a classic example of applications that are restricted by the von Neumann bottleneck. The weights, in the form of large matrices, must be retrieved from memory every time a synaptic layer is utilised. Fetching data from higher-capacity off-chip memory can exacerbate the problem, especially with large artificial neural networks.

1.4.2 Machine Learning Goals

During the early stages of artificial intelligence, the majority of work was theoretical. Researchers would create algorithms and rules based on their understanding of the problem to solve it. This soon became clear that for most problems, it was challenging to explicitly write all the necessary rules. For example, traditional computer vision techniques relied on identifying specific hand-defined features in an image. This approach can be effective for tasks such as tracking moving objects, where basic features can be combined with motion principles to monitor an object's movement from one frame to the next.

However, for tasks such as object recognition, it is extremely difficult to define optimal features and rules explicitly. To distinguish between a cat and a dog, or a boat and a truck, one would need to design feature detectors for specific characteristics such as ears, eyes, mouths, and noses. These features would then be used to define rules for different types of objects. This process would require the creation of hundreds of features and rules, each of which would need to be hand-tuned to work properly on the data.

Machine learning adopts a data-driven approach. Instead of explicitly defining good features and rules, the objective of machine learning is to learn them from the data. The aim is to learn a function that can perform the desired computation on new data points (i.e. different from those used for learning). This is known as generalisation, which enables a machine-learning system to be deployed in the real world.

If the system can only perform well on stimuli it has already encountered, it will always perform poorly on new stimuli unless we can show it every possible stimulus it will ever see (which is impossible for high-dimensional stimuli, such as images). However, if the system can generalize from the training stimuli to similar stimuli, it can perform well on novel stimuli too. The computation desired from a machine learning system is implicitly defined by both the available training data and the learning paradigm.

1.4.3 Learning Archetypes

The learning archetype determines the available information for learning and the desired system output. There are three main types: supervised learning, unsupervised learning, and reinforcement learning. These differ in the type of error signal provided to the learner.

Supervised Learning: Supervised learning involves learning an input-output mapping using both desired inputs and outputs. It can be divided into regression problems, which have continuous outputs, and classification problems, which have binary outputs representing membership in different categories. The error signal is determined by the difference between the system's output and the desired output.

Unsupervised Learning: In unsupervised learning, input data (such as images) is used without desired outputs. Instead, the output is implicitly defined based on the input data. Unsupervised learning includes clustering, which groups input data into sets, and dimensionality reduction, which represents data using fewer dimensions than the original representation. The error signal is defined based only on the input. For dimensionality reduction, the goal is

to create a system that can represent an input with fewer dimensions while still being able to recreate the input from this representation. The error signal in this case would be the distance between the re-creation and the original input.

Reinforcement Learning: Finally, reinforcement learning has a temporal aspect. When given the current state of the environment, the system must choose an action that determines the next state of the environment, which may be probabilistic. The environment rewards certain states, punishes others with negative rewards, and leaves some neutral with no reward. The system aims to achieve maximum total reward over time.

While these are the main categories of learning problems typically studied in machine learning, it is important to note that this is not an exhaustive list. Other paradigms, such as semi-supervised learning, exist where some data points are labelled and others are not, falling between supervised and unsupervised learning. It is worth noting that supervised learning remains the dominant paradigm in object recognition.

As datasets become larger, the cost of fully labelling them increases. Therefore, object recognition is likely to incorporate more semi-supervised and unsupervised methods, and even reinforcement learning. For example, object recognition may be part of a larger problem, such as winning at video games [250].

In supervised learning, there are two traditional types of problems: regression and classification. The difference lies in the output of the system. Regression systems output one or more continuous values by computing a vector-valued function on the input, whereas classification systems compute a single discrete-valued output that represents the class label of the input.

Recently, there has been growing interest in problems that extend beyond these two categories. An example of a task is predicting multiple labels for an image, such as identifying several objects in an image, or determining verb-like correspondences between objects [159]. This can also be extended to generating captions for images [392], where the model output is a sentence that describes the image.

1.4.4 Essential Terminologies

Each machine learning system can be described using four components. These components fully characterise the learning problem. If all components are described in detail, anyone

should be able to reproduce the result. Machine learning research examines at least one of these components since they determine the capabilities of a learner.

Problem Dataset: The available data implicitly defines the problem to be learned. In supervised learning, this consists of input/output pairs, while in unsupervised learning, only inputs are used. In reinforcement learning, the data is an environment that specifies the possible states and the rewards associated with each state.

System Architecture: How the learner is constructed, i.e. what parameters are learned, and how the system output is determined given these parameters and an input value.

Objective function: In supervised learning, two functions are often used to evaluate a particular set of parameters: one for training and one for testing (evaluation). The training objective function is called a loss function, where zero loss indicates 'perfect' performance, and any positive loss indicates how far we are from perfect performance.

Algorithm Optimization: The algorithm used to minimize the loss function aims to find the set of parameters that result in the smallest possible loss value, known as the global minimum, on the given dataset within a reasonable amount of time.

To compare different machine learning systems, researchers typically fix the dataset and evaluation objective function, and vary the architecture, loss function, and optimization algorithm. This enables a system to obtain a score on a particular dataset relative to other systems. However, it is important to note that each dataset may only provide a limited view of the capabilities of a particular system, and it is common to test systems on multiple datasets.

1.4.5 Fitting and Generalization

In machine learning, there are two main challenges to achieving good generalization to new data: underfitting and overfitting. Underfitting occurs when the system is unable to capture the implicit function being learned, often due to an insufficiently powerful architecture. If a linear network is attempting to learn a nonlinear regression function, no matter how effective our learning methods are, the chosen architecture will not be able to compute the desired function with any set of model parameters.

Another possible reason for this occurrence is when the learning methods cannot identify appropriate parameters for the selected architecture, despite their existence. Recent research

has shown that shallow networks can often perform as well as deep networks [13]. However, deep networks are still preferred in most cases because it may be difficult or impossible to learn shallow networks directly from the data. This is an example of a situation where the architecture, specifically the shallow network, is theoretically capable of representing the desired function. However, traditional learning methods are ineffective in training it. Instead, it can only be trained by attempting to mimic the deep network.

Overfitting happens when a model learns relationships in the training data that are not useful for the problem at hand. This occurs when the architecture is too powerful for the dataset, causing it to learn idiosyncratic features about specific stimuli instead of general features that apply to new stimuli. An example of extreme memorization is when a learner memorizes every training example and returns the correct answer for all of them during supervised training. However, when presented with new stimuli, the learner outputs random answers, indicating a lack of generalization. To avoid this, it is important to ensure that learners are able to generalize to new examples.

The characteristics of extreme behaviour can manifest in successful learners, where certain parameters in the model may adjust to accommodate idiosyncrasies in specific stimuli. As a result, the model memorises features about these stimuli to identify them, but in a way that does not generalise to other similar stimuli. Overfitting is easy to detect: it occurs when the model performs significantly worse on novel stimuli than on training stimuli. The existence of a generalization error suggests that the model has acquired characteristics from the training set that are not applicable beyond it.

Detecting underfitting can be challenging. One simple measure is to check if the network is achieving zero training error. If not, it may not be able to fully model the desired function and could be underfitting to some degree. However, this assumes that the training data is perfect, with all labels being correct and unambiguous. It also assumes that current architectures are powerful enough to capture all the nuances of the target function, which is often not the case.

Another method of detecting underfitting is to assume that any model that is not overfitting is underfitting. This is a reasonable assumption because there is a narrow range in which a model can perfectly capture a relationship; it is typically either underfitting or overfitting. Moreover, overfitting is not entirely negative; enhancing the training performance on an already overfitting model can still improve the performance on new examples. Overfitting simply indicates that the model is starting to detect false correlations in the training data.

Additional training may still assist in identifying some genuine correlations, as well as more false ones.

To address underfitting, either make the architecture more powerful (e.g., increase the number of parameters), or improve the learning methods so that can find better parameter values. Early machine learning often failed even on relatively simple datasets due to the limited computing power that prevented the use of larger models. Convolutional neural networks have been successful because they simplify the learning of parameters.

To address overfitting, the architecture needs to be limited in some way so that it learns more good patterns within the data and fewer spurious ones. Alternatively, the dataset could be expanded so that learning spurious correlations is less advantageous. In some cases, when the model is too large for the dataset, reducing the number of parameters can help.

This is especially effective when incorporating prior knowledge of the problem. However, it is important to note that the number of parameters can only be reduced to a certain extent before the model begins to underfit. At this point, it becomes necessary to limit the parameter values, which is known as regularization.

Simpler forms of regularization place a cost on larger parameter values, preventing the learner from overemphasising certain correlations and relying too heavily on any one feature. Dropout is a form of regularization for neural networks that probabilistically silences neurons, preventing the learner from relying solely on a few features. Finally, expanding the dataset can help to prevent overfitting. This is because the learner is required to perform well on a larger number of training examples, which places more constraints on the parameters.

1.4.6 Benchmark Dataset

The dataset has two main uses: training the system to find the optimal parameter values and testing the system to evaluate the parameter values. The ultimate goal is to determine how well a learner generalizes, or performs when exposed to stimuli it has never encountered before. In supervised learning, datasets are usually divided into separate training and testing subsets. This allows the learner to be exclusively tested on examples it has not seen during training.

Hyperparameters, such as learning rates, model size, and initial parameter values, are not learned but must be chosen a priori. Choosing these parameters without knowledge of the test

set is crucial as they can greatly affect the generalization error. Otherwise, hyperparameters may be tuned to the specific test set, resulting in a poor estimator of generalization error as the learner may perform less effectively on examples outside the test set.

To ensure accurate model training, it is common practice to reserve a subset of the data for validation purposes. This subset, known as the validation set, can be used to fine-tune hyperparameters. For instance, a simple approach to finding the optimal learning rate involves testing multiple rates and selecting the one that yields the best performance on the validation set. The resulting model can then be applied to the test set, providing a reliable estimate of generalization error.

Just like the brain has many synapses to solve complex problems, modern machine learning methods have numerous parameters to make them powerful enough to solve real-world problems. To adjust numerous parameters, it is crucial to have datasets that are large enough to encompass a broad range of stimuli that one may encounter in the real world.

For instance, consider all the images that could be captured with a small, low-resolution camera (e.g. 256 x 256 pixels). This encompasses every possible variation of common objects, viewed from any angle and under any lighting condition. Additionally, since each object would only occupy a small portion of the frame, all possible combinations of these objects could be included. The number of potential images is astronomical.

Dataset augmentation is the process of expanding the number of training samples by creating variations on each sample using a core dataset. This can be achieved by including small translations (shifts) of each input, as well as left-right flips. The aim is to increase the diversity of the dataset, which can significantly help reduce the chances of overfitting and better characterize the stimulus space.

Collecting and annotating datasets for supervised learning of object classification is a challenging task. To support the development of new machine learning models and methods, and to facilitate comparison between them, there are several standard datasets publicly available and commonly used in the machine learning community. These datasets are among the most common object classification datasets.

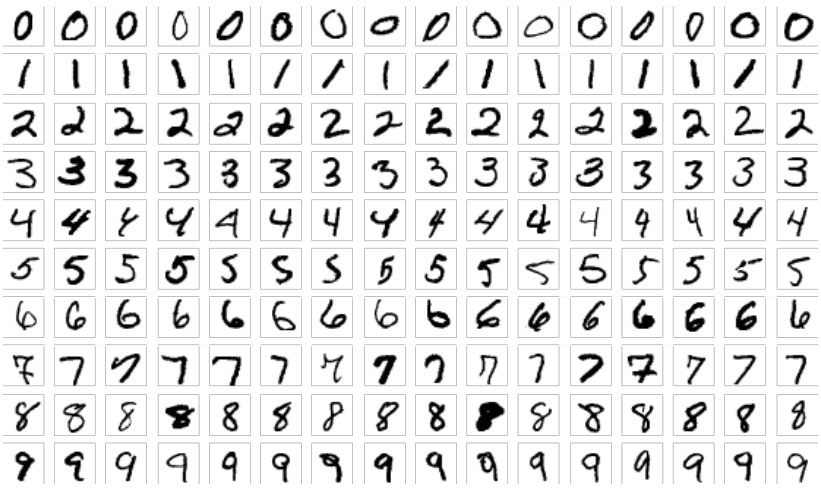

Fig. 1.10 Sample images from MNIST test dataset [187].

MNIST: This is a collection of handwritten digits taken from US zip codes written on envelopes [187]. The images have been heavily preprocessed to ensure that each one contains exactly one centred digit in high-contrast greyscale, essentially black and white. Optical character recognition of handwritten digits is a challenging task. The first successful attempt at this problem marked a turning point in machine learning. Although the dataset is no longer considered challenging for state-of-the-art methods, it is still commonly used as a benchmark for new models and methods due to its relatively small size. This allows for efficient training even with slower methods such as online learning and spike-based methods.

SVHN: This dataset is comprised of digits extracted from Google Street View images of house numbers [264]. The styles of digits, as well as the backgrounds, lighting, colours, and contrast, exhibit significant variation. The images have been preprocessed to center on the target digit for classification, but they may also contain whole or partial distractor digits to the sides. To ensure clarity, technical term abbreviations will be explained when first used. This dataset is considerably more challenging than MNIST, but still relatively easy compared to datasets composed of objects.

CIFAR: The CIFAR-10 and CIFAR-100 datasets comprise small images (32×32 pixels) from 10 or 100 categories, respectively [174]. All images in this dataset are in full colour and were taken from the Tiny Images Dataset [340]. Although the dataset is the same size as the SVHN dataset, it is considerably more challenging. Human performance on CIFAR-10 has been estimated to be 94% accurate [158]. CIFAR-100 is even more challenging due to the increased diversity of object categories. Each category encompasses a wide range of visual

features and has a larger variety of object poses. For instance, CIFAR-10 includes a category named 'dog', which comprises images of various dog breeds captured from different angles and with diverse appearances. CIFAR-100 poses an additional challenge as it contains more categories, making it less likely to guess the correct one by chance. Furthermore, many of the categories are similar, such as 'shrew' and 'mouse', and it is challenging to distinguish between them consistently using small images taken from various angles.

ImageNet: The ILSVRC-2012 dataset, also referred to as the ImageNet dataset, comprises images from the ImageNet database [296]. It was utilised for the 2012 ImageNet Large Scale Visual Recognition Challenge. The ImageNet database consists of medium to high resolution images that are organised into hierarchical categories called synsets (short for 'synonym set'). It contains millions of annotated images. The ILSVRC-2012 dataset is a subset of images used for the 2012 ImageNet Large Scale Visual Recognition Challenge competition. It contains images tagged with 1000 different categories, mainly consisting of animal species, plants, household objects, vehicles, and building/room types (both exterior and interior). The images vary in size, and pre-processing often involves scaling and cropping them to 256×256 pixels. This dataset is more challenging than the previous ones due to the increased number of categories and the presence of multiple objects in the images. It can be ambiguous to determine the primary object of focus in some cases.

Modern machine learning algorithms have performed well in addressing these challenges. Many algorithms have published both top-5 and top-1 results, where top-5 indicates that the image is considered correctly classified if the actual label for the test image is in any of the top five categories predicted by the algorithm. Similarly, top-1 means that the top prediction must match the correct label. Top-5 classification is useful for handling ambiguous images with multiple featured objects and for distinguishing between similar categories that may be difficult to differentiate from all angles, such as certain breeds of dogs.

1.4.7 Network Architectures

The two primary building elements of fully linked artificial neural networks (ANNs) are synapses and neurons. The neurons add up the incoming scaled signals and then convert the sum, while the synapses scale the signals. Neural networks nowadays consist of multilayer perceptrons (MLPs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs), as Figure 1.11 illustrates.
Neurons are shown as circles in the picture, and the connections between those neurons are represented as synapses, also known as weights. The signal travels to the right until it reaches

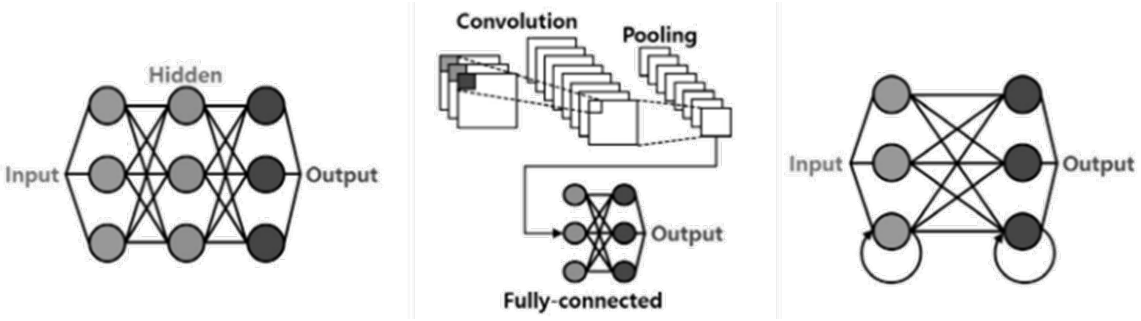

Fig. 1.11 The backbone architectures of neural networks, left to right: multilayer perceptron (MLP), convolutional neural networks (CNN), and recurrent neural networks (RNN) [324].

the output layer after inputs are provided to a neural layer known as the input layer. There are usually several hidden layers between the input and output layers. In addition, bias neurons are often present in both the input layer and the hidden layers. Lastly, synaptic layers are the collections of synapses that connect two nearby neural layers.

More formally, MLPs are back-propagation-trained basic feedforward deep neural networks having an input layer, one or more hidden layers, and an output layer [275]. For MLPs with 3–10 neuron layers, straightforward classification tasks like handwriting pattern recognition are straightforward application cases. Color pictures, which are composed of three 2D drawings with changing pixel intensities in the three recognised colour channels of red, green, and blue, are one type of data that CNNs are designed to function with [232].

Some of the inherent characteristics of these signals that CNNs may employ include connectivity, weight sharing, pooling, and the use of many layers. This led to CNNs making a number of useful advancements and seeing extensive application in the field of computer vision [142]. Typically, 5-100 layer CNN architectures are used for visual processing tasks like facial recognition and image segmentation [333].

RNNs are constructed as networks of neurons with these extra connections, based on the same notion as neural networks with additional feedback loops [90]. The availability of these additional feedback loops contributes to RNNs' capacity to retain information while processing incoming inputs [213]. This architecture is suitable for tasks that need processing data while taking into account older inputs, such historical data, but still dealing with present input [249].

This architecture allows for the training of many behaviours, including those involved in sequential processing tasks, that are not conducive to learning using conventional neural net-

working learning techniques [409]. RNNs have therefore been used in many data processing applications, such as voice recognition, translations, and natural-language processing [109].

In summary, dramatic developments in neural network technology have resulted in several computer breakthroughs that are both more energy efficient and inspired by the human brain [214]. These modifications affect not just software but also physical computer architecture and hardware [99]. New kinds of devices with far superior qualities to traditional memory devices are being investigated extensively [259], with memristor devices being regarded as neuromorphic device options for neural networks [160]. Memristor devices have been shown to offer favourable characteristics for various architectures [288].

1.4.8 Matrix-Vector Multiplication

A weighted bipartite graph is formed by each synaptic layer and the two neuronal layers that encircle it. The nodes in the two neuronal layers are the neurons, and the edges are the synapses in the synaptic layer. A vector matrix product can be used to summarise the summation function of neurons and the scaling function of synapses, $\mathbf{y} := \mathbf{xW}$. Here, $\mathbf{W}$ is a weight matrix, $\mathbf{y}$ is a vector holding the synaptic layer's outputs, and $\mathbf{x}$ is a vector holding the inputs to the synaptic layer.

As was previously established for perceptron, bias neurons, which typically have a constant input of 1, are found in presynaptic layers. The bias neuron's weights may be expressed as a vector since an input of 1 is applied. This modified the equation to $\mathbf{y} := \mathbf{xW} + \mathbf{b}$, where $\mathbf{b}$ is a vector that represents the bias neuron's associated synaptic strengths. On the other hand, the bias weights may be added to the weights matrix $\mathbf{W}$, and the input of 1 can be merged with the remaining inputs, $\mathbf{x}$, for each given layer. This will often be done implicitly and resulted in the form below (1.1).

$$\begin{matrix} & \mathbf{W} & & & & \mathbf{x} & = & \mathbf{y} \\ \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,m} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{bmatrix} & & & & \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} & = & \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \end{matrix} \tag{1.1}$$

As mentioned before, the postsynaptic neurons usually change the signals in addition to summing. Applying an activation function does this. The $(i+1)^{th}$ synaptic layer's inputs can be stated as follows:

$$x_{i+1} = \sigma_i (x_i W_i + b_i) \tag{1.2}$$

The notation x_{i+1} and x_i represents the inputs to the $(i+1)^{th}$ and i^{th} synaptic layers, respectively. The weight matrix of the i^{th} synaptic layer, W_i, is represented by the vector b_i, which is a vector of weights connected to the bias neuron in the presynaptic neuronal layer of the i^{th} synaptic layer. The vector-valued activation function associated with the postsynaptic neuronal layer of the i^{th} synaptic layer is represented by σ_i. Assuming that there are two fully connected layers ANN, the output is as follows:

$$y = \sigma_2(x_2 W_2 + b_2) = \sigma_2(\sigma_1(x_1 W_1 + b_1)W_2 + b_2) \tag{1.3}$$

By adding nonlinearities, activation functions are utilised to expand network capacity [283]. The logistic function (1.4), the hyperbolic tangent (1.5), the rectified linear unit (ReLU) (1.6), and the leaky ReLU (1.7) are the most often used options for activation functions [407]; corresponding scalar formulas for these functions are provided below:

$$g(z) = \frac{1}{1+e^{-z}} \tag{1.4}$$

$$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{1.5}$$

$$g(z) = max\{0, z\} \tag{1.6}$$

$$g(z) = max\{\varepsilon z, z\} \tag{1.7}$$

The existence of an activation function in an ANN determines the requirement for several hidden layers. This is due to the fact that a single linear transformation may be written as a sequence of linear transformations, which are represented as linear products. Take an ANN with two synaptic layers as an example. If no activation functions were used, its outputs may be represented as a single vector-matrix product with bias added.

The network's capacity is increased by non-linear activation functions, which stop several weight matrices (and bias vectors) from "collapsing" into one. Furthermore, if the network is big enough, an ANN with non-polynomial activation functions may accurately estimate any

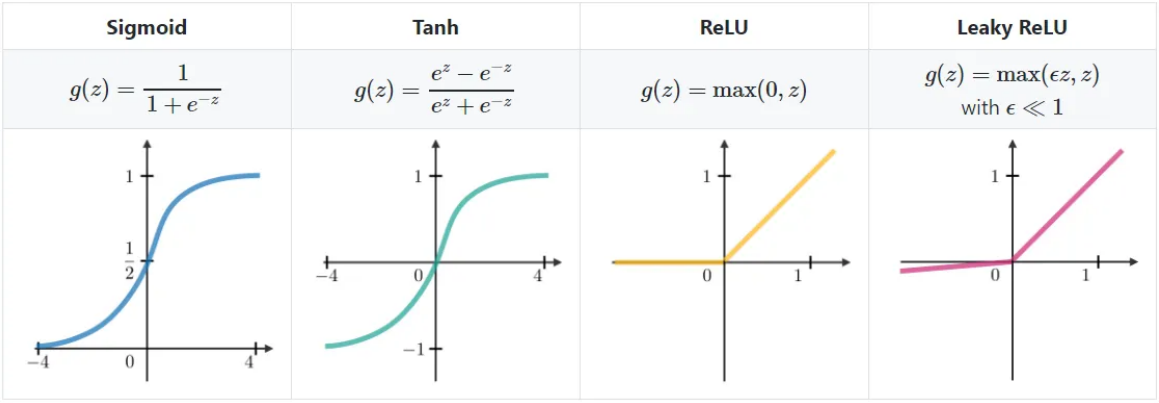

Fig. 1.12 Activation functions commonly used in ANN with the expressions labelled.

continuous function to any degree [193]. Neural networks may therefore be used to fit data of unlimited complexity.

1.4.9 Training and Inference

ANNs are often used for applications like regression and classification. To illustrate, a neural network may be used to classify photographs of handwritten digits. The picture's pixels are translated into a vector $\mathbf{x}$, which is then fed into the ANN, which generates a vector $\mathbf{y}$ with ten elements, each representing the likelihood that the image represents a certain digit.

In general, this process is known as inference, and it involves determining a class (or a specific value in the case of regression) based on the inputs $\mathbf{X}$. The outputs of ANN (or any other supervised learning model) utilised in inference may be stated as $\mathbf{Y} = f(\mathbf{X}, \mathbf{W})$. In the context of artificial neural networks (ANNs), the model parameters, represented by $\mathbf{W}$, represent synaptic weights. The recursive formula of equation (1.2) may be used to fully connected ANNs with n synaptic layers to achieve the specified form.

A supervised learning model's parameters $\mathbf{W}$ are determined via a process known as training [25]. The goal of this procedure is to match certain training input data $\mathbf{X}_{train}$ to training target data $\mathbf{T}_{train}$. Once a model has been trained, its outputs $\mathbf{Y}$ should be able to predict test targets $\mathbf{T}_{test}$ from a set of test inputs $\mathbf{X}_{test}$. Training usually requires specifying a loss function, which defines the closeness of the outputs to the target data and then seeks to reduce it. The squared-error (1.8) and cross-entropy (1.9) loss functions are two of the most commonly

used [169].

$$E(\mathbf{W}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{p=1}^{P}\left([\mathbf{f}(\mathbf{X}_{m,*},\mathbf{W})]_p - \mathbf{T}_{m,p}\right)^2 \tag{1.8}$$

$$E(\mathbf{W}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{p=1}^{P}\mathbf{T}_{m,p} ln\left([\mathbf{f}(\mathbf{X}_{m,*},\mathbf{W})]_p\right) \tag{1.9}$$

The feed-forward inference function, represented by **f**, accepts input data from the matrix $\mathbf{X} \in \mathbb{R}^{M\times N}$, where **M** is the number of instances, **N** is the number of inputs, and **P** is the number of classes. Performance of the function **f** is evaluated using target data $\mathbf{T} \in \mathbb{R}^{M\times P}$. In supervised learning, the network's objective or loss function determines the quantity to optimize, linking the actual outputs to the desired outputs. It is commonly referred to as a cost or loss that we aim to minimize. This cost reflects how poorly the network performs on the training dataset.

A crucial differentiation exists between loss functions and error functions. In regression problems, these functions are often identical, but in classification problems, they differ. An error function measures the amount of error a network produces. In classification, this is usually the number of incorrectly classified examples divided by the total number of examples. A sample is deemed incorrectly classified if the output with the highest value is not the correct output.

The loss function for classification is dependent on the output values. For instance, the function may impose a penalty (i.e. non-zero loss) on an example if the network output for the correct class is not as high as intended, or if the margin between the correct output and the next most active output is not as large as intended. This implies that even correctly classified examples may still have a non-zero loss.

Overall, this pushes the classifier to not only get the answer right, but to get it right by a significant margin, ideally making the classifier more robust and better able to generalise to new inputs. The loss function is also continuous and differentiable, whereas the error function can be discrete; this makes the loss function easier to optimise, as the derivative(s) can be used in the optimisation.

1.4.10 Backpropagation Algorithm

The backpropagation (BP) algorithm, also known as 'backwards propagation of gradients' or 'backprop', is the algorithm responsible for the recent success of machines in object recognition tasks. Although the algorithm was introduced in the 1980s [295], it was only mildly successful at the time due to a lack of computational resources. As computers developed and machine learning became better understood, backpropagation became increasingly useful. This culminated in LeNet [187], a deep convolutional network that marked the first significant success on the MNIST handwritten digit dataset and a seminal success of neural networks in general.

Since then, the algorithm has gained popularity. In the 2000s, backpropagation was used to fine-tune various types of deep networks. It was applied at the end of the training algorithm to make moderate adjustments to network weights, resulting in a significant decrease in error at the end of the training regime. Other algorithms were used for pretraining, which involved determining initial weights that could be moderately successful at a task and provide a starting point for backpropagation.

Layer-wise pretraining is a method in which each layer of a deep network is trained as an autoencoder or restricted Boltzmann machine (RBM) to represent the information in the previous layer. This unsupervised training initializes the network to a state where the final hidden layer retains a significant amount of input information. Backpropagation can then be used to train the output classifier and fine-tune the weights of the previous layers.

Backpropagation has just recently—roughly since 2010—been utilised to train networks from start. This is made feasible by the convergence of three key components. It was possible to train larger models on larger datasets in an acceptable period of time by increasing computing power, particularly with GPUs. Rectified linear units (ReLUs), which are scale-invariant and reduce the likelihood of vanishing and inflating gradient issues, making training less sensitive to the initial weights. When paired with these techniques, convolutional networks—which had been effectively applied a decade earlier—markedly reduced the number of parameters when compared to fully linked techniques like RBMs.

Presently, nearly all deep networks are trained using backpropagation as the primary workhorse. Backpropagation is a technique used to calculate the gradients of the network parameters in relation to an objective function. It solves the spatial credit assignment problem, which involves identifying the hidden units responsible for a particular error at

the network output and determining which parameters should be adjusted to reduce the error most efficiently. In other words, it assigns credit or blame to the hidden units for their contribution to the network output. This was a significant challenge for early neural network researchers.

1.4.11 Stochastic Gradient Descent

Backpropagation offers a method for obtaining the first-order derivative (gradient) of the cost function concerning the network parameters. However, it does not specify how to utilize this gradient to minimize the cost. There are several potential methods to achieve this. The simplest approach involves adjusting the parameters in the direction of the negative gradient. The gradient indicates the direction in which the cost increases most rapidly. To decrease the cost quickly, it is moved in the direction of the negative gradient. This process is known as gradient descent.

Given the complexity of ANNs and data, it is impossible to analytically calculate the set of weights W that minimises $E(\mathbf{W})$. Gradient descent is typically used for this type of minimization. At time step τ, the gradient $\nabla_i E$ of the loss function with respect to weights in the i^{th} synaptic layer is determined. Weights are then gently changed in the opposite direction of the gradient. This technique is done recursively several times and may be formalised using the recursive formula (1.10), where $\eta \in \mathbb{R}_{>0}$.

$$\mathbf{W}_i^{\tau+1} = \mathbf{W}_i^{\tau} - \eta \nabla_i E\left(\mathbf{W}^{\tau}\right) \tag{1.10}$$

By definition, taking only a very small step in this direction will decrease the cost, as the function may only decrease for a very short distance before starting to increase again. However, in practice, we must take a significant step in the gradient direction. The step size is modulated by the learning rate, which is multiplied by the gradient to determine the step. There is a trade-off between a large and small learning rate. A large learning rate allows for quicker convergence on smooth, well-conditioned objectives, while a small learning rate ensures stability and convergence on more ill-conditioned problems. As long as the learning rate is sufficiently small, gradient descent can converge for any problem.

During gradient descent, the gradient is computed across all training samples (known as the batch) for each parameter update, which can be computationally expensive. Stochastic gradient descent (SGD) addresses this issue by using only a portion of the training set to estimate the gradient at each iteration, with the understanding that this may occasionally

increase the overall cost. At each iteration, a noisy or stochastic estimate of the gradient is computed. If the corresponding parameter updates are beneficial on average, the overall cost should decrease. The mini-batch is the set of examples used to estimate the gradient.

The mini-batch size is a crucial parameter in SGD. A larger mini-batch provides a more accurate gradient estimate, but also requires more computation per mini-batch. Therefore, selecting the mini-batch size involves balancing these two criteria. In practice, mini-batches usually contain 20 to 100 examples. SGD is a widely used method due to its simplicity and effectiveness on various problems.

Most neural networks are too large and complex for second-order methods, which require the explicit computation of second-order derivatives (known as the Hessian) and are therefore not feasible. Even methods that estimate the second-order derivatives, such as L-BFGS, cannot handle the size of modern neural networks, as the number of elements in the Hessian is equal to the number of possible pairs of parameters in the model [208]. Hessian-free optimization can avoid the vanishing and exploding gradient problems that are particularly potent in recurrent neural networks [230], making it a popular choice for improving their performance.

1.4.12 Gradients Initialization

During artificial neural network (ANN) optimization, two common issues arise: the vanishing and exploding gradient problems. These problems become more prevalent as networks become deeper or recurrent, as recurrent networks are often optimized like deep networks with tied parameters between layers. The vanishing gradient problem occurs when the gradient's magnitude approaches zero, causing the optimization process to slow down significantly.

A saddle-point in the parameter space is indicated, which is a common issue when optimizing ANNs [58]. If the gradient of one layer approaches zero prematurely, such as when most neurons in a layer become inactive, then the gradients for other layers will also quickly approach zero. This is because the cost can only be reduced to a certain extent when one layer is not functioning.

The problem of exploding gradients arises when the gradients become too large too quickly. This is usually due to instability in the SGD algorithm caused by a learning rate that is too high. Once the algorithm becomes even slightly unstable, it is difficult for it to regain stability. For instance, a high learning rate can cause the algorithm to overshoot the minimum in the

gradient direction and end up in an area of higher cost. This higher cost leads to the next gradient, and consequently the next step, being too large, resulting in an even higher cost. As a result, the algorithm spirals out of control.

The vanishing and exploding gradient problems may arise from improper network initialization. Layers with excessively large or small parameters can cause issues, particularly when using sigmoid nonlinearities, which have zero derivatives for both large negative and positive values. Several initialization schemes have been developed to mitigate these problems. Several initialization schemes have been developed to mitigate these problems. The concept underlying these methods is that the variance information transmitted both forwards (activations) and backwards (derivatives) through the network should remain constant.

1.5 Thesis Outline

Chapter 2

Neuromorphic Modelling

2.1 Introduction

The brain is capable of performing a vast array of computations, with the fundamental units of the brain generally considered to be neurons and synapses. In the context of the nervous system, a synapse is defined as a structure that is capable of facilitating the transfer of an electrical or chemical signal from a presynaptic neuron to a postsynaptic neuron.

In the case of a chemical synapse, the occurrence of a spike in the presynaptic neuron will result in the stimulation of the release of a chemical substance known as a neurotransmitter. Subsequently, the neurotransmitter will migrate to bind with the receptors in the membrane of the postsynaptic neuron.

The effects of different neurotransmitters vary, exerting excitatory or inhibitory effects on the postsynaptic neurons. Many receptors contain different ion channels, which permit ions to flow between the inside and outside of the membrane, thereby generating an excitatory postsynaptic current (EPSC) or an inhibitory postsynaptic current (IPSC).

This chapter offers a concise overview of the pertinent biological details, along with the concepts and models from computational neuroscience that are employed or expanded upon in this study. These details provide invaluable preliminary information for accurately modelling the implementation of silicon oxide device-based neuromorphic hardware and bio-inspired computing.

2.1.1 Neuroscience Primers

Computational neuroscience employs a computational methodology to elucidate the mechanisms underlying brain function. This entails not only identifying the computations performed by the brain but also understanding the interactions between brain elements, such as neurons and synapses, that facilitate these computations.

The field of computational neuroscience has traditionally adopted a bottom-up approach, commencing with the study of individual neurons and subsequently examining how these can be integrated into progressively larger networks, ultimately leading to the emergence of meaningful behaviour. In contrast, psychology and cognitive science have adopted a more top-down approach, commencing with the observation of animal behaviour and subsequently attempting to elucidate the brain mechanisms underlying these behaviours in terms of the higher-level functions of large brain areas.

This distinction has resulted in a historical focus on developing comprehensive mathematical models of individual neurons and small- to medium-sized networks in computational neuroscience. Recently, however, there has been a convergence with other fields, leading to the creation of neurally detailed models capable of reproducing organism-level behaviors.

This section provides some relevant biological details, both at the neural level and at the network level. It should be noted that the models detailed in this dissertaion are still relatively basic in comparison to the extensive body of evidence documenting the neural system [154] which provide the majority of our neural data about these areas. Consequently, this section presents only the most fundamental biological facts relevant to this work.

Neurons represent only one of the numerous cell types within the brain, yet they are the most frequently discussed due to their status as the primary computational entities. Their fundamental function is relatively straightforward: neurons receive input from other neurons, and if that input is sufficiently stimulating, they will fire an action potential (also known as a spike), which propagates to other neurons.

Figure 2.1 illustrates a basic cartoon of a neuron. Neurons can be subdivided into three principal parts: the dendrites, the cell body (soma), and the axon. Neurons receive input currents via their dendrites, which then transmit or channel this into the cell body, called the soma. When a neuron spikes, it sends current down its axon, which results in the release of neurotransmitter(s) at the synapses. These are connections from a neuron's axon to the

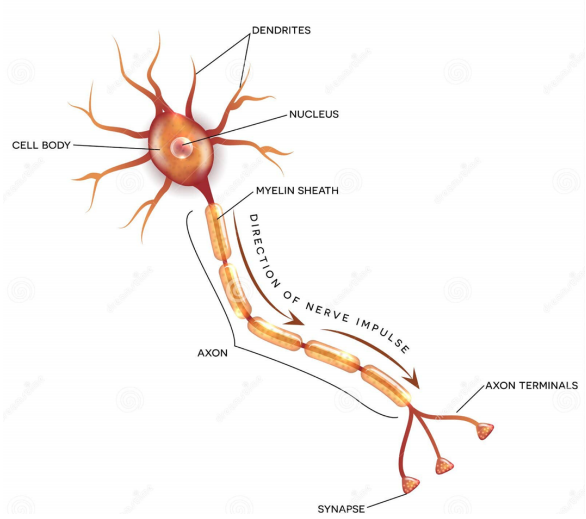

Fig. 2.1 Labeled diagram of the neuron, nerve cell that is the main part of the nervous system. A neuron's dendrites include synapses that allow it to accept input from other neurons. The dendrites carry current to the soma, which is where electrical charge is integrated. If the neuron membrane gets sufficiently polarised, an action potential (also known as a spike) travels down the axon. This causes neurotransmitters to be released at synapses, resulting in currents in the dendrites of postsynaptic neurons.

dendrites of other neurons, and the neurotransmitter release causes dendritic input currents in these other connected neurons.

The soma represents the principal component of the neuron. From a computational perspective, the soma represents the integration point for all incoming currents from dendrites, marking the initiation of the action potential generation process (Figure 2.2). When a neuron is at rest, the soma exhibits a negative charge. This is referred to as the resting voltage and is maintained by ion pumps that regulate the concentration of ions (predominantly sodium, Na^{+}, potassium, K^{+}, and calcium, Ca^{+2}) within the cell.

As the currents arrive from the dendrites, they initiate a process of depolarisation of the cell. Once the voltage within the soma reaches a sufficient level, it initiates the opening of voltage-activated sodium channels, which permit the influx of sodium ions into the cell, further depolarising it. This process persists until the electrical gradient resulting from the accumulation of sodium ions reaches a point where it is no longer in equilibrium with the chemical gradient caused by the imbalance of sodium within and outside the cell. This leads to a notable increase in the neuron's positive charge, exceeding the resting voltage.

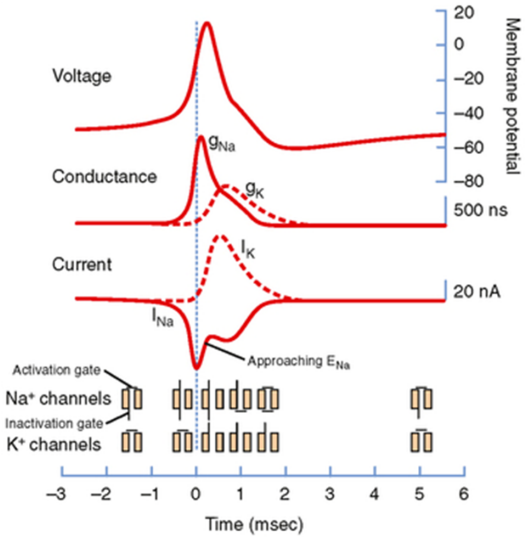

Fig. 2.2 The action potential and the underlying conductance and currents with respect to time [321]. It should be noted that the increased conductance for Na^+ (and its inward flow) is associated with the rising phase of the action potential, whereas the slower increase in conductance for K^+ (and its outward flow) is associated with repolarisation of the membrane and with afterhyperpolarisation. The reduction in I_{Na} before the peak of the action potential (even though G_{Na} is still high) is due to inactivation of the Na+ channels.

Furthermore, this substantial depolarisation also activates voltage-gated potassium channels, which subsequently permit the release of potassium ions from the cell, thereby facilitating repolarisation. Concurrently, the sodium channels undergo inactivation. The opening of potassium channels ultimately results in the cell reaching a voltage below its resting level, a state known as hyperpolarisation. The sodium channels remain inactivated and the potassium channels remain open for a period of time following the spike.

The combination of these factors renders it almost impossible for the neuron to fire during this time; this is referred to as the absolute refractory period. The change in ionic concentrations within the cell is relatively minor during a single spike, but over the course of numerous spikes, the ion pumps are required to maintain the optimal concentrations of sodium and potassium. Other currents, most notably calcium currents, are present in some neurons.

The rapid depolarisation associated with an action potential not only causes an increase in the somatic voltage potential, but also results in partial depolarisation of the axon segments situated in closer proximity to the soma. This results in the opening of sodium channels in that part of the axon, which in turn causes further depolarisation and the opening of sodium channels in the subsequent section of the axon. In this way, the somatic spike triggers a volt-

age wave that travels down the axon, eventually leading to the release of neurotransmitter(s) from synaptic vesicles situated near the ends of the axon.

Axons are responsible for transmitting long-range signals in the brain, and thus exhibit considerable variation in length, contingent on whether a given neuron is connected to neighbouring neurons or to neurons situated in a different brain region. To facilitate long-range transmission, axons are coated in myelin, a substance composed mainly of lipids and thus a good electrical insulator. This enables the propagation of current down the axon. The high proportion of fat makes myelin white; bundles of axons are responsible for the "white matter" parts of the brain. In contrast, the "grey matter" is composed mainly of neuron dendrites, somas, and short-range axons [244].

Dendrites are thin processes that extend away from the soma to connect to the axons of other neurons [146]. They serve to transmit current from synapses with other neurons to the soma. Initially, it was thought that they only conducted current passively; however, research has demonstrated that they possess active conductance mechanisms that are analogous to those involved in spike generation and propagation down an axon.

Dendrites are also traditionally considered to act in a linear manner, whereby inputs from numerous synapses are accumulated over time and space. This remains a prevalent assumption in numerous computational models. However, recent studies have demonstrated that the summation of signals in dendrites is more intricate, exhibiting a combination of linear and nonlinear (sigmoidal) elements [282].

Neurons are connected to one another by synapses, which facilitate the connection between the axon of the presynaptic neuron and a dendrite of the postsynaptic neuron. Upon the occurrence of a spike in the presynaptic neuron, an electrical pulse is transmitted along the axon, reaching the presynaptic terminals of all synapses situated along its length. The synaptic vesicles, which are filled with neurotransmitter, are located at the terminals.

When an electrical pulse is generated, it causes the vesicles to release neurotransmitter into the synaptic cleft, which is the narrow space between the presynaptic terminal and the postsynaptic terminal. This neurotransmitter then activates receptors on the postsynaptic terminal, which open and permit the flow of current into the postsynaptic cell. The specific neurotransmitter utilized by the synapse is contingent upon the presynaptic neuron.

All synapses on a neuron's axon release the same neurotransmitter or combination of neurotransmitters, a phenomenon known as Dale's principle. At the time of its development, Dale's principle was based on the assumption that each neuron produced a single type of neurotransmitter. However, evidence of cotransmission was only discovered in 1976 [30], with the understanding that neurotransmitters can be excitatory or inhibitory. It is not the case that any neurons play both an excitatory and inhibitory role with respect to different postsynaptic cells.

A significant proportion of the most influential findings in computational neuroscience are based on mathematically detailed models of neuronal functioning. One of the most renowned of these is the Hodgkin-Huxley model of the squid giant axon [122]. In recent times, the number of available neuron models has proliferated.

The models currently in use in the literature range from the simplest possible rate-neuron model, namely binary threshold units [323], to complex multi-compartmental models that account for detailed dendritic morphologies [229]. In the context of large-scale neural models aiming to reproduce high-level behaviours, single-compartment neuron models remain the prevailing approach.

These models treat the neuron as a single electrical compartment, combining the dendrites, soma, and axon. In contrast, multi-compartmental models represent the neuron as comprising multiple electrical compartments, with equations that describe the influence of activity in one compartment on that of another.

The number of compartments in a model can vary considerably, from a minimalistic two-compartment structure, typically comprising one for the dendrites and one for the soma, to a more elaborate configuration comprising thousands of compartments. The number of compartments in a model is directly proportional to the amount of computing power required for simulation and the complexity of the resulting behaviour. For these reasons, models that seek to reproduce behaviour predominantly utilise either single-compartment neurons or simpler models that do not model the electrical activity of the neuron at all.

Single-compartmental neuron models can be classified according to two key dichotomies: firstly, the rate-based versus spike-based dichotomy; and secondly, the static versus dynamic dichotomy. The distinction between rate-based and spiking neuron models concerns the nature of the output: is it a continuous value, represented by a firing rate, or a discrete value,

represented by a spike.

The majority of neurons in the mammalian cortex communicate via spikes, making spiking neuron models more physiologically realistic. However, there are limited areas in the mammalian brain, such as horizontal cells in the retina, where neurons communicate via gap junctions, potentially transmitting a continuous value rather than a discrete spike.

The distinction between static and dynamic models concerns the extent to which the model exhibits internal dynamics. In a static model, the output at a given point in time is independent of the neuron's past history and solely contingent on its instantaneous input. In contrast, a dynamic model allows for some degree of dependence on the neuron's past trajectory.

Dynamic neuron models possess a state, which is to say, internal variables that evolve over time and are not directly computable from the present input to the model. They are typically expressed using differential equations. In contrast, static models output a function of the current input only; thus, they do not require differential equations to express them.

It is important to note that the distinction between static and dynamic neurons does not necessarily refer to whether the neuron is being used as a static nonlinearity, evaluated at a single point in time on an input value, or as a dynamic nonlinearity, evaluated over a period of time on an input signal.

Static neurons can be evaluated dynamically by evaluating them independently at each successive point in time on the input value. In contrast, there is no general method for evaluating dynamic neurons statically, as their input at any given point in time is contingent upon their internal state, which is a dynamic process that evolves over time.

In some cases, it is possible to determine a firing rate response curve, also known as an I-F response curve or rate response function, for dynamic neuron models. This curve maps every constant value of the input current to a constant firing rate output. This can be determined analytically or empirically by applying a constant input current and measuring the rate of the output spikes, this can even be done in real neurons in vitro.

However, many neuron models will not output spikes at a constant rate, even for a constant input, due to changing internal dynamics. In such cases, the rate-response function only captures part of the model, and will be different depending on the state of the model and how

Table 2.1 Neuron Model Dichotomies.

	Rate	**Spiking**
Static	Sigmoid Rate LIF	Poisson Spiking
Dynamic	Adaptive Rate LIF Sigmoid with State	Hodgkin-Huxley LIF

it is measured or calculated.

Table 2.1 illustrates the manner in which a small selection of neuron models align with these dichotomies. In the static rate category, the models in question take in a continuous value and output a continuous function of that value. This encompasses the majority of non-linearities employed in machine learning (such as sigmoids or rectified linear units), in addition to the analytic firing rate for the LIF neuron model.

In the static spiking category, models are characterised by the output of a discrete (binary) function of their continuous input. This category is primarily composed of Poisson-spiking neurons, which fire probabilistically based on their instantaneous firing rate. This implies that the probability of firing at a given time is independent of past firings. Furthermore, Poisson spiking can be applied to a dynamic rate model, whereby the probability of spiking is independent, but the spike rate is a dynamic process.

Dynamic rate models possess an internal state that influences their continuous output, in addition to the input. This encompasses the analytic LIF rate model with adaptation, which possesses an intrinsic adaptation parameter that increases when the neuron is active and discounts the firing rate. Furthermore, the category encompasses sigmoid neuron models that possess an underlying voltage that is dynamically correlated with the neuron inputs [204].

The dynamic spiking category also includes the LIF neuron model, the Hodgkin-Huxley model, and more intricate multi-compartmental models. Within the category of spiking models, a distinction can be made between those that output instantaneous spikes (e.g., LIF) and those that output spikes that vary across time (e.g., Hodgkin-Huxley). Recently, another study have employed such models in the context of deep networks [101], subject to biological constraints.

Another potential distinction can be made between models that output stereotyped spikes, where the size of each spike is identical, and those that output variable-sized spikes. Models

of cortical neurons that output instantaneous spikes (e.g., LIF) almost invariably output stereotyped spikes. This is due to the fact that spikes in cortical neurons are almost identical in terms of magnitude, duration, and shape.

It is widely believed that none of these factors carry information. Even among models that have spikes that vary across time (e.g., Hodgkin-Huxley), the spike magnitude, duration, and shape are quite consistent between spikes. Therefore, the majority of spiking cortical neuron models exhibit stereotyped spikes.

By modelling the spike separately from the rest of the neural dynamics, it is possible to separate time scales, thereby avoiding the need for additional computational resources to model the spike trajectory, which is characterised by stereotyped details of little interest [1].

2.1.2 Neuron Models

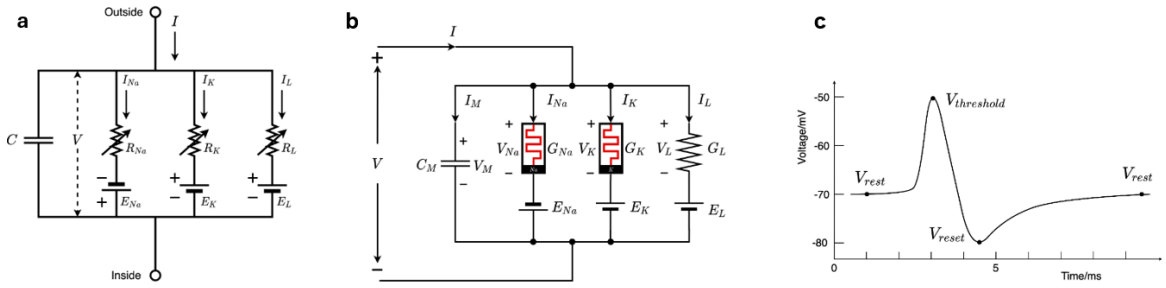

Fig. 2.3 Hodgkin-Huxley neuron model. (a) An equivalent circuit for the HH models [122]. (b) An equivalent circuit for memristive HH model [51]. (c) An action potential waveform, which demonstrates the resting, threshold, and reset potentials.

A variety of neuron models have been developed to describe the complex dynamics of biological neurons. Among these, the Hodgkin–Huxley (HH) model is one of the most widely used, comprising a set of nonlinear differential equations that accurately approximate the electrical signals of neurons.

Figure 2.3(a) depicts the HH neural model, wherein the time-varying nonlinear conductor $R_{Na}(GNa)$ and $R_K(G_K)$ represent the sodium and potassium channels, respectively, while the linear conductor $R_L(G_L)$ simulates leak channels and C models the membrane of a neuron.

The equations of the HH model are presented below:

$$C\frac{dV_m(t)}{dt} = I_C(t) + \sum_k I_k(t) \tag{2.1}$$

In this context, V_m represents the membrane potential. $\sum_k I_k(t)$ denotes the sum of the ionic currents flowing into the neuron. This can be formulated by three ion currents, as follows:

$$\sum_k I_k = C_m\frac{dV_m}{dt} + G_K n^4(V_m - V_K) + G_{Na}m^3(V_m - V_{Na}) + G_L(V_m - V_L) \tag{2.2}$$

$$\frac{dn}{dt} = \alpha_n(V_m)(1-n) - \beta_n(V_m)n \tag{2.3}$$

$$\frac{dm}{dt} = \alpha_m(V_m)(1-m) - \beta_m(V_m)m \tag{2.4}$$

$$\frac{dh}{dt} = \alpha_h(V_m)(1-h) - \beta_h(V_m)h \tag{2.5}$$

The reversal potentials V_K, V_{Na}, and V_L are the three parameters in question. The rate constants α_i and β_i, which depend on the membrane potential, describe the behaviour of the i^{th} ion channel. The maximal value of the conductance is represented by G_K, G_{Na}, and G_L. Finally, the dimensionless quantities n, m, and h, which lie between 0 and 1, are associated with three ion channels.

In order to achieve the optimal fit for human cardiac action potentials, the HH model is reduced by setting the leakage channel conductance to $G_L = 0$ [265].It has been demonstrated that G_{Na} and G_K are memristors [52], as illustrated by the equivalent circuit in figure 2.3(b).

The integrate-and-fire (IF) neuron [185] constituted one of the earliest computational models of a neuron. This model was developed prior to the ability of researchers to measure the electrical and chemical changes occurring in a functioning neuron. It is based on the premise that the neuron membrane can be modelled as a capacitor that stores charge over time [1].

As the name suggests, the IF model exhibits two principal behaviours: The model integrates current over time, as would be expected of a capacitor, and fires when the voltage reaches a threshold. Furthermore, the model may or may not incorporate a leak term, which represents a resistor in parallel with the capacitor that permits the dissipation of charge over time.

The model with a leak term is typically designated as the leaky integrate-and-fire (LIF) model. While the term "integrate-and-fire (IF) model" can be applied to either the leaky or non-leaky

model, in this thesis it will be used to refer specifically to the non-leaky version of the model.

With the objective is to identify how the neuron's membrane voltage evolves over time and, based on this, to determine when the neuron spikes. The charge Q across a capacitor is given by $Q = V \times C$, where V is the voltage across the capacitor and C is the capacitance. Differentiating this with respect to time, we find that the membrane voltage *V(t)* of the neuron is governed by:

$$C\frac{dV(t)}{dt} = J(t) \tag{2.6}$$

In this context, *J(t)* represents the input current to the neuron over time, whereas C denotes the membrane capacitance. It should be noted that current is the time derivative of charge. Equation 2.6 demonstrates that the IF neuron simply integrates the input current over time. It is still necessary to identify the point at which the neuron spikes.

This is achieved by defining a threshold voltage, V_{th}, which is exceeded when the voltage passes this threshold, resulting in the neuron firing. This is a fundamental principle in neurophysiology: once the neuron voltage passes a threshold, the neuron begins firing a spike, and once this firing process begins, it is almost impossible to reverse.

A reset procedure is finally defined. Once a neuron has fired a spike, the membrane voltage is reset to the resting potential, V_{rest}. This phenomenon can be attributed to physiological processes. Following the occurrence of a spike in a neuron, other ionic currents, typically potassium, are initiated, leading to a restoration of the membrane voltage towards the resting potential.

Some integrate-and-fire models may also incorporate an absolute refractory period, defined as a time interval during which the voltage is maintained at the resting potential V_{rest} following a spike. In cortical neurons, post-spike potassium currents are sufficiently strong to prevent another spike from occurring for a considerable duration. This time interval is referred to as the absolute refractory period. The model accounts for this by holding the membrane voltage at V_{rest} for a duration equal to the absolute refractory period.

The leaky integrate-and-fire (LIF) model [170] incorporates an additional physiological factor: Neuron membranes are not perfect capacitors; rather, they slowly leak current over time, pulling the membrane voltage back to its resting potential. Therefore, the membrane

is modelled as a capacitor and resistor in parallel, which allows for the neuron to exhibit a degree of "forgetting": in the absence of any input, the membrane voltage will return to its resting potential [171]. The LIF dynamics are captured by the following equation:

$$C\frac{dV(t)}{dt} = J(t) - \frac{1}{R}(V - V_{rest}) \tag{2.7}$$

In this model, R represents the membrane resistance, and the remaining parameters are consistent with those of the IF model. The resetting procedure is also identical to that of the IF model.

The LIF model comprises a number of parameters, including C, R, V_{rest} and V_{th}. It is possible to normalise the model in order to reduce the number of parameters while maintaining the full dynamics of the original model. In particular, the model can be manipulated so that the normalised voltage lies within the range [0, 1], with a normalised resting potential of zero and a normalised firing threshold of one. Initially, Equation 2.7 is multiplied by R to give:

$$\tau_{rc}\frac{dV}{dt} = RJ(t) - V + V_{rest} \tag{2.8}$$

$$\tau_{rc} = R \times C \tag{2.9}$$

$$\bar{V} = \frac{V - V_{rest}}{V_{th} - V_{rest}} \tag{2.10}$$

$$\bar{V_{rest}} = \frac{V_{rest}}{V_{th}} \tag{2.11}$$

By substituting $\bar{V}$ and $\bar{V_{rest}}$ into equation 2.3 to give:

$$\tau_{RC}(V_{th} - V_{rest})\frac{d\bar{V}}{dt} = RJ(t) - \bar{V}(V_{th} - V_{rest}) \tag{2.12}$$

$$\tau_{rc}\frac{d\bar{V}}{dt} = \frac{R}{V_{th} - V_{rest}}J(t) - \bar{V} \tag{2.13}$$

$$\tau_{rc}\frac{d\bar{V}}{dt} = \bar{J}(t) - \bar{V} \tag{2.14}$$

When the firing threshold for the new equation $\bar{V_{th}} = 1$, the voltage resets to $\bar{V_{rest}} = 0$, and $\bar{J}(t) = \frac{R}{V_{th} - V_{rest}}J(t)$. It can be observed that $\bar{J}(t)$ is merely a linear transformation of $J(t)$. Consequently, Equation 2.14 retains the full dynamics of Equation 2.7 for a scaled input, but

with only one parameter, τ_{RC}.

It should be noted that both $\bar{V}$ and $\bar{J}$ are unitless quantities. Conventionally, the unitless space is employed exclusively, and the quantities are often referred to simply as V and J, despite the fact that they are not voltages or currents. This simplifies the mathematical representation, without limiting the generality of the models.

Equation 2.14 provides an exact description of the circumstances under which the model neuron will spike in response to a given input current, *J(t)*. However, in some cases, it is sufficient to consider only the spike rate, that is, the number of spikes per second that the neuron will produce in response to a given input current.

In the case of the LIF model, it is possible to determine the analytical firing rate for a constant input current. This is achieved by calculating the inter-spike interval (ISI), which is the time between one spike and the next. The firing rate is then given by the inverse of the ISI. When a constant input current, $J(t) = j$, is provided, it is possible to solve Equation 2.14 in order to find the neuron voltage over time.

$$V(t) = (V(0) - j)e^{\frac{-t}{\tau_{rc}}} + j \tag{2.15}$$

In the absence of spikes, the objective is to ascertain the time required for the voltage to increase from$V(0) = 0$ to $V(t) = 1$. This property will only occur if $j > 1$. Substitution into Equation 2.15 and subsequent solution for t yields:

$$t = -\tau_{RC} log\left(-\frac{1}{j}\right) \tag{2.16}$$

Incorporating the refractory period and performing the inversion, the spike rate r for the LIF neuron is given by:

$$r = \begin{cases} \frac{1}{t_{ref} - \tau_{RC} log\left(1 - \frac{1}{j}\right)} & \text{if } j > 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.17}$$

The LIF model is one of the most widely utilised simplified neuron models [186]. The simple equivalent model is illustrated in Figure 2.4(a). In this model [322], a resistor R, connected in series with a DC source V_{rest}/V_{reset}, is connected in parallel with a capacitor C.

A postsynaptic neuron receives a synaptic current *I(t)*, generated by presynaptic spikes.

A proportion of the current *I(t)* flowing into *C* results in an increase in the membrane potential *V(t)*. The charge leakage occurs via resistor *R*. When *V(t)* reaches a threshold value, the neuron generates a spike. Following the generation of a spike, the membrane potential is reset to the reset value.

In the absence of *I(t)*, the voltage across *C* is eventually settled at V_{rest}, representing the cell's resting potential. During the refractory period t_0, a neuron is incapable of spiking. Figure 2.4(c,d) illustrates the LIF neuron dynamics for the case of a *DC* input current and a zero rest and reset potential, $E_{reset} = E_{rest} = 0$ [335].

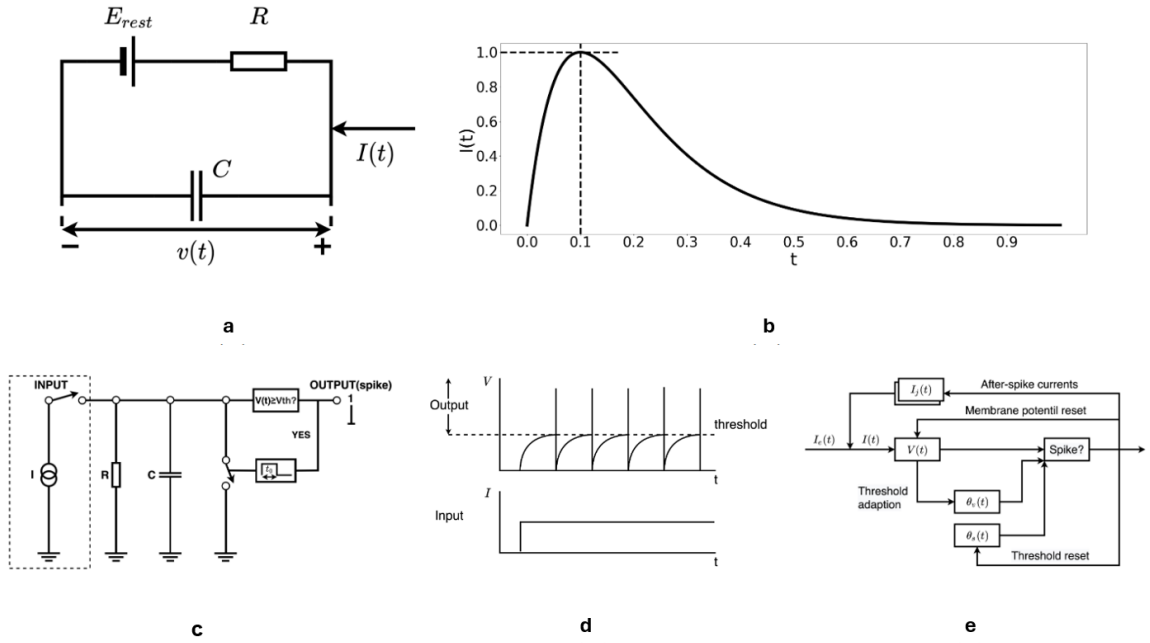

Fig. 2.4 The LIF neuron model. (a) Schematic diagram of the LIF electrical model. (b) Input current in the form of an alpha function, $\tau_\alpha = 0.1, I_0 = 1$ (c)The LIF model allows for the control of spiking behaviour through a comparison of membrane potential and threshold at each time step. Upon the triggering of a spike, a voltage-controlled switch discharges C for a duration corresponding to the refractory period t_0 [335]. (d) A simulation of constant firing frequency for DC current input in which t_0 is hidden. DC input current and output spikes are both shown. (e) A generalised LIF model with threshold control [338].

The input synaptic current, *I(t)*, can be described by a time-varying alpha function. However, alternative functions may be employed, including "Instantaneous Rise and Single Exponential Decay," "Biexponential Functions," "Sawtooth," and "Pulse Function." The alpha

synaptic current is modeled by Equation 2.15, and the resulting plot is shown in Figure 4.2(b).

Nevertheless, Figure 2.4(a) lacks a circuit for resetting the system when the threshold is reached. In order to evaluate the inequality $V > V_{threshold}$, it is necessary to use an active circuit, such as a comparator. Upon reaching the threshold, the membrane potential must be reset in accordance with the illustration in Figure 2.4(d). Therefore, the LIF model's generalized version necessitates additional overhead, as illustrated in Figure 2.4(e).

In the generalized LIF model, the reset behavior necessitates external control to pull down the membrane potential below the resting potential. This external control requires intricate, active circuits comprising MOS transistors, resistors, and even a silicon-controlled rectifier (SCR), which represents a significant drawback in a neuromorphic system due to its substantial power and area consumption.

In order to address the limitations of conventional neuron models, memristor technologies can be employed to emulate biological neurons with the objective of reducing energy consumption and increasing packing density. The HH model has been emulated by utilising two memristors in parallel with two capacitors, respectively mimicking two channels that are coupled to each other with a resistor.

Additionally, an input resistor and an output impedance comprising a resistor and a capacitor have been proposed [281]. The LIF neuron has been demonstrated using a diffusive/volatile memristor in parallel with a capacitor, and the neuron has been employed in a fully memristive neural network [367].

Despite the existence of memristor-based neuron models, the lack of accessible experimental data and hardware for prototyping memristive circuits represents a significant obstacle. A significant number of state-of-the-art experimental demonstrations have relied on bespoke fabrication processes that cannot be reproduced using off-the-shelf memristors.

Furthermore, the limited choice of commercially available, low-cost, discretely packaged memristors, which are known to be highly sensitive and stochastic in behaviour, has compounded the issue. The challenge in developing hardware prototypes has made it difficult to perform experimental validation.

In addition to the acceleration of neural networks, the design of a solid-state brain that can harness the neural code has gained increasing attention as a means of processing vast quantities of sensory data without being constrained by the von Neumann bottleneck. The solid-state brain is structured in a manner that emulates the cerebral cortex, with neurons interconnected by a vast array of variable synapses.

It is hypothesised that neurons and synapses can be realised with memristors integrated with a complementary metal-oxide semiconductor (CMOS) process. However, the design of a large-scale solid-state brain remains elusive in neuromorphic computing due to the considerable overheads associated with mimicking the surface area of neural tissue, constrained power consumption, routing, and the massive parallelism of synaptic connections.

2.1.3 Synapse Models

Prior to model internal neural dynamics, it is essential to model the dynamics of the synapses that connect neurons to one another. Synapses exert a significant functional effect as a low-pass filter on the spikes that pass through them. A spike in the presynaptic neuron elicits an extended current pulse in the postsynaptic neuron. This pulse can be conceptualised as a low-pass filtered version of the presynaptic spike.

The simplest model of a synapse is that of a first-order low-pass filter. The impulse response of a filter describes the manner in which the filter responds to an infinitesimally short input of unit integral, which is called an impulse. This idealised impulse is also a reasonable model of a spike, and thus the impulse response also describes what the postsynaptic current will look like in response to a presynaptic spike. The impulse response of the first-order low-pass filter is as follows:

$$h(t) = \frac{1}{\tau_s} e^{\frac{t}{\tau_s}} \tag{2.18}$$

The synaptic time constant, denoted by τ_s, is defined as the length of time over which the postsynaptic current is spread. Given that the impulse response is an exponential function, the exponential synapse model is therefore a suitable description.

$$h(t) = \frac{1}{\tau_s^2} e^{\frac{t}{\tau_s}} \tag{2.19}$$

It was determined that a second-order lowpass filter is a superior model for a synapse [216]. The impulse response of this filter is defined by the equation 2.19. This function is referred

to as the alpha function, and thus the model is designated as the alpha synapse model. Both of these models are based on the current generated by a spike in the postsynaptic neuron, which is a current-based synapse model.

Other current-based synapse models exist; a popular one is the double-exponential model, which is similar to the alpha synapse but with two time constants, thereby affording greater control over the rise and fall of the impulse response. Where The double-exponential model with both time constants set to the same value reduces to the alpha synapse.

Many of the other, more realistic synapse models are conductance-based, meaning that they model the conductance of the neural membrane at the synapse. The current flowing into the neuron is dependent on both the conductance and the voltage across the membrane. The latter undergoes a change as the synapse becomes active.

One of the primary objectives of computational neuroscience is to ascertain the manner in which the brain represents—or encodes—information. To this end, researchers have put forth a multitude of potential coding schemes that neurons could utilise for information encoding. One key distinction between rate coding and temporal coding is the following dichotomy.

In a rate code, the sole pertinent measure is the firing rate (i.e. the number of spikes) of a neuron over a given period of time. An exemplar of a rate code is motor neurons in the peripheral nervous system. The contraction of a muscle is contingent upon the number of spikes per unit time; thus, only the rate of motor neuron spikes is significant [96]. In a temporal code, the time of individual spikes is also a factor. An exemplar of a timing code is the early auditory system, where precise spike timing facilitates the localisation of sounds [39].

The precise definitions of rate and temporal codes remain contentious, with differing interpretations presented by various authors [59]. To illustrate, a neuron may discharge a number of spikes in rapid succession, followed by a period of quiescence. A second neuron may be observed to fire the same number of spikes, but in a more evenly distributed manner over a given period.

One possible interpretation is that both neurons have the same firing rate, but the first has all its spikes occurring near the beginning of the period, in which case the timing of the spikes is a significant factor. An alternative perspective posits that the instantaneous firing rate of the

first neuron fluctuates over the period, whereas that of the second neuron remains constant. This suggests that it is the instantaneous firing rate, rather than the overall firing rate, that is of primary importance.

This highlights a limitation of solely examining the overall firing rate (i.e., the number of spikes) of a neuron over a given period. It is unclear which period should be considered. In real neurons, the period of time that is relevant for counting spikes depends on parameters such as the membrane time constant tau of the postsynaptic neuron.

For this reason, neuroscientists will often differentiate between rate and timing codes based on the frequency of alterations in the instantaneous firing rate. If the rate of firing exhibits rapid fluctuations, and if these fluctuations contain information about the stimulus (and are not simply spurious variation or "noise"), then the code is said to be temporal; otherwise, it is a rate code.

Once again, there is a lack of consensus regarding the minimum frequency of fluctuations in firing rates that must be observed to qualify as temporal codes. One possible definition is relative to the stimulus. The firing rate of neurons can be triggered to change rapidly in response to fast-changing stimuli, regardless of whether the code in use is rate or temporal.

If the temporal code is defined as having meaningful firing rate fluctuations at a faster time scale than changes in the stimulus, then it can be differentiated between codes that have fluctuations because they are using them for coding and those that simply have fluctuations triggered by the stimulus. According to this definition, neurons using a temporal code will have a fluctuating firing rate in response to a constant stimulus, whereas neurons using a rate code will have a non-fluctuating firing rate.

Both rate codes and temporal codes describe the encoding properties of individual neurons. Additionally, one may inquire about the coding properties of a group (also known as a population) of neurons. The concept of population coding pertains to instances where a representation is distributed across numerous neurons within a population, such that the represented value cannot be decoded from the activities of a limited number of neurons.

To illustrate, the simplest method of extrapolating the concept of rate or temporal coding to multiple neurons would be to have numerous neurons all implementing the same code. In other words, all neurons will exhibit a similar firing pattern when representing a given value,

due to their comparable tuning properties. This results in a significant degree of redundancy between neurons.

In contrast, population coding entails each neuron representing a distinct aspect of the represented value. To illustrate, if the objective is to represent head direction, there are neurons that represent a head that is fully turned to the left, others that represent a head that is fully turned to the right, and still others that represent a centred head. Additionally, there are neurons that represent values in between these three head directions. The direction in which a neuron is most active is referred to as its preferred direction.

It should be noted that each neuron exhibits some degree of variance, whereby it will fire for head directions that are relatively close to its preferred direction. Conversely, the further the actual head direction is from a neuron's preferred direction, the less it will fire. By utilising the activities of all neurons in the population, it is possible to decode the head direction with a high degree of accuracy.

From a population perspective, it is possible to differentiate between codes that exploit synchrony between neurons and those that do not. This can be facilitated, for example, by coincidence detection, whereby a postsynaptic neuron will only fire if the spikes of two of its input neurons are coincident, that is to say, they fall within the same (small) temporal window.

This distinction between temporal and rate coding schemes is further exemplified by the fact that only temporal codes can take advantage of correlations between individual spikes of neurons. Rate coding schemes, on the other hand, can only take advantage of synchrony between neurons in terms of synchronised fluctuations of their instantaneous firing rates, since they lack the temporal precision to co-ordinate individual spikes.

2.2 Methodology

2.2.1 Modified Device Stack

A distinct set of devices with disparate top electrical contacts were characterised, one with conductive indium tin oxide (ITO) in lieu of gold. The bottom contact and oxide layer remained unaltered and consistent with those observed in the gold-contacted devices presented in the preceding section. When subjected to stress, the ITO-contacted device exhibited a distinct response compared to the gold-titanium contacted device. Instead of a gradual and

smooth increase in conductance, the response was more erratic and chaotic.

Table 2.2 Comparison of the modified device stacks.

Original Stack		Modified Stack	
Layer	Film Thickness (nm)	Layer	Film Thickness (nm)
Au	110	ITO	30
Ti	3	Ti	3
SiOx	35	SiOx	20
Mo	280	Mo	150

The aluminium-contacted devices have yet to demonstrate the occurrence of current transients following the application of stress. In addition to failing to exhibit current transients, any increase in conductance induced by the constant current stress is also observed to be more volatile than that observed in the other devices, with the devices returning to a high resistance state within a couple of hours.

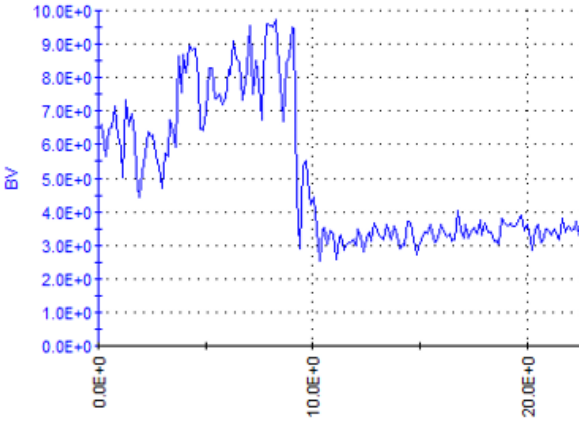

Fig. 2.5 Stressing responses of ITO top contacted device. With the objective to ascertain the stress responses of the ITO top-contacted device, the voltage across the device was monitored while it was subjected to a constant current of $-0.5\mu A$. It was not possible to apply larger currents. The response was observed to be less smooth when compared to that of the gold-contacted device.

In contrast, the ITO contacted devices did exhibit transients, but interestingly, only partially. A typical transient observed in ITO devices is plotted in Figure 2.5. It exhibits the initial increase in conductance as is typical with current transients, but does not then start reducing in conductance. Instead, the device exhibits a chaotic spiking-like behaviour which, if observed

for too long, will cause the device to switch to a low resistance state.

The observation of a partial current transient in ITO-contacted devices is a significant finding. As will be discussed in the following section, this evidence is indicative of the transient being the result of multiple simultaneous changes occurring in the device. Furthermore, it supports the hypothesis that the top metal-insulator interface plays a role in generating transients.

The hypothesis that alterations to specific interfaces of the device can influence the characteristics of the current transient is supported by findings in tantalum oxide-based devices [344]. In this study, a layer of Al_2O_3 was deposited between the tantalum oxide bulk and the titanium nitride electrodes, which reduced the prominence of the current transient in the absence of the buffer layer.

To illustrate, the gradual decline in conductance of the transients is exclusive to the gold-contacted device, indicating that it is either due to the characteristics of the metal-insulator interface or the disparate responses to stressing at this interface that determine whether the decaying behaviour is manifested.

In contrast, the initial increase in conduction is observed in both the ITO and gold-contacted devices. This suggests that the behaviour is less affected by the top metal-insulator interface and may be located in the bulk oxide layer or at the bottom metal-insulator interface. The disappearance of the slower decay in conductance with the change in top electrode may provide insight into the physical model describing the current change.

2.2.2 X-ray Photoelectron Spectroscopy

X-ray photoelectron spectroscopy (XPS), also known as electron spectroscopy for chemical analysis (ESCA), is a widely utilised surface analysis technique in materials science [81]. It enables the acquisition of information regarding the composition and electronic structure of matter [270]. Furthermore, additional chemical information, such as binding energy and oxidation states, can be obtained through the utilisation of this technique.

Photoelectron spectroscopy is founded upon the photoelectric effect, which posits that a flux of photons can eject material electrons [227]. The effect was first observed by H. Hertz in 1887 [118] and subsequently elucidated by Einstein in 1905 [68]. The advent of photoelectron spectroscopy can be traced back to the late 1950s. Utilising X-ray radiation for excitation, K. Siegebahn conducted a study on the energy levels of core electrons in atoms

[266].

In XPS, samples are irradiated by X-rays, resulting in the ejection of electrons. These electrons are then collected in order to infer information about their composition and electronic structure. As the exact binding energy position of the core level depends on the chemical environment of the atom from which the electrons were ejected, Siegebahn designated this technique as electron spectroscopy for chemical analysis (ESCA).

The identification of elements and information regarding chemical bonding are inferred from electron energy and energy shift, respectively. The analysis process, which is conducted in an ultra-high vacuum, requires high energy and short-wavelength radiation. Furthermore, depth profiling is achievable through the use of ion sputtering [270].

X-ray photoelectron spectroscopy (XPS) is a technique that employs energetic X-ray radiation to infer information about the surfaces of materials in a vacuum. By utilising only the elastic electrons emitted from the tested sample, information regarding the sample composition can be retrieved from its spectrum.

To optimise the mean free path of the photoelectron and minimise sample contamination, measurements are conducted in an ultra-high vacuum. It is important to note that in multi-component samples, such as a stack of thin-film structures, spectral overlap from multiple elements may occur.

Figure 2.6 depicts a schematic representation of an XPS system. An X-ray source is employed to irradiate the sample, which is situated within a vacuum chamber. A vacuum pump is connected to the system, enabling the reduction of the pressure to a range of 10^{-8} to $10^{-9} mbar$ for the purposes of analysis. The ejected photoelectrons are detected by means of electronic lenses. It is possible to adjust the voltages during this process in order to select electrons of a specific kinetic energy.

A hemispherical analyser (HSA), which comprises a pair of charged plates, is employed to select electrons of specific energy (also referred to as 'pass energy'). The electron detector quantifies the flux of incoming photoelectrons with the specified pass energy, a process also designated as 'fixed analyser transmission' (FAT) mode. By varying the lens voltage across a range of electron energies to ascertain the variation in flux at the detector, the spectrum of

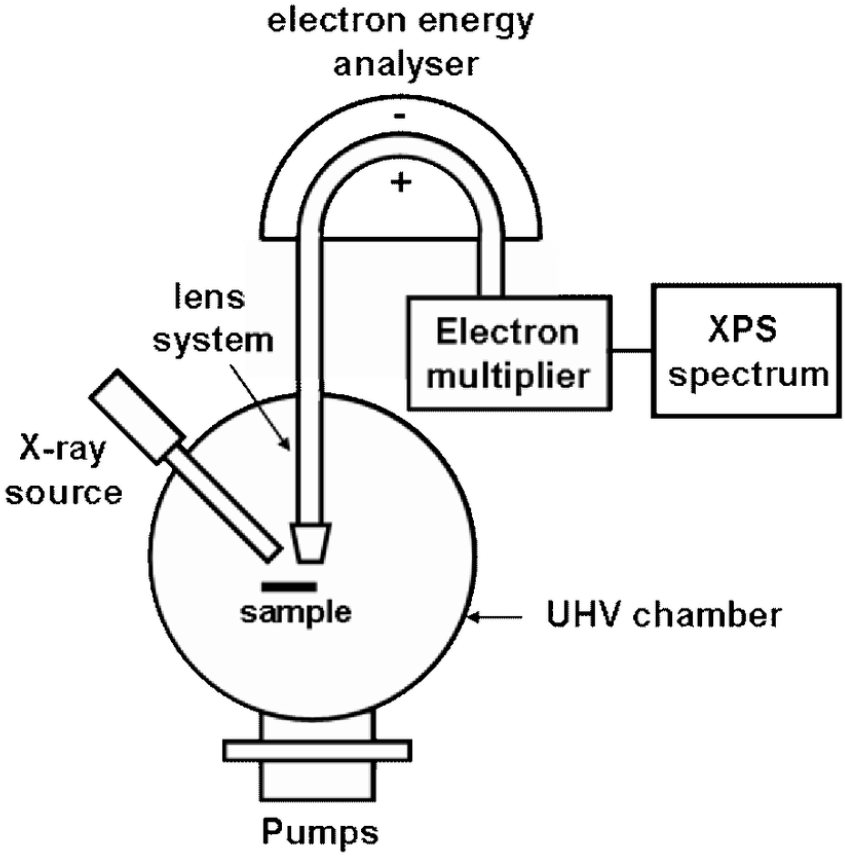

Fig. 2.6 Schematic of an XPS instrument. The X-ray radiation interacts with the sample and emitted electrons are focused onto the analyser which measures their kinetic energies and indirectly their binding energies [293].

the measured sample can be generated and displayed.

In order for the incident radiation to ionise matter and extract its electrons, the energy of the radiation must be sufficient to eject electrons into the vacuum. The energy required for this process is known as the binding energy E_b for core electrons. The electron energy E_k can be expressed in terms of the incident x-ray energy $h\upsilon$, where h is Planck's constant and υ is the photon frequency, the sample work function ϕ, and the electron binding energy , as follows:

$$E_k = h\upsilon - E_b - \phi \tag{2.20}$$

In a given spectroscopic measurement, the frequency υ is known and the constant of Planck's law of radiation, h, is constant. Consequently, the energy of electrons in the conduction band, E_b, can be measured to estimate the energy of electrons in the valence band, E_k. The latter varies depending on the element, and therefore different peaks characterising the composition of the sample will appear in the XPS spectrum.

X-ray radiation has the capacity to penetrate deeply into a sample, yet only electrons situated in close proximity to the surface are able to escape. As a consequence of the high degree of scattering that occurs within the sample, the photoelectrons generated are subject to a significant loss of energy. The surface sensitivity of the technique is determined by the low escape depth (0.5-2 nm) of the elastically scattered electrons [270].

The energy resolution of the measured spectrum is susceptible to a number of parameters. These parameters include the diameter of the analyser, the energy spread of the X-ray source, and the pass energy. Conversely, the ionisation process results in the formation of localised charges at the surface of the sample (sample charging).

The presence of these localised charges may result in a broadening of the spectrum lines. It is therefore necessary to implement an effective charge neutralisation process. This is achieved by injecting electrons into the vacuum chamber using an ion gun, which neutralises the tested sample and improves its photoelectron emission.

It should be noted that the pass energy of the detector determines the energy of the electrons that are detected, and thus the detector energy resolution. This quantifies the ability to resolve peaks that differ in energy by a small amount. The lower the energy pass, the greater the resolution power of the analyser.

In contrast, Auger peaks manifest at elevated binding energies. Radiation-induced photoemission gives rise to shell vacancies, which can be filled by excited electrons undergoing decay and emitting Auger electrons. X-ray emission can also occur during the ionisation process in heavy atoms. Many samples exhibit degradation under exposure to X-rays.

The probing method frequently permits focusing the X-ray beam onto a narrow, specific spot, thereby reducing sample damage to an absolute minimum. Despite XPS being a surface analysis technique, depth profile measurement can be performed nonetheless [94]. This can be typically achieved by integrating an ion beam source into the XPS system to mill the sample.

Through ion milling, the sample material can be removed in a sequential manner while its composition is characterised. However, the milling process may result in alterations to the composition and cause damage to the sample. The introduction of energetic ions during the milling process may also result in the implementation of material, leading to changes in the composition with depth.

2.2.3 Model Fitting Framework

This section presents the development of empirical models designed to track the rates of increase and decrease in conductance separately. The models are primarily intended to quantify the rate of change in conductance. The question of a physical model will be addressed in the following section.

The models explored here are derived from the relaxation experiments that were previously outlined in the preceding chapter. It was evident that the gradual decline in device current delineated the upper limit of the maximum device current, while the initial surge in current approached this maximum but did not exceed it.

$$I(t) = f_{inc}(t) \times f_{dec}(t) \quad \forall \{f_{inc}(t) \in [0 \rightarrow 1] : f_{dec}(t) \in \mathbb{R}\} \tag{2.21}$$

This is represented by the product of two time-dependent functions, $f_{inc}(t)$ and $f_{dec}(t)$. The increase in current is analogous to a charging term, $f_{inc}(t)$, which rises from 0 to 1. Initially, this function defines the device current. However, as $f_{inc}(t)$ approaches 1, it then allows the function it is multiplied with to define the total current, in this case $f_{dec}(t)$.

Although this equation forms the basis of the empirical model, questions remain regarding the specific forms that $f_{inc}(t)$ and $f_{dec}(t)$ should take and the most appropriate means of comparing their effectiveness. The efficacy of each fitting equation is evaluated based on two criteria: the quality of the fit and the degree of realism of the fitted parameters in relation to the underlying physical system from which the model is derived.

The correspondence between each term of a given model and a physical property is contingent upon the physical system from which the model is derived. These properties are assigned a range of values that are deemed realistic. Values outside of this range may indicate that the assumed model is not applicable. The specific correspondence between physical properties and terms is model-specific and will be detailed later in conjunction with the model.

The discrepancy between the fitted equation and the original experimental data is referred to as the residual. This can often be a useful visual indicator of the quality of the fit. An optimal fit would manifest residuals that are centered around zero, exhibiting no systematic offsets or time-variant components.

The residuals in this form indicate that the fitted equation tracks the experimental data well, with the variances in the residuals around zero assumed to be a form of noise or variance in the original data. In contrast, a less optimal fit would exhibit systematic offsets that vary over time. This indicates that either an additional term is absent or the incorrect function has been selected.

However, while residuals are useful for visually assessing a fit, they are less so when larger datasets are being fitted. Therefore, only one residual for each model is assessed, which is representative of the model's performance. The same experimental data will be fitted for each model, thus ensuring an accurate comparison. In order to assess a model's goodness of fit across a whole dataset, a single numerical metric that can quantify the fit is preferred.

A more quantitative description of the goodness of fit is provided by the R^2 measure. The R^2 measure is a statistical tool that enables the comparison of the variance between the observed data points and the model's predicted values against the variance of the observed data and the mean of that data. In other words, it can be acknowledged that the most straightforward

model for predicting a dataset would be to assume the mean value of the dataset in all cases.

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \tag{2.22}$$

$$SS_{res} = \sum_i (y_i - f_i)^2 \tag{2.23}$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{2.24}$$

In this case, the variance is defined by (2.22), Where y_i is an individual datapoint and $\bar{y}$ is the mean average of the dataset, which is equal to the variance of the dataset. The variance of the dataset against the model's predictions can also be calculated with (2.23), Where f_i is model's predicted value at i^th index, for any other model developed.

If the variance is similar to that of the dataset, the model is no more than a simple mean value predictor, and thus the data fit is poor. An alternative indication of a good fit is provided by a model with a variance much less than that of the dataset. However, residuals should still be checked. The R^2 value shown in (2.24), defined by the ratio of the variance of the model to the variance of the dataset, is based on this premise and increases as the model's fit improves.

Notwithstanding the possibility of attaining an optimal fit, the values of the fitted parameters remain uncertain. To illustrate, a minimum in the fitting error could be achieved by a range of parameter values. This range is referred to as the confidence bounds, which can be interpreted as the range within which the fitting algorithm is certain the final value lies.

For example, 90% confidence bounds will define a range within which the algorithm is 90% sure the optimal value can be found. If a higher confidence is required, the range will generally increase. Consequently, there is a trade-off between certainty and specificity. In this work, the standard confidence threshold of 90% was employed.

The choice of model for a particular dataset is influenced by the magnitude of the confidence intervals. If a model results in fitted parameters with large confidence intervals, it can present a challenge when interpreting the results, particularly if the changes in these values are small. Consequently, when selecting a model for the analysis of a dataset, preference will be given to models with smaller intervals.

2.2.4 Empirical Model Definition

The current transients observed in amorphous silicon oxide devices appear to result from two distinct changes occurring within the device simultaneously, leading to both an increase and a decrease in conductance. The two changes in conductance exhibit a number of distinguishing characteristics.

Firstly, the increase in conductance occurs significantly faster than the subsequent decay. Secondly, in terms of volatility, the increase in conductance relaxes to its initial state within tens of milliseconds, whereas the decay in conductance can take hours to fully reset. Thirdly, in terms of their material dependence, the decay in conductance can be removed by changing the material of the electrodes [222].

This has led to the conclusion that the two changes are driven by different mechanisms and can exist in isolation, which will be detailed in the following sections. The SPICE models for each of these processes will first be presented separately, and then the two will be combined to obtain the final model.

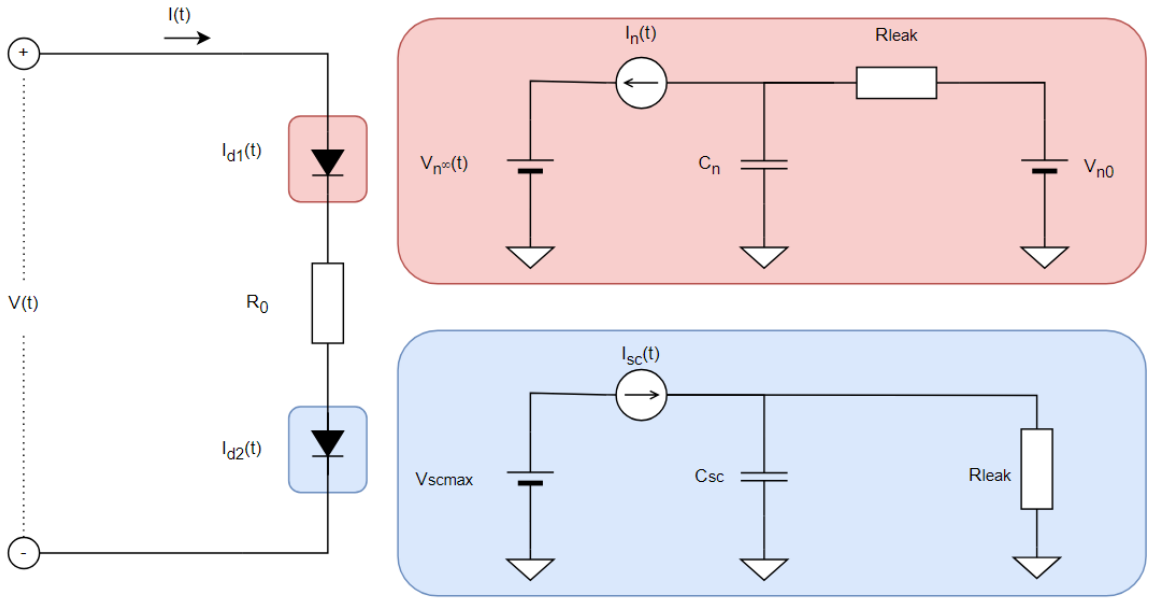

Fig. 2.7 SPICE Model diagram. The SPICE model generates an output current, *I(t)*, based on an input voltage, *V(t)*. The circuit incorporates a single resistor, R_0, along with two diode models that serve to attenuate or amplify the voltage applied to the resistor. These diode models are represented by the sub-circuits highlighted in blue and red.

The accelerated rise in conduction appears to be attributable to the phenomenon of charge trapping. This is corroborated by its shorter timescales, greater volatility, and experiments in

which optically injected carriers were demonstrated to accelerate the process. One potential mechanism by which charge trapping affects device conductance is through modulation of the height of a Schottky-like barrier [55], which may be influenced by the population of interface states [332].

Although Schottky barriers are typically formed between metal-semiconductor interfaces, studies have identified analogous barriers within memristor devices [107]. The silicon oxide devices exhibit rectifying behaviour, indicating the presence of an interface barrier at one of the metal-insulator interfaces. Additionally, the filaments in the devices are composed of silicon-rich regions within the oxide, suggesting that the interface between these filaments and metal contacts may resemble a Schottky interface.

The Schottky barrier is modelled as a voltage drop, which acts to reduce the voltage across the active layer of the device, as illustrated in 2.7. As the height of the Schottky barrier diminishes, the potential across the active layer increases, resulting in a greater device current. The voltage drop resulting from the Schottky-like barrier is a function of time and is modelled using the sub-circuit illustrated in red in 2.7.

The magnitude of the voltage drop is represented within the sub-circuit by the voltage across the capacitor, C_n. The initial value of the aforementioned variable is discharged by the current source, V_{n0}. The rate of discharge is defined by the current source, $I_n(t)$, whose magnitude is proportional to the difference between the current voltage drop across the Schottky barrier and its final equilibrium value, $V_{n\infty}$.

$$I_n(t) = \alpha \times [V_n(t) - V_{n\infty}] \tag{2.25}$$

In this equation, α corresponds to the probability of trapping a carrier, while the voltage difference relates to the concentration of unpopulated states. The current source discharges the capacitor to a final equilibrium voltage, $V_{n\infty}$, where the trapping and de-trapping currents are equal. An additional leaking resistor, R_{leak}, is introduced to correctly adjust for the rectifying behaviour of the device.

There is compelling evidence that the observed decline in conductance can be attributed to the movement of charged ions within the oxide thin film. This hypothesis has been put forth in the majority of publications on such current transients [361]. This is largely based on the observation that the decay in conductance occurs over a time scale of tens to

hundreds of seconds, which is too long to be associated with the trapping of electrons or holes.

Furthermore, in accordance with the drifting defect hypothesis, it has been demonstrated that the process can be reversed by applying a voltage of opposite polarity, despite the device exhibiting significantly reduced currents in the opposite polarity. The model adhere to this approach and hypothesise the presence of migrating defects. Of particular significance is the assumption that the ionic current induced by the migrating space charge is negligible in comparison to the electronic currents, as predicted [248].

It is assumed that the electronic current flows through a conductive channel within the oxide. In our silicon oxide devices, this is a silicon-rich filament that forms during electrical stressing. Such filaments are common in memristors and have been observed in our devices using etching C-AFM techniques [29].

In order for an electronic current to be induced through these filaments, it is necessary that a potential be present across the channel. In our model, the migrating space charge modulates the current by reducing the potential experienced along the filament. It is probable that this space charge is a positively charged ion that has accumulated in proximity to the upper gold electrical contact. This assertion is corroborated by the observation that the impact of the accumulation of space charge can be negated by modifying the material of the top electrical contact from gold to indium tin oxide [222].

The precise nature of this ion remains uncertain. A substantial amount of oxygen migration and associated oxygen vacancies have been observed throughout the device when under bias [353], indicating that this could be a viable candidate. However, in other devices, hydrogen has been identified as a defining factor in device behaviour. Additionally, hydrogen has been detected in significant quantities within our devices [178]. Given the lack of certainty regarding the identity of the ion, we assume a generic space charge.

The effect of the space charge on the conductive channel can be described as follows. In the initial phase, the space charge exhibits a homogeneous distribution. Upon the application of a voltage to the device, a force is imparted on the space charge, resulting in its drift and accumulation at the device electrode. This accumulation effectively blocks the space charge from exiting the oxide.

This accumulation results in the formation of a region of higher space charge concentration, which consumes a portion of the potential applied across the device. This results in a reduction in the potential drop across the conductive channel. As the space charge accumulates, this voltage drop increases, meaning less potential is dropped across the channel and a reduction in device current is observed.

Eventually, the drifting force imparted on the space charge will reach equilibrium with the diffusion and Coulombic repulsion formed by the accumulated space charge, leading to a steady state condition. When the potential is removed, the space charge diffuses back to its original distribution.

The aforementioned process is modelled using the circuit illustrated in Figure 2.7. The conductive channel is represented by a fixed resistance, designated as R_0. The voltage across the conductive channel is defined by the applied potential at the terminals of the device, $V(t)$, and the voltage source, $V_{sc}(t)$, which represents the voltage drop caused by the accumulated space charge. This voltage source is time-dependent and is defined by the subcircuit shown in blue. As the voltage of this source increases over time, the voltage across the fixed resistor drops and the device current also reduces.

$$I_{sc}(t) = [V_{scmax} - V_{sc}(t)] \times [\mu\,(V(t) - V_{sc}(t))] \tag{2.26}$$

The blue sub-circuit illustrated in 2.7 is responsible for monitoring the accumulation of the space charge and its corresponding voltage drop, which is represented by the voltage across the capacitor C_{sc}. The capacitor is charged by the current source, $I_{sc}(t)$, which generates a current in (2.26). The capacitor is defined by the voltage applied across the device and μ, which symbolises the mobility of the space charge. The current source draws charge from a voltage source representing the steady-state voltage drop, i.e. the maximum voltage drop consumed by the accumulated space charge V_{scmax}.

$$I_d(t) = (I_0 + I_\delta V_n(t)) \times \left(e^{\frac{V(d_1,d_2)}{nV_t}} - 1 \right) \tag{2.27}$$

To complete the model, the subcircuits for both potential drops are combined as illustrated (2.27). The two voltage drops act upon a single resistor representing the conductive channel, R_0. In practice, this channel does not exhibit ohmic conduction and thus its resistance will have a voltage dependence. This is taken into account while collecting the meta-parameters.

2.3 Results

2.3.1 Conductance Variation Mechanisms

The initial step is to ascertain the location within the device stack where alterations are taking place that are responsible for the observed reduction in conductance. The absence of decay occurring concurrently with the alteration of the top electrode suggests that the causal factor responsible for the observed conductance decay is situated at the interface between the top electrode and the amorphous silicon dioxide.

Given the slow dynamics of the change in conductance, it is plausible that a drift of some mobile defect is responsible. It is well established that silicon dioxide films are susceptible to the influence of alkali mobile ions [318], including sodium and lithium ions, which are all characterised by a positive charge [398]. The drift of these mobile charges can significantly affect the potential drops at metal-oxide interfaces, as well as modulate barrier heights when allowed to accumulate.

If some positive mobile ion, regardless of the species, existed in the oxide of the device, it would be attracted to the top electrode, which is at a negative potential. This would cause an accumulation of positive space charge at the interface, which would in turn reduce the potential across the oxide. Nevertheless, it can be argued that alkali metals, such as sodium and potassium, are unlikely to be the cause of this positive space charge, given that they do not migrate at room temperature.

Instead, they require temperatures in excess of 100 degrees Celsius (212 degrees Fahrenheit) [60]. The current transients presented in this thesis are all observed at room temperature, which suggests the need for an alternative candidate to explain the mobile space charge, in particular one that is mobile at room temperature.

It is noteworthy that modelling of the temperature within analogous TaO_x devices has indicated the potential for increases in oxide temperature of up to 100°C with applied voltages of -0.7 to -1.8V due to Joule heating [313]. This would imply that if comparable effects were present during the current transient, then elevated temperatures within the oxide could be occurring and potentially facilitating the migration of alkali metal defects.

It seems plausible to suggest that the proton [123] is a likely candidate for positive ions that are mobile at room temperature. The presence of ionised hydrogen in silicon dioxide films

has been repeatedly observed to be both stable and consistent [352]. It has been demonstrated that protons can influence the electronic properties of capacitor devices in which protons are trapped within the oxide [351]. Their long-term stability has been demonstrated through multiple cycles of migration between device electrodes [369].

It is commonly assumed that these ions are introduced during the growth of the oxide [350]. Furthermore, their concentration has been demonstrated to increase through annealing in an atmosphere at temperatures above 200 degrees Celsius [203]. However, their presence has also been introduced electronically via the electrolysis of water within the device and via radiation [379]. It is also noteworthy that their migration has been shown to occur repeatedly at room temperature.

This raises the question of why the accumulation of protons occurs exclusively in the gold-contacted devices, rather than in the ITO. Given that gold is an inert metal and is unlikely to be reduced by protons, the accumulation at the gold interface is to be expected. In contrast, there is a substantial body of evidence indicating that the ITO would be reduced in the presence of protons.

Although ITO contacts are often considered to be inert in certain electrochemistry scenarios, this is not always the case. Their reduction is, in fact, heavily dependent on the pH of the electrolyte. For instance, the reduction of the electrode has been observed on numerous occasions in acidic electrolytes [53, 307].

The reduction of ITO in the presence of acids has been demonstrated in both electrochemical experiments conducted at room temperature [365] and in instances where ITO has been exposed to a hydrogen plasma [15]. In one study, the application of negative voltages to an ITO electrode immersed in hydrochloric acid resulted in the formation of spherical structures at the grain boundaries of the ITO film, which exhibited a metallic-like appearance [88].

Following characterisation with Energy dispersive x-ray Spectroscopy (EDS) [129] and X-ray Diffraction (XRD) in a separate study [210], the spherical regions were found to be depleted of oxygen or exhibited only peaks of indium and tin, providing compelling evidence that these spheres were metallic. The same spherical structures were observed in ITO films exposed to a hydrogen plasma, which, when analysed with Auger spectroscopy, again revealed a lower oxygen concentration in the spherical regions.

The reduction of ITO by protons may provide an explanation for the absence of space charge accumulation in ITO-contacted devices. Instead of accumulating, the protons reduce the ITO, producing water as a byproduct, which would not contribute to a positive space charge. Furthermore, the reduction of the electrode may also elucidate the more erratic current-time response observed in ITO devices, as the electrode structure undergoes substantial alterations.

The potential for structural changes to occur in both the oxide and the metal contacts makes it challenging to determine the specific role each plays in modulating the device's conductance. In order to investigate the effect of the contact, it would be beneficial to fabricate and characterise devices with a variety of contact materials.

Any discrepancies in the observed behaviour between the devices could be ascribed to the metal or the metal-insulator interface, whereas any enduring effects could be attributed to the oxide. To further examine any behaviour attributed to the oxide, devices of varying oxide thicknesses could be fabricated. This may reveal a dependence on the oxide thickness, which could be supporting evidence for the oxide having a role in the changing device conductance.

However, it is important to exercise caution when drawing conclusions from this approach, as the change in oxide thickness will also modify the magnitude of the current density flowing through the device, potentially affecting the interfaces. The fabrication of devices with different oxide thicknesses and a variety of metal contacts is a future research direction.

If the hypothesis that the transient is the result of two separate changes is true, then it would suggest that devices could potentially be made to exhibit only one of these changes in isolation. Fabrication of such a device would provide strong supporting evidence for the hypothesis and a clear demonstration that the changes are separable. Modification of the top contact material appears to facilitate this separation.

Devices fabricated with a conductive ITO top electrode, in lieu of the gold-titanium contact, do not exhibit the anticipated decay in conductance; rather, they display only the initial increase. Figure 2.8 illustrates the current-time response of an ITO-contacted device. As observed previously in the gold devices, the current begins to increase; however, it never reaches the inflection point. Instead, it continues to increase in conductance, becoming progressively noisier until the device undergoes breakdown.

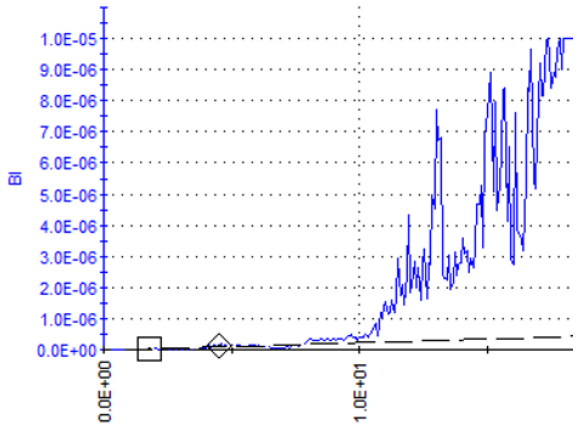

Fig. 2.8 The current-time response for a device with a conductive ITO top electrode. The current is generated in response to a step potential applied to the top contact with respect to the bottom. It exhibits only the increase in conductance and not the decrease. As the current increases, the noise level also rises until the device reaches a point of breakdown. The structure of the device is identical to that of the gold-contacted devices, with the exception of the change in the top electrode's material from gold-titanium to indium tin oxide (ITO).

The ITO device has undergone a comparable stressing process to that of the gold devices, which is outlined in the preceding chapter. In the initial stages, the devices exhibit only capacitive currents and possess a very high resistance. Subsequently, a constant current stressing procedure is employed to produce a more conductive device.

Following a period of relaxation, a repeatable transient is produced, provided that the applied voltage is maintained for a sufficiently brief duration to prevent breakdown of the device. A comparable phenomenon has been documented [254] in a variety of oxides and is frequently employed in the replication of short-term potentiation of synapses [408, 38].

Furthermore, this absence of slower dynamics may corroborate with previous findings [248], where the slower dynamics were postulated to be attributable to oxygen vacancies. It is conceivable that the ITO contact is more prone to exchange oxygen with the oxygen vacancy than the inert gold contact.

An alternative hypothesis is that the Au contact is diffusing through the oxide, whereas the ITO contact is not. Previous observations have shown that Au can form conductive bridges between two contacts through a thin film of *ZnO* [276]. Given that gold is known to diffuse in silicon dioxide films, this could be a possibility [215]. However, TEM analyses of the devices studied in this thesis have not produced observable gold filaments, casting doubt on

this hypothesis [242].

An additional potential explanation for the observed discrepancy in behaviour between the Au and ITO contacts is the possibility of differences in their respective work functions. While not directly measured on the samples in question, the work function of gold is reported to be between 4.9 and 5.2 electronvolts (eV) [342], while thin films of indium tin oxide (ITO) have been measured to have a work function between 4.25 and 4.28 eV [303].

This could result in a difference in work function of approximately 1 eV. Such differences would lead to offsets in the band alignments at the contact and oxide, which could affect which traps within the oxide the electrons are injected into. As discussed in the previous chapter, the disappearance of the slower decay in conductance with the change in top electrode is suggestive of proton migration playing a role in the slower decay in device current.

2.3.2 Oxygen Exchange Experimentation

2.3.3 Model Fitting and Evaluations

2.4 Conclusion

Neuromorphic modelling is therefore concerned with the creation of artificial systems that emulate the functionality of biological neural systems, particularly in terms of their physical implementation. The term was first used in the late 1980s to describe digital and analogue hardware that is organised in a more brain-like manner than traditional computer hardware [237].

One of the fundamental concepts underlying neuromorphic systems is parallel distributed processing. Neuromorphic systems arrange computations at the neural level, with a specific focus on facilitating rapid communication between neural processing units. This contrasts with other parallel distributed systems, such as graphics processing units (GPUs), which are typically optimized for independent parallel computations and exhibit limited communication between units.

Chapter 3

Memristive Devices

3.1 Introduction

The initial primary focus of this thesis is on the characterisation of silicon-based memristors, which are the fundamental components of memristive systems. Since the discovery of the memristor and its importance to replicating synaptic activity had such a profound impact on the field of neuromorphic engineering, investigating additional nanoelectronic components and behaviours in this context will lead to new neuromorphic computing applications.

This chapter investigates a phenomenon known as the "current transient" that has yet to be deliberately applied to the demand of neuromorphic computing. The current transient phenomenon can be similarly represented by the current flowing through a defective capacitor in response to a step potential to produce rich dynamics, both growing and decreasing in conductance, and can be beneficial in a computational device.

This chapter begins by documenting and characterising the current transients based on available literature. The experimental procedures used throughout the chapter were then described, and strategies were developed to aid in the characterisation of current transients. This provides a deeper understanding of the physical models underpinning the transients, allowing for the further development of an integrative memristive system based on silicon oxide samples that are already available.

3.1.1 Foundational Properties

Fundamentally, the processes of capacitive decay and dielectric relaxation are ideal to define a capacitor's response to a step voltage. Applying a constant voltage across its terminals causes

the device current to decline until it ultimately comes to rest at a constant leakage current. This, however, is not always the case. When the voltage is applied for an extended period of time or at a high enough temperature, the current flowing through the device begins to grow as the oxide gets faulty and its resistance falls. This is known as oxide deterioration, an umbrella word for an oxide coating that becomes faulty over time as a result of environmental stress factors [98].

One specific form of this resistance degradation addressed here is the "current transient". This is distinguished by their characteristic form when plotted on a current-time graph as illustrated in Figure 2.1. When a voltage is applied to the device, the current grows swiftly to a maximum and then gradually decays, resulting in a distinctive peak in device current. Surprisingly, whereas oxide degradation is essentially a permanent impact, the change in oxide resistance that happens during a current transient is not; rather, it is volatile.

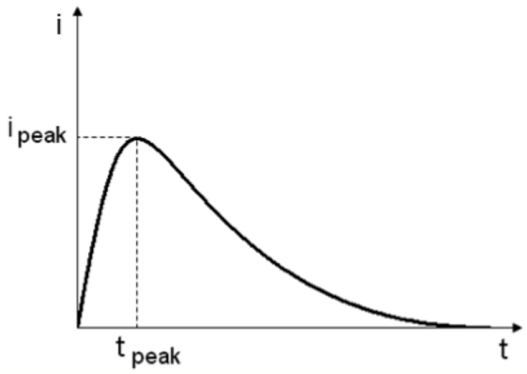

Fig. 3.1 Current transient caused by a change in temperature for a closed circuit [403], the plot is for illustration purpose only.

The most common differentiating feature is the combination of their distinct peak and their extended time periods during which they occur. Previously published papers documented peak periods ranging from 5 to 20 seconds, and the full transient is frequently seen for 10s to 100s of seconds [298]. They are both slow pace and long-lasting occurrences. Timescales of this magnitude are not usual for integrated resistor-capacitor (RC) circuits because they demand enormous capacitance, which necessitates physically much bigger capacitors [69].

One appealing aspect of the phenomena is the much longer time periods of the current transient's responses in contrasted with a traditional RC circuit. However, the dynamics of the transient are not fixed. They are known to be accelerated by increases in applied temperature [220], voltage [404], and when irradiated with a laser [200]. All of these variables can cause

the transient to accelerate, with the peak coming sooner in time. When pushed far enough, the peak disappears completely and we only see a decreasing current, which is most likely owing to the instruments' temporal resolution.

Given that the features of transients have been similar across published research, it is worth noting that these studies have used a variety of oxide materials. However, despite being present in a variety of device setup, the current transient is not equally observed in every instance, and this is especially noticeable with regard to device polarity. Early investigations on current transients only detected transients in a single voltage polarity [219], however other devices in the literature now produce transients in both polarities [41].

Transients have been reported in a wide range of materials, implying that they are a generic phenomena that might occur in a wide range of MIM devices under the correct conditions. The current transient's characteristics deviate from the typical resistance switching behaviours reported in silicon oxide devices. They are analogous and volatile as compared to devices that display non-volatile binary switching when run in a normal resistance switching mode. Understanding the source and features of current transients is significant because they may have ramifications for a variety of research areas studying metal-oxide structures, such as transistor gates or the comparatively emerging subject of resistive switching memory.

3.1.2 Current Models

The most well-known model for explaining this behaviour is the Space Charge Limited Current (SCLC) hypothesis, which was created for describing a general space charge within a perfect insulator [224]. A variety of theories have been offered up to describe this phenomenon [184]. Since charged oxygen vacancies are thought to constitute the space charge in question, this initial model was first associated with them.

Along with the electronic current, an ionic current also flows as a result of the drift of charged defects under an applied bias. The electrical contact at the other end blocks the charged defects, causing them to build up. The Coulombic repulsion that results from this buildup prevents further migration, which in turn reduces the ionic current as it starts to oppose itself. Because of this, earlier research assumed that the transient behaviours was caused by a general space charge inside of an insulator without any trapping in order to develop the SCLC model.

The original study only considered in one dimension and made the assumption that the conduction is planar. This might represent an ionic current or an electronic current, depending on whether the charges and mobilities correspond to electrons or some other charged defect inside the oxide. Recent study has assumed that the latter is the case. The mobility is frequently thought to refer to moving oxygen vacancies inside the oxide [404], or in other instances, to moving ions [41]; in all cases, it is supposed that the current is ionic.

Originally, the current of the model is defined as:

$$j(t) = j_{cond} + j_{disp} \tag{3.1}$$

$$j_{cond}(x,t) = qn(x,t)\mu E(x,t) \tag{3.2}$$

$$j_{disp}(x,t) = \varepsilon \frac{\partial E(x,t)}{\partial t} \tag{3.3}$$

$$qn(x,t) = \varepsilon \frac{\partial E(x,t)}{\partial x} \tag{3.4}$$

where $j(t)$ is the total current, j_{cond} is the electronic conduction current, j_{disp} is the displacement current, q is the the charge of a single electron or the respective ion measured in Coulombs, n is the space charge concentration, μ is the oxide space charge mobility, E is the oxide local electric field, L is the oxide thickness, ε is the oxide dielectric permittivity, x is the position within the oxide ranging from 0 to L, j is either the electronic or ionic current depending on the space charge, and lastly the Poisson's equation is denoted with equation 3.4.

$$j(t) = \varepsilon \left(\frac{\mu}{2} \frac{\partial E^2(x,t)}{\partial x} + \frac{\partial E(x,t)}{\partial x} \right) \tag{3.5}$$

$$j(t) = \frac{\varepsilon \mu}{2L} \left(E_a^2(t) - E_c^2(t) \right) \tag{3.6}$$

$$j(t) = \frac{\mu Q(t)}{2L} \left(2E_a(t) - \frac{Q(t)}{\varepsilon} \right) \tag{3.7}$$

By substituting into the first equation, the device current density description can be derived to be (3.5). Assuming the voltage applied is fixed, the boundary conditions for the electric fields can be defined as $E_a(t) = E(L,t)$ and $E_c(t) = E(0,t)$ for the anode and the cathode respectively. Integrating (3.5) with respect to x from the cathode to the anode yield equation 3.6 which describes the current as a function of the electric fields across the electrodes.

Note that the applied voltage was assumed to be non-varying with time, $\int_0^L \frac{\partial E(x,t)}{\partial t} dx = \frac{\partial}{\partial t} \int_0^L E dx = \frac{\partial V}{\partial t} = 0$, in a single dimension while the problem is intractable for spherical

and cylindrical geometries [184]. This equation can further be rewritten with respect to the insulator total charge (3.7), with the relationship between the cathode and anode electric fields given as $E_a(t) = E_c(t) + \frac{q(t)}{\varepsilon}$, where $Q(t)$ is the insulator total charge.

$$\frac{E}{E_a^2} = \frac{\mu}{2L}\partial t \tag{3.8}$$

$$E_a(t) = \frac{V}{L}\left(1 - \frac{\mu V}{2L^2}\right)^{-1} \tag{3.9}$$

After deriving an equation describing the device current, the subsequent step is to obtain the equation of the SCLC model that connects the mobility of the space charge with the time at which the peak τ appears. It is assumed that this is the instant when the charge front reaches the anode of the device. The derivation starts with the assumption that the field at the cathode is zero, $E_c(t) = 0$.

The next assumption is that the conduction component of the current is zero when evaluated at the anode of the device, $j_{cond} = 0$, as no charge has reached the anode yet, leaving $j(t) = \varepsilon\frac{\partial E(t)}{\partial t}$. This, combined with (3.6) and the prior assumption that the field is zero at the cathode, results in a differential equation having a solution as provided by equation 3.9.

$$L = \mu\int_0^\tau E_a(t)dt \tag{3.10}$$

$$L = -2L\left[ln\left(1 - \frac{\mu V t}{2L^2}\right)\right]_0^\tau \tag{3.11}$$

$$\tau = \frac{2L^2}{\mu V}\left[1 - exp\left(-\frac{1}{2}\right)\right] \cong 0.787\frac{L^2}{\mu V} \tag{3.12}$$

After the field at the anode has been determined, it is now possible to calculate the time it takes for the front of the space charge to reach the anode. Assuming that the space charge's velocity is given by $\mu E(x,t)$, and the field is constant between the charge front and the anode, it can be assumed that the field experienced by the front of the space charge is equal to the field at the anode, $E(x,t) = E_a(t)$.

The time it takes for the charge front to cross the device can be determined by solving the integral (3.10), to get an analytical and approximated solution (3.11). This analysis has arrived at a significant equation of the SCLC model which has been widely used in the literature on current transients, frequently to determine the mobility of the space charge. To determine the equation's validity, it is necessary that the original model of the device's space

charge traversing holds true.

3.1.3 Additional Considerations

Naturally, SCLC is not the only model used to explain these transients. While SCLC argues that the transient is the outcome of a substantial ionic current, it could also be inferred that the ionic currents are insignificant, and the transient results from a variation in the electronic conductivity of the bulk and interfaces [248]. This conclusion was reached by simulating the redistribution of vacancies, electrons, and holes within the BST oxide layer bounded by two Schottky contacts at the $BST - Pt$ interfaces. The simulation was conducted using a finite difference method, accounting for the redistribution of each particle species that is influenced by drift and diffusion.

$$j_k = -D_k \frac{\mathrm{d}C_k}{\mathrm{d}x} + \frac{Z_k}{|Z_k|} \mu_k C_k E, \text{ where } k \in \{n, p, V_{\ddot{o}}\} \tag{3.13}$$

The simulation iteratively establishes the particle redistribution and subsequently computes the total device current, which is based on the velocity and distribution of the particles. The total device current is composed of the electron, hole, and vacancy currents. For each particle species, the current comprises a diffusion and drift term, as outlined by the Nernst-Einstein equation 3.13, where k indicates particle species which can be electrons, holes or charged oxygen vacancies, j_k is the current density, D_k is the diffusion coefficient, C_k is the particle concentration, Z_k is the charge in Coulombs, μ_k is the mobility, and E is the local electric field. This method is beneficial since it allows for separation of the electronic and ionic currents, which is not feasible in an experiment.

By varying the temperature of the simulated device, it was determined that the activation energy for increased conductance is 0.8eV, which is deemed reasonable for vacancies within a BST device [412]. However, as the peak of the transient has already taken place by the time the vacancies have redistributed to a considerable extent, it is advisable to refrain from asserting that the activation energy inferred from the peak characterises the migration of vacancies.

Instead, the published results suggest that this could be attributed to the migration of electrons and holes, which predominantly occurs prior to and during the peak. Other researchers have also reached a comparable conclusion [412]. For instance, electronic traps were observed with apparent activation energies ranging from 0.18-0.3eV. It was concluded that oxygen

vacancy migration results in alterations of electron/hole concentrations, ultimately regulating bulk conductivity.

With two conflicting theories in existence, certain studies aim to determine whether the effect is electronic or ionic (SCLC). For instance, transients in some devices were analysed by considering both the SCLC model and the modulation of electronic conductivity [357]. It was found that the model of electronic conductivity produces a more credible value for the dielectric constant, suggesting it is the superior model.

However, a different study [69] aimed to determine whether the effect is due to ionic or electronic currents by assuming that electronic traps would be influenced by the oxide's crystallinity. It was discovered that almost identical transients were observed in the amorphous and polycrystalline devices, suggesting that the effect is of an ionic nature, which is also further supported by a previous claim [404].

However, in a subsequent publication [70], it was determined that an alternative theory, based on the existence of shallow electronic traps, also favoured an electronic explanation. It seems that the issue of whether this effect is ionic or electronic has not been definitively resolved. Nonetheless, the prevailing trend in the literature has been towards mostly electronic conduction, which is altered by the movement of vacancies.

Retrospectively, the SCLC model has been found to be helpful, but it has not yet been fully validated. It has been observed that higher temperatures and applied voltages cause an increase in the rate of the transient as predicted, and peak time plotted against reciprocal voltage has been found to be linear in certain regimes. However, the original space charge limited current (SCLC) model can enables us to provide more detailed scrutiny.

Take the thickness dependence of peak time, as stated in (3.12), as an example. This phenomenon results from vacancies traversing the oxide from one electrode to the other, and it should lead to the peak time being proportional to the square of the device thickness, L. Although no research has specifically investigated this issue, published data from a related study can be utilised to provide a preliminary assessment of this dependence.

Even more concerning is the predicted straight line fit for peak time versus reciprocal voltage plots, which is often utilized as the primary indicator of SCLC behaviour (3.12). Of the papers that present a straight line, many of them only have a few data points claiming a

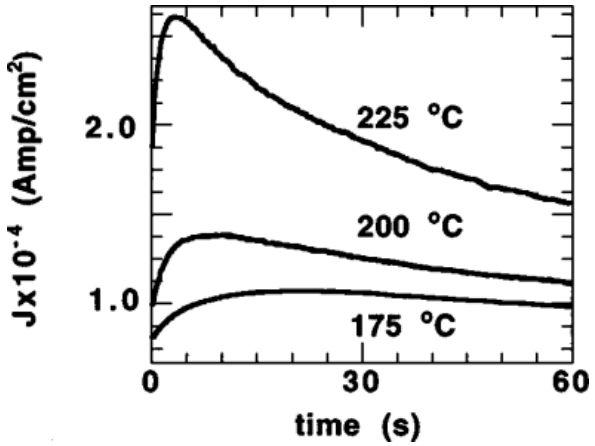

Fig. 3.2 Transients are plotted for BaSrTiO3 (BST) devices [27]. The peak at lower temperatures is significantly wider than at higher temperature.

straight line. Additionally, these plots are reliant on measuring peak times, which is not an easy process, as shown in Figure 3.2, where the shallowness of some curves obstructs the peaks.

This introduces a large degree of uncertainty for shallower transients, making the selection of peak times potentially subjective. What's more, some studies have shown that this linearity only occurs within a specific region while in others, it becomes exponential instead. Although these inconsistencies have not been studied thoroughly as yet, they do warrant a scepticism in applying the SCLC theory to the current transients.

The importance of verifying these models is exacerbated by their use in determining physical properties of devices. In some cases, these properties include mobility values and activation energies taken directly from the SCLC equations. However, further studies have delved deeper into this area. Naturally, these findings carry significant implications, particularly in the nascent field of defect engineering. The validity of the SCLC model applied in these contexts determines the certainty of these conclusions.

3.2 Methodology

The devices investigated in this thesis were developed by the Electronic Materials and Devices group in the department. Despite the fact that the fabrication process was described in detail here for completeness, the tasks described here were not personally carried out, so all credit goes to the rest of the research group.

3.2.1 Device Fabrication

The device investigated in this thesis has a metal-insulator-metal (MIM) structure and is manufactured on a silicon wafer. A thick silicon dioxide layer is thermally accumulated onto the wafer preparatory to the bottom electrode to prevent interactions between the bottom metal contact and the wafer. After that, the bottom electrode and thin film oxide are deposited unpatterned throughout the whole sample. Finally, during the deposition process, the top electrical contacts are patterned into squares with sides varying from $200\mu m$ to $800\mu m$ in Figure 3.3. Photolithography is not employed for patterning since a contact mask is used.

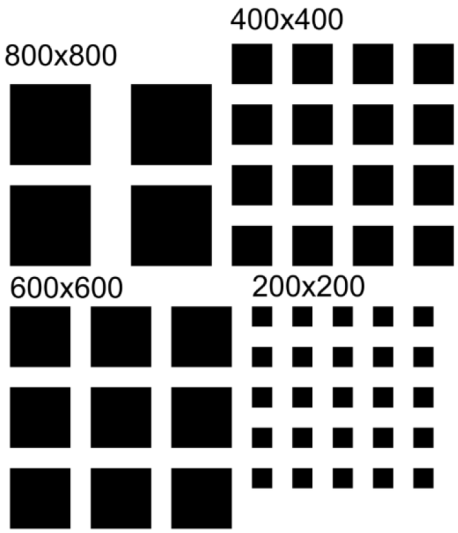

Fig. 3.3 Device layout and dimensions of the top electrical contact.

To increase adhesion, a second titanium buffer layer is placed between the top metal contact and the oxide. This adhesive layer is less than ideal since it can cause additional imperfections to migrate within the oxide. Despite this worry, research using electron energy loss spectroscopy (EELS) and transmission electron microscopy (TEM) have shown no evidence of titanium interface migration in the devices [242].

It has been claimed that asymmetry in the device's construction, as well as an active and inert electrode, are necessary to identify stable switching. The molybdenum contact can be crucial as an oxygen reservoir, rapidly exchanging oxygen between the electrode and silicon oxide layer, which is similar to an active electrode, according to a recent experiment [56]. The materials used for the top and bottom electrodes are different and weren't explicitly chosen for this project; rather, other group members had already picked them to create

high-performance resistance switching memory.

The device layers remain mostly unchanged throughout the investigation. The top electrical contact is made of a different material in the experiment than the bottom electrical contact, which is made of a thin film of molybdenum. The oxide layer is made of an amorphous silicon oxide thin film. The selection of gold as the top electrical contact may cause gold atoms to diffuse through the film and migrate into the oxide.

Although gold has been seen to diffuse through silicon oxide layers at high temperatures, resistance switching memory frequently overlook this phenomenon or presume it does not happen. A profilometer is used to assess the thickness of the layers. To guarantee excellent conductivity throughout the device, the bottom electrode is 300 nm thick. The oxide slim film is 35nm in depth. The thickness of the top electrical contact varies depending on the substance; gold has a thickness of 110 nm, while ITO has a thickness of 50 nm.

RF sputtering, a physical vapour deposition process, was used to deposit all of the device layers. Deposition is carried out at low pressure in a typical inert gas environment by blasting the intended material with a plasma, which causes the expulsion of atoms from the target. Depending on the gas pressure inside the chamber, the expelled atoms either follow a direct ballistic path or take a random walk until they land on the sample. A greater gas pressure will result in more collisions and an increased random walk, whereas a lower gas pressure produces a more direct ballistic trajectory.

The substance being deposited, known as the target material, is initially solid. By applying a strong electric field to the sputtering gas (argon), the plasma is created. Either a DC or an AC field is possible. However, an AC field that oscillates at an RF frequency of 13.56MHz is necessary for dielectric targets like SiO2. Sputtering often results in amorphous films with sub-stoichiometric oxides. The devices' SiOx oxide has a stoichiometry of 1.9, while the film's roughness appears to be determined by the RMS roughness of the underlying molybdenum layer, which ranges from 0.9 to 1.5 nm [162].

After being sputtered, film thickness is measured via a contact profilometer with a 0.5nm precision. During this procedure, a diamond tip is used to make contact with the sample and scan across the surface. Utilising a feedback loop, the tip's height is adjusted to maintain a consistent force against the sample's surface as it scans, giving the measurement of the sample height. The sample's surface height changes in direct proportion to the change in tip

height. Layer thicknesses of a device stack are measured in relation to one another using a staircase-like pattern that is created during production.

3.2.2 Electrical Characterisation

The amount of current passing through the device is a significant observable. This includes details on the oxide layer's bulk conductivity as well as the interface barrier heights. The difficulty, however, is in minimising any deviations or nonlinearities brought on by the measuring apparatus itself, with probe contact resistance serving as one such example. It is necessary to choose how to make contact with the device electrodes before conducting current measurements. There are essentially two methods: either the circuit is wire bonded inside a chip carrier, or the contacts are directly probed with tiny metallic probes using micromanipulators.

After deciding on a contact technique, the next choice is how currents will be monitored. Both a 2-wire measurement and a 4-wire measurement are frequently available as options. The sample's conductivity serves as the basis for the decision. The easiest way to measure electrical resistance is to apply a set voltage and track the total current passing through the object. Only two electrical connections are formed, thus the term "2-wire measurement" for this procedure. Ohm's Law is used to determine the device resistance by connecting a voltage source, an ammeter, and the device in series.

One of the most crucial parameters to take into account when describing thin films is contact resistance, which may be reduced by placing metal contacts on the sample during manufacturing. Fortunately, the device resistance usually outweighs the electrical resistance, making this method valid in the majority of instances. However, when resistance is small, the parasitic resistances of the measuring circuit become notable and must be eliminated by using a 4-wire resistance measurement.

Thus, the device resistance determines whether to use a 2-wire or 4-wire resistance measurement. The devices examined in this thesis have high resistance, ranging from kiloohms to megaohms. The parasitic resistances of the measuring circuit, like the contact resistances, are insignificant at this level. The issue of measuring device currents must now be solved once the device has been attached. Again, there are a variety of techniques that might be applied; the one selected will often depend on the size of the current being measured.

The average current range for the devices is 100nA to 1mA, therefore a picoammeter is required to detect considerably lower currents on the order of picoamps to nanoamps. Picoammeters minimise current readings by a number of methods that differ across manufacturers. The majority of them employ a transimpedance amplifier to magnify the signal while an op-amp converts the input current to a voltage. Once again, how this is implemented differs from manufacture to manufacture and is frequently protected intellectual property that is not revealed. The equipment used in this instance is the Keithley 6430 sourcemeter, which combines a picoammeter and a low noise voltage source into a single device.

In some cases, the device requires the application of voltage transients that are more complex than step potentials, such as pulses or custom spike trains. In these cases, the Keithley's sampling frequency is insufficient to generate such signals. An arbitrary signal generator is used in its place to create voltage transients, and a current preamplifier is connected in series with the device to amplify device currents. In particular, the oscilloscope (Rigol DS4024) and current preamplifier (SR570) are used.

3.2.3 Device Stressing

The method for reliably and repeatedly inducing the current transient phenomena is presented in this section. This not only enables the induction of current transients in a range of devices, but it also enables the progression of the transient from a barely detectable state to the dominant device activity. The approach described here has opened up the possibility of using the behaviour as a computational device while also enabling a more detailed characterisation of the behaviour.

Current transients were previously believed to be caused by oxide imperfections implying that if a device exhibits current transients, it only has to be designed as a faulty device. Fortunately, the fabrication and control of oxide defects is a current study area and is also known to as defect engineering. The act of applying sufficiently strong electric fields to a MIM device to cause a partial and reversible breakdown is known as electroforming. This is a common technique in memristor- and resistance-switching-memory (ReRAM)-based research, and it can also be simply seen in the silicon oxide devices utilised in this thesis.

Current transients were previously believed to be a result of oxide imperfections implying that if a device exhibits current transients, it essentially has to be designed as a faulty device. Fortunately, the fabrication and control of oxide defects is an ongoing research area and is also known to as defect engineering. The act of applying sufficiently powerful electric fields

to a MIM device to cause a partial and reversible breakdown is known as electroforming. This is a common technique in memristor- and resistance-switching-memory (ReRAM)-based research, and it can also be simply seen in the silicon oxide devices utilised in this thesis.

Observing that the conductance of a device showing transients is more comparable to the HRS of an electroformed device than to the LRS can provide insight into the link between electroforming and a current transient device. This led to the idea that, if a slightly more subtle electroforming technique were utilised, that is, before the major switching event, the current transient may be created. Because the electroforming process itself seems to be the outcome of a positive feedback mechanism, it is difficult to achieve a delicate electroformation [172].

When a voltage flows to the device, charge is injected into the oxide, creating defects such vacancies that increase the device conductance. This procedure has inherent instability. Higher current densities brought on by the increase in conductance speed up the creation of defects, which in turn causes current densities to keep rising until a catastrophic breakdown happens. When high constant voltages (13V to 15V) are supplied to the device, it is easiest to see this exponential rise in conductance. A regulated, gradual, and delicate modulation of device conductance is difficult to perform when this positive feedback effect dictates the change in conductance.

It is obvious that electroforming's fundamental positive feedback must be avoided. This is accomplished by abandoning voltage-based electroforming in favour of constant current electroforming, often known as "stressing the device" to distinguish it from the more widely used electroforming. A comparable voltage is provided at the peak of continuous current straining as is done during electroforming. The main distinction is that once the device begins breaking down, the voltage is reduced as a result of the current source's negative feedback.

Instead of applying a set voltage, the device is provided with a constant current. This has the benefit of reducing the voltage across the device as it grows more conductive in order to maintain a constant current, which in turn delays the creation of oxide defects. This will go on once an equilibrium is established when the applied voltage is decreased to a level where formation no longer takes place, but is still great enough to keep the continuous current flowing. Now that the process presents negative feedback, it is considerably more conducive to causing subtle alterations in device conductance.

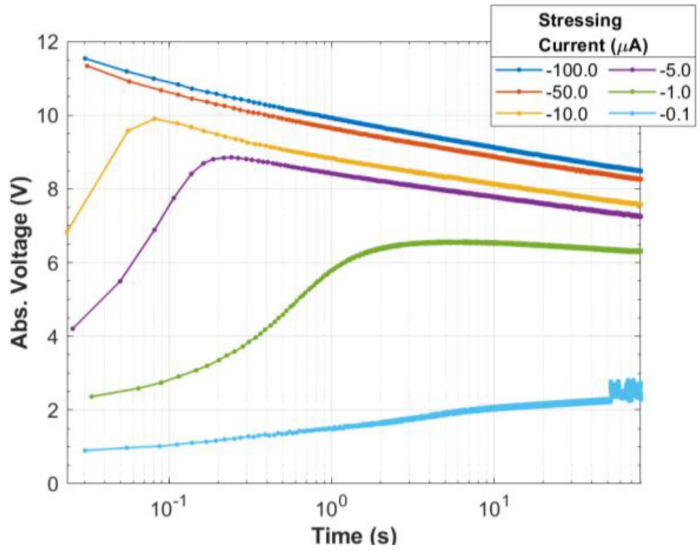

Fig. 3.4 The device's voltage is recorded while subject to constant current stress, with six devices tested at various current levels. The voltage across each device is plotted during the initial constant current stress for six devices, each with a different current magnitude.

Essentially, the devices are subjected to stress by grounding the molybdenum contact and applying negative currents to the gold contact, resulting in electron injection at the gold contact and hole injection at the molybdenum contact. The magnitude of the constant current is adjusted based on the desired behaviour, with values ranging from $-0.1\mu A$ to $-100\mu A$ for a duration of 100 seconds. During stressing, the voltage induced across the device gradually decreases, eventually reaching a steady state value as shown in Figure 3.4.

It's important to take into account that after being stressed, the device doesn't immediately display the current transient. The relaxation process can be accelerated by providing a positive potential to the above gold-titanium contact, but it must still be allowed to rest for 24 hours. When allowed to rest, the device will eventually show current transients. This delayed response is not unexpected as a result of a drifting defect with restricted mobility. While the generated defects are still present, the relaxation time allows this to return to its resting distribution at thermal equilibrium. It is expected that the stressing results in significant drifting and defect generation.

3.2.4 Induced Transient

The established method enables the induction of current transients in devices, and it may be done so gradually. The prominence of the current transient can be altered when the devices are subjected to varying levels of stress by varying the current's magnitude. To

be more precise, a device's current peak becomes more evident the more stressed it is. It was previously claimed that this was caused by the thinner devices' higher current densities masking the current transient [70].

Instead of looking at current densities, a possible reason of the prominence of current transients may be based on the total number of defects present in the oxide. It follows that thicker devices will have more defects overall if it is assumed that the concentration of defects in the oxide layer remains relatively constant throughout growth. In this instance, it is also assumed that more oxide defects will be produced the more aggressively the device is stressed. Therefore, it is possible that the total number of defects present in the oxide, rather than the conductance of the device, can be used to predict the transient's amplitude.

A device being stressed does not ensure that there will be current transients. As soon as electrons are introduced at the top gold-titanium contact, the device only displays current transients when it is stressed with negative currents. The response is different when electrons are injected at the molybdenum contact. The structural alterations that stressing causes in the device can be used as evidence for the reason only negative currents can cause current transients. As anticipated, the stressing process results in structural flaws at the electrical contact, as has been seen in the past when comparable devices were electroformed and switched repeatedly [371, 372].

Negative currents are used to stress devices, which involves grounding the molybdenum contact while applying negative voltages to the gold contact to inject electrons into the oxide. This causes a high number of tiny holes to form in the contact, which are often clustered together in one area. It is unclear where these traits came from. Since the specific area where the faults are present does not match the location of the probe on the object being examined, a mechanical explanation is ruled out.

When the device is strained with positive currents, or electrons are introduced into the molybdenum contact, noticeably distinct faults are generated. The overall contact looks to get rougher, not just the individual holes. The flaws in this instance are repeatable across devices and cover practically the whole contact. It should be emphasised, nevertheless, that these flaws are not necessary for observing current transients, and they were avoided in the devices examined here.

3.3 Results

Current transients are difficult to define because they are elusive, sometimes mistaken for faults, and only seen in a small number of the devices in a batch. This is due in part to the difficulty of developing devices that reliably produce transients. It is ideal to be able to gradually create an effect and link its presence to any changes in device attributes in order to define a behaviour in detail. It is understandable that the analysis that could be done was restricted since published research on current transients relies on the irregular appearance of current transients. Therefore, it is obvious that the lack of a method to create transients is a crucial tool if we are to fully comprehend this phenomena.

3.3.1 Neuromorphic Behaviours

In essence, the behaviour of the current transient phenomenon can represent the ability to exhibit both analogue potentiation and depression under the same voltage polarity and amplitude. Potentiation and depression are two fundamental processes that occur within synapses and are the foundations of more complex learning rules. During potentiation, a synapse's conductivity increases, whilst during depression, it decreases.

Changes caused during potentiation or depression can also vary in volatility: they may persist and be long-term (Long-Term Potentiation (LTP) and Long-Term Depression (LTD)) or their effects can reverse over time and be considered short-term (Short-Term Potentiation). be regarded as short-term (Short Term Potentiation (STP) and Short Term Depression (STD)).

The device's potentiation and depression can be best demonstrated by applying a step potential. When a negative step potential is applied to the top gold electrode of the device, transient current is observed, as shown in Figure 3.5. Initially, a rapid potentiation occurs, taking just a few seconds, but it quickly reaches a peak beyond which the depression phase begins. The device conductance is gradually dominated by competition with depression, ultimately causing the conductance to fall below its original level. This occurs over a longer period of time, lasting tens of seconds and continuing for tens of minutes.

The transient current response of the device to a range of DC voltages is plotted in Figure 3.5. For each voltage the mean of 3 trials is plotted, with error bars indicating the maximum and minimum of all trials. At lower voltages, i.e. -1 V, negligible depression is observed, but it becomes progressively more prominent at larger voltages. Potentiation is observed for all

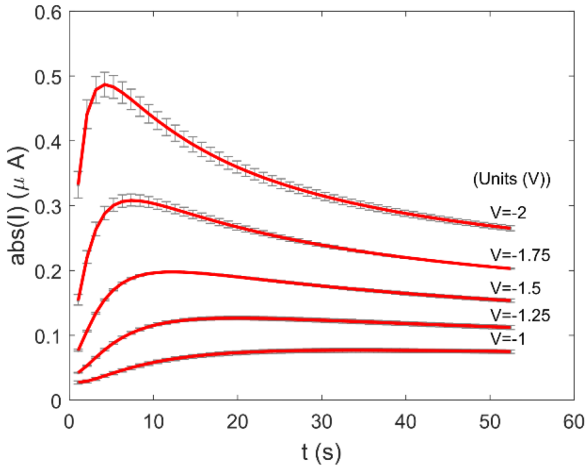

Fig. 3.5 The transient currents of amorphous silicon dioxide thin films were analysed for various voltages. The voltage is applied to the gold electrical contact while the molybdenum contact is grounded. For each voltage, the average of three trials is recorded and presented graphically, with error bars indicating the range of the three trials.

voltages, including -1V, and appears to accelerate with increasing voltage.

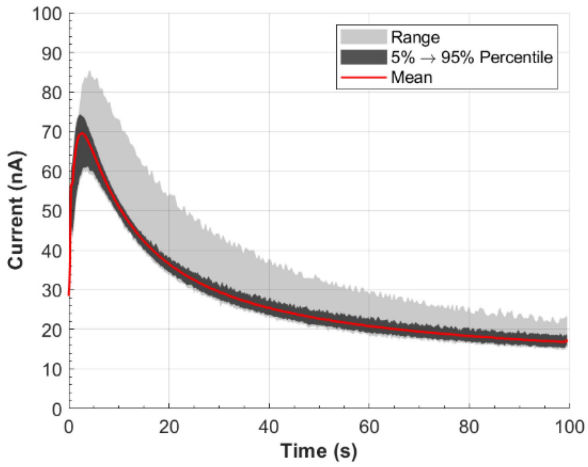

Fig. 3.6 The mean, range and 5th - 95th percentiles are presented graphically for 132 current transients that were induced by applying a -0.8 V step potential to the gold electrical contact while the molybdenum was electrically grounded. A single-order low pass filter with a cutoff frequency of 10 kHz was applied to the data to eliminate noise that may mask the variation between trials.

It is worth highlighting the repeatability of the device's response as presented in Figure 3.6, where a collection of 132 current transients induced by a step potential of -0.8 V applied to the gold electrical contact is displayed. Although resistance switching devices frequently

display variable and stochastic responses, it is apparent that the device's performance is consistent and foreseeable, as demonstrated by the range (light grey) and the 5th and 95th percentiles (dark grey). The range has a wider spread than the 5th and 95th percentiles, attributed to bigger currents observed in the first three trials, resulting from a settling process across numerous trials.

This statement only holds true if the device is allowed to fully relax between trials. To achieve relaxation, both electrical contacts should be grounded and the device left to rest. The reset process is gradual, necessitating a resting period of one hour to guarantee complete relaxation. Accelerated relaxation can be attained by applying a positive potential to the gold contact, as opposed to grounding it.

While this behaviour is noticeable during step potentials, it is crucial to examine whether the same behaviour can be reproduced when operating neuromorphic synapses using pulse trains. We demonstrate that this is achievable by administering a sequence of Gaussian pulses to the appliance. The pulses have a full width half maximum (FWHM) of 20 ms, an amplitude of -3 V, a period of 300 ms, and are once more implemented on the gold contact. Note, while the pulses have a negative amplitude, the device current has been inverted in the following figures to enhance clarity.

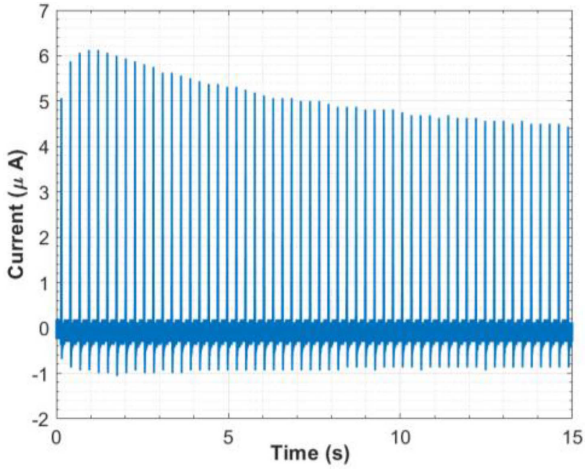

Fig. 3.7 The current that passes through the device while a spike train is applied is measured. The pulses are applied to the gold electrical contact while the molybdenum contact is grounded. Each pulse has a Gaussian form with a Full Width at Half Maximum (FWHM) of 20 ms, an amplitude of -3V, and a period of 300 ms. The magnitude of the current has been inverted for clarity. The device was electrically stressed with a constant current of $-100\mu A$ for 100 seconds.

The response to the series of Gaussian pulses is shown in Figure 3.7, indicating a behaviour similar to that in the DC measurements. An initial potentiation lasting about four pulses is followed by a longer depression period. This indicates that a combination of potentiation and depression can be achieved in both DC and spike train operation, promising the suitability of the device for use in spiking neural networks.

One possible advantage of the subthreshold regime lies in its substantial resistance value (approximately 10 MΩ). Typically, resistance switching devices applied for neuromorphic computing alternate between high resistance states (about 100kΩ) and low resistance states (about 1kΩ).

Nevertheless, in the subthreshold regime, our device remains within a range of resistances comparable to or even higher than the high resistance state of binary resistance switching devices. This indicates that the device may function with lower current consumption than a typical resistance switching device.

However, it must be acknowledged that inherent limitations exist. The broad range of resistance in switching devices reduces their sensitivity to noise or voltage fluctuations, while also enabling a wider range for programming. Thus, there exists a trade-off between current draw and resistance to noise. Additionally, the subthreshold regime poses the challenge of voltage dependency in depression mechanisms. As depicted in Figure 3.5, depression is seldom observed below -1V. This, in turn, constraints the operating voltage of subthreshold circuitry wishing to utilize depression dynamics.

3.3.2 Transient Tunability

The capability to choose between potentiation and depression is crucial for circuit designers who intend to exploit the device in neuromorphic circuits. The more adaptable approach is to regulate the magnitude of the applied pulses. Figure 3.5 demonstrates that lower voltages, such as -1V, display a certain level of potentiation but not depression. This is also true for low-voltage spike trains.

Figure 3.8 depicts the current response to spike trains of varying pulse amplitudes. Depression is not observed with lower voltages. It is possible that the absence of depression at lower voltages indicates a threshold electric field required to induce the change from potentiation.

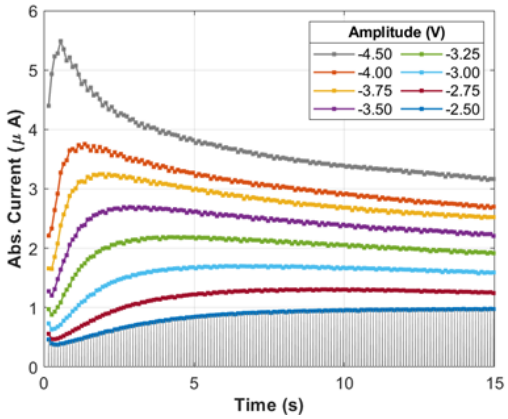

Fig. 3.8 The plot illustrates the device current for Gaussian spike trains with increasing amplitude. Grey-coloured Gaussian current pulses represent the smallest amplitude (-2.5 V). To improve clarity, only the peaks of each pulse are plotted for higher amplitudes. The pulses have a Full-Width at Half-Maximum (FWHM) of 20 ms and a period of 100 ms. Priorly, the device underwent electrical stress when exposed to a constant current of -50μA for 100 seconds.

This implies overcoming an activation barrier to initiate defect drift.

On the contrary, depression may be the intended response. In this instance, the amplitude of impulses may be raised to the point where the initial potentiation is overcome. Figure 3.9 depicts the percentage upsurge in device conductance from its initial value. The experiment is replicated for spike trains of differing amplitudes. It can be seen that spike trains with an amplitude < -3.75V are largely potentiating, with the change in conductance remaining positive. However, when the amplitude is increased to -4.5 V, the spike train leads to negative changes in conductance, resulting in depression.

An alternative and enduring method to select a particular behaviour is to modify the electrical stress that the device experiences after production. As outlined earlier, the devices undergo initial stress through the application of a constant current to the device. The current magnitude is modifiable, which, in turn, alters the degree of stress imposed on the device, thereby allowing adjustment of the device's current transient response. In figure 3.4, the response to stress is then plotted for six different devices, with each being subjected to varying currents ranging from -0.1 μA to -100 μA. Each device is subjected to the current for 100 seconds.

The resulting current transients display a noticeable potentiation for higher stressing currents, accompanied by an increasing conductance. However, the device stressed with the smallest

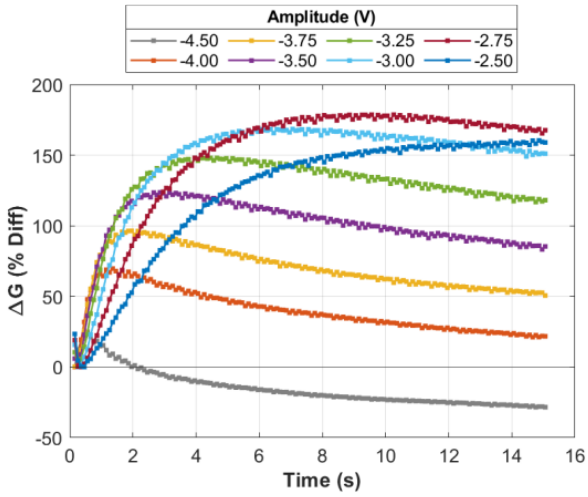

Fig. 3.9 The percentage difference in conductance caused by each spike train pulse is represented on a graph. Larger amplitudes mainly cause depression, whereas smaller amplitudes lead to potentiation. Before testing, the device underwent electrical stress through a constant current of -50μA for 100 seconds.

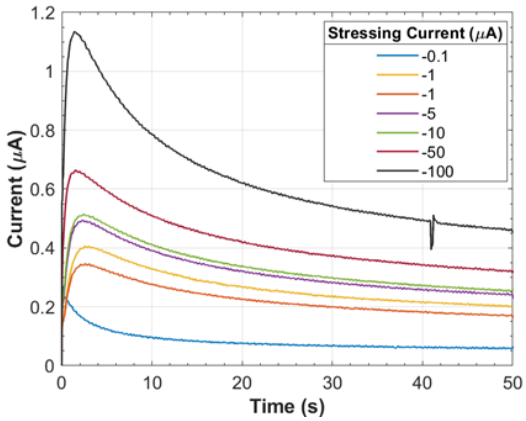

Fig. 3.10 The device current is plotted in response to a step potential of -1.25V for devices initially stressed to varying degrees. There is no potentiation in devices stressed to a lesser extent, while heavily stressed devices exhibit greater potentiation and depression. For clarity, the absolute value of the device current has been plotted.

current, -0.1μA, shows no potentiation, only depression. Conversely, the device exposed to the maximum current, -100μA, potentiates up to almost 10 times its initial conductance.

These methods offer the chance to modify the level of potentiation taking place in the device, ranging from minimal to noticeable. It is crucial to observe that this adjustment is irreversible and would typically be determined during circuit manufacture. In contrast, the method of altering spike amplitude is adaptable and can be modified while in use.

Given the device's ability to display both an increase and decrease in conductance, each occurring at varying rates and with different relaxation rates, it is unavoidable that the steady state conductance will differ across various spike train periods. Before investigating the computational potential of this behaviour, it must first be confirmed.

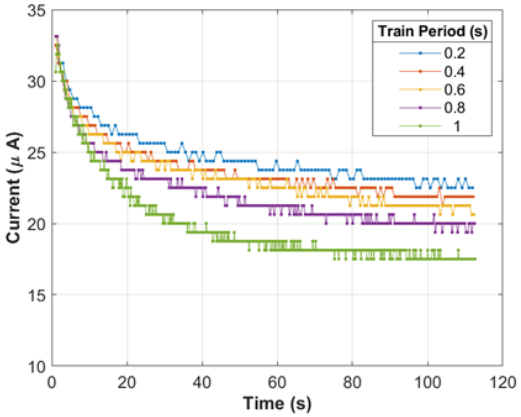

Fig. 3.11 The device's maximum current is graphed for each pulse applied to a 400 by 400 μm device. The pulses take the shape of a Gaussian curve with a FWHM of 50 ms and an amplitude of -2.25 V. For clarity, the absolute value of the device's current has been shown.

In Figure 3.11, a graph plots the device current produced by spike trains with varying inter-spike time periods. The device ultimately settles into a steady state conductance in response to each spike train, where conductance changes induced by the applied spikes are counterbalanced by the device's relaxation dynamics. As anticipated, the steady state conductance is found to be dependent on the frequency of the pulses applied to the device - higher frequency pulses lead to decreased conductivity. These findings imply that this conduct could potentially be utilised to adjust the device resistance in response to spike trains of different frequencies.

3.3.3 Homeostasis Applications

The original aim of this study was to illustrate the potential application of distinctive memristive device behaviour for innovative computing architectures. One noteworthy application to emphasise is the neuron's capacity to adjust based on a firing frequency analogous to homeostasis in biology [339].

In biological systems, homeostasis refers to the maintenance of preferred operating conditions or particular states [348]. An example of this is the regulation of body temperature. When applied to spiking neural networks, homeostasis plays a role in maintaining spiking activity to avoid excessive power consumption through unnecessary spike events. It can also protect against faulty neurons entering a chaotic, high-activity state. Without homeostasis, such events could cause the entire network to enter a chaotic state [226].

This process is also known as habituation, a type of homeostasis in which a change in input results in a temporary change in output that eventually diminishes to a steady-state value. There are already a few practical examples of this in recurrent neural networks, one of which resulted in a 20% improvement in classification accuracy on the MNIST dataset when compared to a neural network without homeostasis.

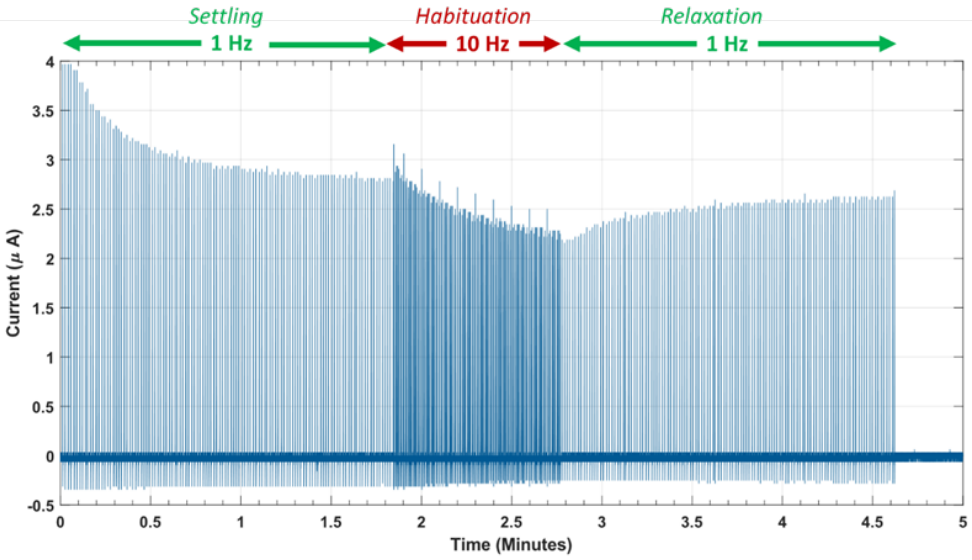

Fig. 3.12 Response of the device to varying frequencies of spike pulses. The device was initially operated with a sequence of 1Hz spike pulses, followed by a period of 10Hz spike pulses, before finally returning to 1Hz spike pulses.

The previous section showed that the steady state conductance of the device is reliant on the firing rate of the related neuron. The device conductance decreases at higher spiking frequencies, but has a greater steady state at lower frequencies. This behaviour is suitable for inhibiting the impact of suboptimal input neurons that have entered into a chaotic state of high frequency.

Figure 3.12 demonstrates this concept in practice. The unit is connected to a signal generator that produces Gaussian pulses with a low frequency of 1Hz, which is the neuron's typical background activity. In these conditions, the device attains a steady-state conductivity. An input neuron is simulated as transitioning to a dysfunctional state of higher activity at t = 1.8 minutes, potentially caused by circuit damage. The frequency of the spike train is increased to 10Hz.

Without homeostasis, this mistaken input could cause the linked output neuron to reach a comparable state, which could then transmit through subsequent layers of the neural network. Luckily, when subjected to the high-frequency spike train, the conductivity of the device reduces due to the enduring behaviour of the current transient device. This decreases the overall current supplied to the linked output neuron, decreasing the chance of this high-frequency input triggering equivalent actions in the following neurons.

While the temporary suppression is advantageous for safeguarding the network against excessive firing, it should not be a permanent solution. If the input neuron recovers to normal functional conditions, the inhibition should be lifted, and the neuron should be allowed to resume its place in the network. At t = 2.8 minutes, evidence was presented to support this idea. With a 1Hz spike train, the sample recovers to the background level of activity. The device conductance responds to the fluctuation of current transient conductance changes by returning to its former steady state value prior to the neuron's faulty phase.

The ability of protective systems to return to their original state when normal operating conditions are restored is a fundamental characteristic of homeostatic processes. Although the concept of homeostatic habituation has been demonstrated with the current temporary device, there are still limitations. For instance, it is impossible to determine a specific and constant conductance value while the device is operating in its present transient state. If the device is electroformed similar to a regular memristor, it allows for the weight to be adjusted to a variety of analogue values.

To achieve homeostatic behaviour in a physical system, it may be necessary to connect the device in series with another device that is programmable. The good news is that the programmable device and the current transient device are architecturally identical and manufactured using the same method. The devices can be combined on a single wafer since the only difference between them is the method in which they are electrically stressed.

3.3.4 Physical Implications

Initially it was thought that the transient was the consequence of an ionic current superimposed on an electronic current [224, 184]. The ionic current is the outcome of a field-driven drift of some space charge, which is often assumed to be charged oxygen vacancies in more recent research [298]. The variations in conductance seem to stem from two distinct modifications happening simultaneously in the sample. An ionic current could only account for this observation if the drifting space charge were capable of drifting at a constant velocity permanently.

A more appropriate explanation is that the transient is an electronic current that is altered by changes in the device's conductance, with two simultaneous changes causing the transient [248]. This is because the electronic conductance of an MIM device can be modified in several ways, some of which may occur simultaneously. The literature review identifies a general trend favouring an electronic explanation over an ionic explanation.

There are, however, numerous mechanisms through which the electrical conduction of an MIM device can be adjusted. An MIM device can be abstracted as a series combination of three distinct components: a metal-insulator interface, a bulk insulator, and a second metal-insulator interface. These components determine the device's conductance; however, in specific cases, one of these three components may be significantly more resistive than the others. In such circumstances, that particular component alone may provide a good estimate of the device's conductance.

So far, there has been no research to determine which layer, if any, dominates the conductance of a device that shows current transients. Most devices described in literature display rectifying behaviour, implying an interface-defined conductance. However, it is important to note that even if a device is rectifying, this does not necessarily indicate that the dynamics are always accurately described by an interface model.

In the reverse polarity, when the interface exhibits the least conductivity, the conductance of the device can be accurately captured by the interface model, since it is expected to have high resistivity. However, in the forward polarity, when the interface is most conductive, the bulk oxide may have a greater impact on describing the device conductance, necessitating a bulk-based model. Without knowledge of which layers define the device conductance, it is necessary to investigate every layer within the device to determine the physical changes that could be causing the current transients.

The bulk oxide, where the crystalline oxide is highly insulating as a result of the oxide's large bandgap. At high voltage and low temperature, current through the oxide occurs predominately via tunnelling- either directly or via the Fowler-Nordheim mechanism from Table 1.1. However, the presence of electron trap states within the oxide results in the possibility for higher current densities through the use of these traps as alternative conduction pathways.

This thesis examines a sample of amorphous silicon oxide. Such oxides contain a higher concentration of oxygen vacancies, which affect electronic conduction by acting as electron/hole traps. Additionally, these oxides have an efficient pathway to creating oxygen vacancies. In the amorphous phase, the silicon dioxide has wide $O-Si-O$ bond angles [71].

The sites can trap a maximum of two electrons in this broad bond, which decreases the energy barrier required to generate the vacancy [92]. The existence of oxygen vacancies and an efficient route to their generation results in the degradation of the oxide. Conduction takes place via vacancy trap sites despite the wide bandgap.

There is evidence of the formation of conductive bridges through the oxide via trap states according to TEM [258, 383] and CAFM [29] etching techniques. At higher temperatures, trapped electrons can be excited from one trap state within the bandgap to its neighbour, or across the oxide through the conduction band, via Poole-Frenkel conduction due to thermionic emission within these states [87].

$$\sigma_{PF} = q\mu_n n_{PF} = q\mu_n n_0 \left(\frac{-q(\Phi_B - \Delta\Phi_{PF})}{k_B T} \right) \tag{3.14}$$

$$j_{PF} = E\sigma_{PF} = Eq\mu_n n_0 \left(\frac{-q(\Phi_B - \Delta\Phi_{PF})}{k_B T} \right) \tag{3.15}$$

The Poole-Frenkel effect raises the conductance of an oxide, σ_{PF}, by increasing the amount of free electrons in the conduction band relative to its thermal equilibrium concentration,

n_0 to n_{PF}. The applied field reduces the barrier height, Φ_B, necessary for an electron to be thermally excited from its trap state into the conduction band, causing this rise. For an applied field, E and equation (3.15) where $\Delta\Phi_{PF} = \sqrt{\frac{q^3E}{\pi\varepsilon}}$, the drop in barrier height represents the increase in conductance produced by the reduction in barrier height [332]. Because this impact is reliant on the thermal energy of the electron, the number of extra electrons is based on a Boltzmann distribution and is determined by the thermal energy, k_BT, of the electron.

It is evident that the conductance of a device, regulated by the Poole-Frenkel effect, may be influenced by a change in the carrier concentration of the trap sites, n_0, and a reduction in barrier height, which varies with the local electric field, E. Both of these factors will be considered when examining the model as a potential explanation for current transients.

Poole-Frenkel, however, is not the sole cause of conductivity through trap sites [64]. The transfer between proximate traps can occur via tunneling, termed trap-assisted tunnelling (TAT) [143]. This method has already been proven to sufficiently account for conduction in amorphous silicon dioxide sheets showing resistance switching behaviour [240]. One potential TAT model is founded on an inelastic tunneling process, ITAT [136].

Both the Poole-Frenkel and ITAT models are capable of explaining the alterations in conductance witnessed during the current transient. The rise in conductivity, which is probably due to charge trapping, can be effectively accounted for by these models. Take, for instance, the Poole-Frenkel model (3.15), in which the current of the device results from the trap population, n_0, and the likelihood of an electron making a thermal leap from the trap to the conduction band. The model details steady-state conditions, where the trap populations remain constant.

$$E(x) = \frac{V}{d} \tag{3.16}$$

$$E(x) = \frac{V}{d} - \frac{Qx}{\varepsilon d} \tag{3.17}$$

$$E(x) = \begin{cases} \frac{V}{d} - \frac{Qx}{\varepsilon\delta} & \text{if } x < \delta \\ \frac{V}{d} - \frac{Q}{\varepsilon} & \text{if } x \geq \delta \end{cases} \tag{3.18}$$

It is conceivable, though, that this term could be treated as time-dependent. At first, with the device grounded and at room temperature, the traps may be largely depleted, corresponding to the low conductance observed when the step voltage was first applied to the device. With the

voltage now applied, however, traps that were previously depleted or had a low probability of being filled when grounded may now have a higher probability of being filled, corresponding to an increase in the probability of a trap being filled, n_0.

As the device becomes more conductive and higher current densities flow through it, the number of electrons trapped within the oxide increases. This eventually leads to a higher equilibrium or steady state value of n. A similar argument can be made for the ITAT model, which acknowledges that the probability of traps being populated varies with time and already accommodates transient effects. In order to observe an increase in conductance, the current flowing into the traps must exceed the current flowing out of the traps.

The decline in conductance may also be accounted for by positing an indirect interaction through the local electric field and a mobile defect drifting under the influence of the applied field, e.g. a charged oxygen vacancy. For instance, the Poole-Frenkel mechanism exhibits an exponential dependency on the local electric field to mitigate the trap depths faced by the trapped electrons. Decreasing the electric field would exponentially lessen the device's electrical current.

Mobile ions might have the ability to diminish the electric fields encountered by most of the traps. Consider an ideal insulator with a thickness of d, in which no space charge is trapped. In this case, the electric field would remain constant across the oxide layer (3.16). If a space charge, Q, is introduced into the oxide and distributed homogeneously, the electric field exhibits a gradient (3.17). The charge within the oxide decreases the field experienced by the traps.

However, when distributed across the oxide's thickness, the drop in potential is shared amongst many traps, resulting in less significant differences for each trap. If the charge is mobile and compelled to accumulate at one of the interfaces by an applied field, the decrease in the local electric field is focused near the interface. Consequently, numerous more traps now experience the lower electric field resulting in an overall decreased device current.

While there have been discussions about the potential explanations for the bulk effect, interface-based models may also explain the changes in conductance observed during the current transient. Memristor devices have previously demonstrated resistance switching, which interface models have explained. The models suggest the existence of a Schottky

barrier at the metal-oxide interface, based on the rectifying nature of the device.

The Schottky-like barrier arises due to a defective oxide material behaving akin to either an n-type [89] or p-type [301] semiconductor. Upon contact with the metal, charge transfer takes place between the metal and defect sites within the oxide, caused by the offset of work functions. Consequently, a space charge and electronic barrier are formed, hindering any additional charge transfer across the interface. The conductance of the device changes due to alterations in either the barrier height, which allows for larger thermionic currents, or its width, enabling greater tunnelling currents. It has been suggested that the barrier is controlled by one of two means.

One explanation is derived from interface states [308]. The interface states, also known as surface states, have an influence on the space charge within the interface [17]. These states may decrease or increase the barrier height or width depending on their type, whether they are n-type or p-type states [55]. If the concentration of these interface states is high enough, the barrier height may be solely defined by the interface states, leading to fermi pinning, rendering the metal/oxide work functions redundant. Charge trapping in these interface states could clarify the modulations within the barrier height or width. For instance, it was suggested that surface states of a metal-amorphous silicon Schottky barrier decreased the barrier height [381].

Alternatively, it has been suggested that the alteration in barrier height stems from a migration of oxygen vacancies towards the interface, determining the band alignments of the two materials [9]. Research has revealed that elevated levels of oxygen vacancies led to a larger bandgap in the bulk. This presents a model that revolves around the accumulation of vacancies, contributing to the expansion of the insulator's band-gap at the interface and thus an elevation in the barrier height.

To conclude, there are still several potential explanations that may account for the alterations in conduction that cause the current transient. However, none of them have been identified as the best explanation. If the device has bulk limitations, then the decline in conductance due to field driving could potentially be explained by a mobile space charge that reduces the local electric fields. This could apply to both a Poole-Frenkel and trap-assisted tunnelling model. Meanwhile, the increase in conductance driven by charge could be explicable through the population of traps required for electronic conduction to take place.

This applies to both Poole-Frenkel and trap-assisted tunneling models. Alternatively, if the conductance is limited by the interface, the increase in conductance due to the driven charge may be a result of trapping within surface states at the interface, which reduces the barrier height through Fermi-pinning. Meanwhile, the decay caused by the field can be explained by charge defects drifting towards the top contact, modulating the band alignments of the metal contact and oxide. Equally, while these two groups are distinct - bulk or interface limited models - the answer may well involve a fusion of the two.

3.4 Conclusion

This chapter demonstrates a device that can exhibit potentiation and depression of conductance under the same voltage polarity in the sub-threshold regime. The methodology explains how to induce this behaviour within standard resistance switching devices and how to select specific dynamics. The presented methodologies will allow researchers to replicate the behavior in their respective devices and instruct circuit designers on how to fine-tune the behavior for their specific application.

The current transient property of a device that can be designed to generate conductance potentiation and depression under the same voltage polarity. The emphasis is on immediate applications and implications for spiking neural networks. The slow dynamics of conductance depression could be useful in establishing long-term homeostasis and habituation, yet the volatile potentiation may be more appropriate for Short-Term-Memory. Similarly, the convergence of these two actions on a solitary device may have unanticipated advantages in more integrated memristive systems.

In regards to spiking neural networks, implementing synapse weight update rules such as spike-rate-dependent plasticity (SRDP) [132] can be achieved through the subthreshold regime. In SRDP, synaptic weights are updated based on the frequency of the input signal. This is different from spike-timing-dependent plasticity rules, which adjust the weight according to time intervals between spikes from the presynaptic and postsynaptic neurons. The homeostatic behaviour, which is a direct function of the spike train frequency, has an instant impact on the weight update process between neural network layers.

Moreover, the volatile nature of conductance changes in the subthreshold regime resembles a forgetting process [197]. This phenomenon has been utilised to reduce synaptic weights

without requiring inhibitory pulses, thereby simplifying circuit complexity. However, the effects of this forgetting process on network performance in the context of SNNs are yet to be comprehensively studied, and thus cannot be accurately gauged.

The applicability of this trait is restricted to circuits with slower dynamics due to its slow behavior, which may be a disadvantage in situations where speed is a concern. Fortunately, the majority of signals and monitoring techniques operate on a one-minute segment basis, making this behaviour beneficial for the future applications potential of the project [354].

If circuit designers intend to utilize this behaviour, it is imperative to address the physics that underpins it. Further verification is required for existing models like the SCLC model and other electronic explanations to develop accurate simulations/models and identify the fundamental limitations of behaviour. In order to advance research on this topic, future work should concentrate on discerning the location where these changes in conductance are transpiring, specifically at the interface or the bulk.

This could be accomplished by adjusting the electrode materials to potentially reveal interface effects, and modifying oxide thickness and area to potentially disclose bulk effects. This demonstrates potential evidence for conductance decay due to a process occurring at the interface, but it is not a definitive outcome. Conducting further research in this field could assist in identifying which of the bulk or interface models are relevant. Once the location of the alteration has been identified, the findings regarding the driving force can suggest a physical model for the phenomenon. Charge trapping is expected to enhance the device's conductance, while a field-driven drift of defects should decrease it.

The capability of the subthreshold regime to generate both potentiation and depression with identical polarity voltage pulses is a distinctive behaviour that specifically deals with the current obstacles that circuits face when sourcing pulses of opposite polarities or when neurons have access to both ends of a synapse, resulting in intricate signal routing. This operational regime has potential applications in the field of neuromorphic computing, where circuits can utilize the behaviour for single-rail power supplies, thereby simplifying circuit construction and layout. Furthermore, the intricate dynamics may offer opportunities for novel types of neuromorphic circuits, as noted in the discussion of homeostasis within individual synapses.

From a wider perspective, this operational approach additionally reinforces the concept of completely memristive circuits. Identifying and presenting an intermediate regime between

the pristine and electroformed states of MIM devices with qualitatively different characteristics has prompted consideration of how these behaviours may be combined. This raises questions regarding new computations that could be particularly suitable for enhanced signal processing applications in the future.

Chapter 4

Biorealistic Computing

4.1 Introduction

Traditional neuromorphic circuits consist of two main components: neurons and synapses [238]. Neurons are active devices that act as decision-making units. When the input threshold is surpassed, an acceptable condition is met, and a voltage spike is generated. Synapses are usually passive elements that connect neurons with each other. This is also where primary calculations of the inputs are performed, as well as a potential storage mechanism.

The neurons and synapses are linked to create a neural network, which is used in all modern machine learning methodologies. Rather than responding to continuous signals, these networks communicate information through spikes. Spike trains convey data in both the form and timing of the spike, making them ideal for implementing synaptic learning principles. It is worth noting that this method adheres to the typical framework within the field, however, it presents a highly simplified conception of the biological system.

Modern artificial neural networks and neuro-computing architectures usually neglect the principles of neuroscience [279]. As a consequence, essential elements of the organic cerebral processing systems are either disregarded or overlooked. The aim of biorealistic approaches is to mimic the functions of computational cells using electronic devices. The creation of these circuits in CMOS was traditionally known as neuromorphic engineering, with backpropagating networks being a more direct influence from nature.

4.1.1 Spiking Deep Networks

Deep artificial neural networks (DNNs), particularly convolutional neural networks, represent a significant triumph in the field of modern computer vision. They have demonstrated remarkable efficacy in recognizing a diverse array of objects within expansive, intricate images [175]. However, these networks have been engineered for and operate exclusively on rate-based neurons. The question of how they can be executed on spiking neurons represents an emerging frontier of investigation.

The question arises as to why such networks should be run in spiking neurons. There are two principal motivations behind the creation of deep spiking networks. The first is to enable the operation of some of the large CNN models that have recently demonstrated success in numerous object recognition and other tasks on spiking neuromorphic hardware. This will facilitate the development of energy-efficient systems capable of performing object recognition in real time on robotic platforms, where current technology is too energy-intensive to allow for deployment on mobile robots, for example.

The second motivation is to incorporate additional brain-like components into machine learning models. The field of neuroscience presents a multitude of distinctive challenges pertaining to the mechanisms of learning in the brain. These include the complexities associated with the nonlinear characteristics of neurons, particularly in relation to their firing thresholds, as well as the intricacies of spike-based communication and its inherent discreteness and variability.

Although the spiking deep networks presented in this chapter are not designed to be models of brain-like learning processes, the challenges addressed here are also faced by the brain. Some of the ideas presented in this section provide insights that motivate the development of more biologically plausible learning mechanisms.

Spiking deep networks facilitate the transmission of information between neurons in the form of discrete spikes. The initial distinction to be made regarding spiking networks is that they encompass an additional temporal dimension, which is not typically present in rate-based DNNs. In other words, a spiking neuron is a process that evolves over time, sometimes emitting spikes, sometimes not.

It is only possible to discuss this process over time; examining it in one instant provides little insight. In contrast, in rate-based networks, we typically present an input and can

instantaneously determine the activities of each subsequent neuron in the network, since they do not change over time.

A second notable distinction is that spikes are discrete and identical in nature. The sole information conveyed by a spike is the time at which it occurred. As previously discussed, this indicates that there are two principal categories of codes that a neuron can utilise: rate codes and timing codes.

In the case of a timing code, the focus is on the times of individual spikes. In contrast, if a rate code is employed, the focus shifts to the number of spikes occurring within a specific time window, or potentially the relative timing of spikes in relation to one another.

When examining the number of spikes within a specified time interval, it becomes evident that the resulting rate is inherently discrete. If there are n spikes within the t-second window, the firing rate can be expressed as n/t, where n is an integer. For a fixed window t, the firing rate can only assume a discrete set of values, specifically all the integer values of n.

A third distinction pertains to the inherent variability of the output of spiking neurons, which differs from that of their rate-based counterparts. In the event of a constant input, a rate neuron will provide a constant output, that is to say, the firing rate corresponding to that input.

It should be noted that rate neurons are capable of exhibiting internal dynamics, as exemplified by the adapting version of the rate-based LIF. Consequently, when presented with a constant input, the output of these neurons will not be constant. Nevertheless, their output remains considerably less variable than that of their spiking counterparts.

In contrast, spiking neurons will output a spike train, which, when filtered by a synapse, results in an oscillating signal whose variability depends on the firing rate. Consequently, even when presented with a constant input, a spiking network will exhibit variability in the inputs to each neuron and the outputs of the entire network.

DNNs are typically formulated as rate-based models, wherein the nonlinearity activity is understood to represent the firing rate of a neuron. To illustrate, a ReLU can be conceptualised as a neuron that is silent in the event that its input is less than zero, and whose firing rate increases in a linear fashion as the current rises above zero.

Moreover, cost functions associated with rate-based DNNs are based on the firing rates of the output neurons. In the context of classification, the chosen class is the output unit with the highest activity. One approach is therefore to treat spiking networks similarly and train the network so that the unit corresponding to the target class will have the highest activity.

It should be noted that this activity does not necessarily correspond to a neuron firing rate. Indeed, it is more likely to be the filtered, weighted sum over the spiking activities of the final layer of neurons in the network. Alternatively, the network can be trained so that the first neuron to spike will be the chosen class.

At the level of a single neuron, this distinction is no longer applicable. The activity of a neuron firing at a regular rate can be captured by its inter-spike interval, defined as the time between one spike and the next. Assuming the neuron is in a resting state, the inter-spike interval is equivalent to the time before the first spike of a neuron, plus the refractory period. The key design decision is how information is transmitted between neurons.

4.1.2 Memristive Frameworks

Modern deep learning algorithms are subjected to neuroscience-inspired restrictions by Spiking Neural Networks (SNNs) [337], which have shown notable increases in runtime efficiency [362]. Neuromorphic hardware has demonstrated considerable reductions in latency and energy consumption [387], by switching from full precision and fixed precision activations of artificial neuron models to temporally-encoded data representations collected by spiking neurons [414].

The considerable success of error backpropagation in training deep learning models has led to the development of numerous related training algorithms tailored for spiking neural networks (SNNs) [376]. These algorithms, which are guided by surrogate gradient descent, have been designed to address the non-differentiability of discrete spikes, which is a limitation of traditional gradient-based methods [262]. This proliferation of SNN usage is accompanied by the development of modular deep learning programming packages [114] that have optimised autodifferentiation for CUDA acceleration [274].

In parallel with these advances in training SNNs, the past decade has seen significant developments in brain-inspired devices, circuits, and architectures that integrate neuronal dynamics to enhance the hardware integration of SNNs and their constituent parts. Memristors and resistive RAM (RRAM) constitute a significant aspect of the exploratory research conducted

in the field of SNN implementation [49], as they serve as a natural conduit between SNN algorithms and accelerators [52]. They have been extensively utilized as both synapses and as spiking neurons.

At the ionic level, memristive synapses have been integrated into systems that naturally implement the spike-timing-dependent-plasticity (STDP) update rule using higher-order device dynamics [309], as evidenced by the literature [206]. An alternative application of ion-driven dynamics is the implementation of the memristor as a neuron, where nonlinear conductance evolution gives rise to abrupt switching that can be used to emit sudden voltage spikes [205].

This approach [16] is typically coupled with capacitive integration and has been referred to as a 'neuristor' [61], and a 'Memristive Integrate-and-Fire' (MIF) neuron [108, 156, 415]. Similarly, the leakage of ions through the membrane of biological neurons can be implemented using resistive dissipation [212], as in neuristors, as observed in nanowire networks [121, 416], or via the dynamic movement of ions in single devices [417].

At the architectural level, RRAM has been identified as a promising candidate for in-memory compute (IMC) architectures [199] due to its capacity to parallelise matrix-vector multiplication independently of time complexity when integrated as large-scale, modular arrays [76].

In contrast to the mapping of neurons, memristive synapses map neural network weights to device conductances. In general, RRAM IMC architectures are designed to be trained offline with weights mapped on-chip for inference and deployment [410]. Consequently, RRAM synapses should be stationary and only used for weight read-out. Higher-order dynamical behaviours of memristors are abstracted away and treated as non-idealities.

An additional challenge associated with RRAM-based IMC is the cost of communicating analog current signals along lengthy bit-lines and conversion into the digital domain. These issues have prompted the utilisation of binary activations in the form of spike-based IMC accelerators, which have been demonstrated to mitigate the challenges associated with mixed-signal computation by eliminating the necessity for extensive Analog-to-Digital Convertor (ADC) data conversion [78].

The majority of deep learning acceleration using memristors can be classified into one of the aforementioned categories: memristive neurons, memristive synapses that learn via

associative learning, and IMC accelerators. A limited number of designs have integrated memristive neurons and memristive synapses [367]. This is a praiseworthy achievement, as the intrinsic switching dynamics of memristive systems are leveraged to accomplish data-driven operations [336].

The consequence of allowing hardware to behave naturally is that a designer is no longer able to rely on synchronous, clock-driven processing and is susceptible to fault injections resulting from nonlinear ionic dynamics. Allowing the intrinsic dynamics of memristive hardware to 'teach itself' serves to exacerbate the challenges associated with training MSNNs.

This limitation has restricted the demonstration of MSNNs to unsupervised learning tasks that have been shown to solve simple, low-dimensional pattern recognition problems via local learning rules (typically STDP) and associative learning. Such tasks include the classification of a variety of characters and numbers, including a subset of the MNIST dataset.

A vast array of work has been conducted which integrates memristors with brain-inspired architectures [156]. This spans from low-level analogue action potential emulation to discrete spiking dynamics and non-spiking IMC processors [78]. The focus here is on prior work which uses nonlinear dynamics in memristive neurons together with memristive synapses, which also includes an associated demonstration of synaptic optimisation to achieve a data-driven outcome.

A fully memristive neural network (MSNN) is defined as an array that employs the nonlinear switching dynamics of memristors to trigger action potentials, with memristive weights utilized as neural network parameters. The 8×8 crossbar array presented [367] has been demonstrated to integrate a fully MSNN, including memristive synapses and neurons.

The synaptic array has been trained using unsupervised STDP to classify four letters in a 24-pixel grid. While the task achieved is considerably simple, the fully memristive experimental demonstration paves the way for the development of new training methods.

Another work employs the use of half-wave rectification [164], situated between crossbar arrays, to facilitate the processing of ReLU activation within the analog domain. Although not fully memristive nor a 'spiking' network, this approach offers a compelling illustration of the potential for successive analog activation transfer between RRAM crossbars, obviating

the need for intermediate data conversion.

This process bears resemblance to the transmission of analog action potentials between layers in biological systems. The training procedure employs gradient-based optimization, incorporating device non-idealities during the forward pass. This strategy has yielded a test set accuracy of 93.63% on the MNIST dataset.

In the referenced literatures, convolutional SNNs with memristors are employed [360], with both networks having undergone pretraining as non-spiking networks, which are then mapped or converted into the spiking domain [378]. Both networks exhibited satisfactory accuracy on the MNIST dataset; however, they did not demonstrate the capacity to process more complex, real-world data.

This discrepancy may be attributed to the significant disparities between the networks that underwent training and the MSNN that was implemented. A dense MSNN is adopted using a similar approach to that can be used here [67], and thus has minimal hardware requirements at run-time. The training process translates the switching dynamics of the memristive neuron into a firing rate, which may be the reason why a relatively low accuracy of 83.2% was achieved on the MNIST dataset.

The majority of these works present persuasive evidence utilising in-house fabricated arrays [253], either as standalone crossbars or as back-end-of-the-line (BEOL) integrated arrays with foundry-made chips. In contrast, the objective here is to utilise bespoke fabrication capabilities. Previously, memristors were employed solely in the forward pass, as their devices are not designed to be reprogrammed during inference. Consequently, their method does not necessitate switching to generate spiking dynamics.

Gradients can therefore be deterministically calculated partially off-chip. An alternative approach that harnesses memristive dynamics in the forward-pass computation in the network. Consequently, the MSNN approach can leverage the benefits of spike-based processing, such as sparse processing and lower data collision rates.

In order to facilitate and emulate the training process of memristive networks, a variety of valuable frameworks have been developed, each addressing specific niches within the field. These include MemTorch [183], NeuroSim [44], and the IBM Analog Hardware Acceleration Kit [292], which implement non-spiking networks that adopt mixed-signal bit-line

charge/current accumulation/summation processing.

In these simulators, memristive dynamics are accounted for during weight updates and otherwise fixed during inference. To complement these tools, NeuroPack [130] specifically targets the simulation of spiking networks, where memristive dynamics are also factored in during the weight update process and fixed during inference. Spiking dynamics are triggered by pulse-based input voltages.

In terms of hardware implementation, the conventional use of RRAM in circuits often necessitates a considerable amount of overhead to convert analogue currents into digital voltages, which in turn results in a significant power consumption [32]. In many instances, the power and area demands of the ADCs and digital-to-analogue converters (DACs) exceed the overhead brought on by RRAM, thereby negating the advantages of memristors. In contrast, spike-based approach eliminates the need for ADCs and DACs, thereby substantially reducing the cost of peripheral circuits.

4.1.3 Analogue Hardware Challenges

Novel computer hardware solutions that employ analogue devices still exhibit limited precision and unreliability. However, both physical and algorithmic techniques can be employed to mitigate these issues. In contrast to digital technology, the analogue approach inherently involves a degree of imprecision.

Analogue devices, such as RRAM, are susceptible to a number of issues, including stuck states, device-to-device variability and I-V nonlinearity. However, the development of advanced fabrication methods and circuit-level optimisations has enabled the mitigation of some of these non-idealities, while algorithmic techniques have also been shown to be effective in reducing their impact.

In contrast to the digital paradigm, where a multitude of physical imperfections are effectively concealed within a bit representation (either '1' or '0'), analogue electronics is confronted with significant challenges due to the intrinsic imprecision associated with non-discrete systems. Even with a minimal amount of non-idealities, it is challenging to encode information with perfect precision using an exact conductance value.

However, non-idealities do exist and can result in significant deviations from ideal behaviour. These include the device becoming stuck in certain conductance states, undergoing changes

in conductance over time, showing non-linear current-voltage characteristics, or displaying non-linear conductance modulation in response to voltage stimuli.

It could be argued that analogue computing's more fundamental challenge lies in its reduced precision compared to digital computing, especially when digital systems utilise 16 or more bits of representation. While these issues may be grounds for disqualification in many applications, this may not be the case for machine learning applications, which often employ reduced precision computing, even within digital systems.

In general, machine learning models demonstrate a degree of robustness to minor alterations, such as the presence of noise [47]. In the event of significant deviations from ideal conditions, hardware imperfections may result in a decline in accuracy. However, this does not necessarily render the system inoperable. It is, therefore, crucial to comprehend the impact of non-idealities and to ascertain how they can be effectively mitigated.

In the context of linear algebra applications, a proportional relationship between voltage and current (i.e. Ohmic behaviour) is the preferred option. This is due to the fact that Ohm's law is employed in the implementation of multiplication, as previously discussed. Nevertheless, exceptions to this linear relationship do arise, particularly in the case of high-resistance devices [242].

A number of approaches exist at the device and circuit level that facilitate the resolution or even circumvention of the issue of nonlinearity. During the fabrication of RRAM devices, the adoption of a hot-forming step can result in the generation of more linear characteristics [330].

In the case of individual device programming, the adoption of a transistor-to-resistor ratio (1T1R) architecture can facilitate the precise tuning of memristor conductance, despite the presence of any I-V nonlinearities [196]. Alternatively, a charge-based accumulation approach can be employed, wherein a constant voltage is applied, but the input is encoded into pulse width [7]. This eliminates the dependence on the shape of the I-V curve.

It is possible that some memristive devices may become fixed in a specific conductance state. This phenomenon has been observed following processes such as electroforming, as well as after several successful programming cycles [149]. In general, the greater the discrepancy between the intended and actual conductance, the greater the potential for adverse effects. Therefore, it is of paramount importance to identify methods for the prevention or mitigation

of faulty devices.

The overall effect of a device becoming stuck is contingent upon the behaviour of other devices, and thus this phenomenon can be employed to mitigate the negative effects. To illustrate, if a device becomes stuck, its negative effect may be counteracted by adjusting the conductance of another device in the differential pair [207]. On occasion, such an adjustment may occur accidentally, whereby both devices in a differential pair become stuck simultaneously.

As an alternative, if faulty devices can be identified prior to programming, more sophisticated mapping strategies can be employed. The most significant weights can be mapped onto crossbar rows and columns with the lowest incidence of stuck devices [93]. The most significant terms refer to the weights that could have the greatest impact on accuracy. One way to identify such weights is to calculate of sensitivity $\Delta w_{i,j} := -\eta \frac{\partial E}{\partial \Delta w_{i,j}}$, for each weight $w_{i,j}$, where E is the back-propagated loss at the current neuron and η is the learning rate.

The term 'limited dynamic range nonideality' is used to describe a situation whereby the $\frac{G_{on}}{G_{off}}$ ratio is relatively small, which can ultimately result in a reduction in effective precision. In the context of other non-idealities, such as device variability, limited dynamic range can result in a reduction in the number of distinguishable states that are available.

If each state is associated with a certain amount of absolute variability, it is evident that a larger dynamic range is preferable, as it allows for a more effective differentiation between those states that are less distinct. The impact of the dynamic range is highly dependent on the specific application. Should one desire to utilise analogue arrays for the storage of digital information, an enhanced dynamic range will facilitate a greater precision in the number of equivalent bits.

Nevertheless, when considering the acceleration of linear algebra operations (and, by extension, machine learning), such comparisons cannot be made with the same degree of ease. As these hardware accelerators are based on analogue computation, the concept of 'bits' – although potentially useful – does not apply directly. In analogue contexts, an error is defined as any deviation from the intended value. The magnitude of the error is the key factor in determining the severity of the mistake.

In the context of inference applications, a large dynamic range is not a crucial factor. If a naive mapping scheme is employed whereby the value of a weight is represented using a single conductance value, then the inaccuracies produced by this imperfect mapping can be addressed with a $\frac{G_{on}}{G_{off}}$ ratio of as low as 3 [241]. In other contexts, the impact of limited dynamic range cannot be evaluated without first understanding the nature of other non-idealities, namely the deviations they cause.

Line resistance represents a non-ideality that arises from the presence of non-zero interconnect resistances in crossbar arrays. In the event of its presence, this results in discrepancies from the ideal computation of vector-matrix products. While the impact on accuracy can be significant, there are both physical and algorithmic techniques that can be employed to mitigate it.

One of the most straightforward methods for mitigating the impact of line resistance is to enhance the ratio between the resistance of the devices and the resistance of the interconnecting wires. As resistance is inversely proportional to the cross-sectional area of the wire, one method of reducing interconnect resistance is to increase the width of the wires [195].

However, this can be challenging in dense arrays, and an alternative approach is to use more conductive materials, for example, $2nm$ platinum nanofins [280]. Another approach is to increase the resistance of the crossbar devices, although this can sometimes result in less stable device behaviour.

At the circuit level, a variety of techniques may be employed which utilise the systematic properties of line resistance in different ways. For instance, a technique designated as double biasing can facilitate a more symmetrical distribution of electric potentials within crossbar arrays, thereby attenuating the impact of line resistance effects [126]. As the size of the crossbar array increases, voltage drops tend to accumulate.

Therefore, splitting up the array into smaller units [388], or even organising them in three-dimensional structures [391] can help to mitigate this issue. In considering the specific applications for which crossbar arrays are to be employed, algorithms may be deployed to ascertain optimal mappings from software parameters to physical quantities, such as voltage and conductance. A nonlinear mapping from weights to conductances can be employed to counteract the detrimental effects of line resistance.

Alternatively, sensitivity analysis can identify the weights that are most sensitive, and thus map them closest to the applied voltages, where their contribution would be disturbed the least [3]. In the specific context of supervised learning, input intensities may be predicted, and the inputs with the highest expected intensity (as well as the corresponding weights) can be mapped closest to the outputs in order to minimise the negative effects of line resistance.

When training networks directly on crossbar arrays, i.e. in situ, linear adjustments of conductance are the preferred approach [31]. In order to ensure a linear response, the system must be modified physically. Some previous studies have proposed adjusting the device structure [380], typically by introducing additional layers [382]. An alternative approach is combining memristive devices with CMOS transistors, which help to improve the linearity [6].

Random telegraph noise (RTN) is defined as the occurrence of unpredictable switching between two or more discrete voltage levels in electronic devices [286]. This phenomenon is frequently observed in memristors. RTN is more commonly experienced in devices with higher resistance, which can impede the use of such devices for reducing power consumption or mitigating the effects of line resistance. To circumvent RTN or at least mitigate its effects, it is necessary to modify the fabrication process. For instance, some studies have demonstrated that non-filamentary devices can assist in reducing this type of noise [36].

Once the specific application where memristive crossbars will be employed is identified—for instance, classification using neural networks, a pertinent metric such as accuracy, may be optimised instead of attempting to address individual non-idealities. This methodology is more technology-agnostic, as the nature of non-idealities frequently differs between technologies. However, approaches that optimise the metrics pertinent to the application tend to be algorithmic and, thus, more readily transferable.

In the field of machine learning, averaging approaches have the potential to enhance the accuracy and robustness of models, particularly in situations where the memristive implementation is susceptible to non-idealities. One strategy is to utilise multiple networks in parallel and compute their average outputs. Additionally, stability over time can be a crucial consideration, as certain non-idealities, such as RTN, are stochastic in nature. By averaging over time, the effects of these non-idealities can be mitigated [358].

The statistical approaches employed in modern machine learning are based on the minimisation of deviations from ideal behaviour in the training data. This can be extended to

incorporate the non-ideal effects of the hardware on which the model will be implemented. In some instances, this can be achieved by introducing non-ideality-agnostic noise during training in order to enhance the robustness of the networks [396]. Alternatively, noise can be designed to reflect the nature of the non-idealities, thereby enabling the model to adapt more effectively to the various shortcomings of the hardware [131].

4.2 Methodology

4.2.1 Learning Rules

Synapses are capable of undergoing changes in their structure and function, a process known as synaptic plasticity. In neuronal systems, the strength of synapses undergoes changes in accordance with the occurrence of spikes in presynaptic or postsynaptic neurons, a process known as synaptic plasticity. Indeed, memory can be conceptualised as a vast neural network. In other words, the synaptic weight is a fundamental determinant of learning and memory processes.

Two principal forms of synaptic plasticity have been identified: long-term plasticity (LTP) [20] and short-term plasticity (STP) [419]. Synapses may undergo strengthening or weakening, and may also exhibit memory retention over a relatively long time, which is referred to as Long-Term Facilitation (LTF) or Long-Term Depression (LTD), respectively. If the change occurs within a relatively short time, it is referred to as Short-Term Facilitation (STF) or Short-Term Depression (STD).

The concept of synaptic long-term potentiation (LTP) has already been incorporated into the training process of deep neural networks (DNNs). This involves the concatenation of all synapse weights into a large multi-dimensional matrix, enabling the identification of the optimal weight matrix through error backpropagation. However, the mechanisms and learning rules in neuroscience are not identical.

One of the most celebrated learning rules in neuroscience is the Hebbian rule [116]. The most concise summary of this rule is: neurons that 'fire together, wire together' [312]. The Hebbian rule can be interpreted as a rate model defined by the neuron spiking rate. It is a local rule, and it requires neurons to be simultaneously active [95]. The general model for

this local rule can be defined as follows:

$$\frac{dw_{ij}}{dt} = F\left(w_{ij}, M, v_j^{prev}, v_i^{post}\right) \tag{4.1}$$

Where w_{ij} is the synaptic weight, M is the effect of the neuromodulator, v_j^{prev} is the presynaptic neuron firing rate, and v_i^{post} is the postsynaptic neuron firing rate. A Taylor expansion of equation 4.1 with respect to the rate is:

$$\frac{dw_{ij}}{dt} = a_0(w_{ij}, M) + a_1(w_{ij}, M)^{prev} v_j^{prev} + a_1(w_{ij}, M)^{post} v_i^{post} + a_2(w_{ij}, M)^{corr} v_j^{prev} v_i^{post} + ... \tag{4.2}$$

In the absence of a spike at either the presynaptic or postsynaptic neuron, the effect is represented by a_0. When spikes occur solely at the presynaptic neuron, a_1^{prev} is the expansion coefficient. Similarly, when spikes occur exclusively at the postsynaptic neuron, a_1^{post} is the expansion coefficient. Finally, when spikes occur at both the presynaptic and postsynaptic neurons, a_2^{corr} is the expansion coefficient.

There are additional terms of higher orders, v_j^{prev} and v_i^{post} , which are represented by ellipsis. The coefficients are contingent upon the parameters w_{ij} and M. In accordance with varying parameters and conditions, the Hebbian rule can manifest as LTF or LTD. It is noteworthy that the Hebbian rule constitutes a set of learning rules, rather than a singular, fixed rule.

Another prevalent learning rule is spike-timing-dependent plasticity (STDP). The STDP process entails an increase or decrease in synaptic weight contingent on the time interval between pre- and postsynaptic spikes. The total weight change from neuron j to neuron i is defined as follows:

$$\Delta w_{ij} = \sum_n \sum_f W\left(t_i^n - t_j^f\right) \tag{4.3}$$

In this context, the term t_i^n denotes the spike times of postsynaptic neuron i, while t_j^f indicates the spike time of presynaptic neuron j. The variables n and f are used to count the pre- and postsynaptic spikes, respectively. The term *W(x)* refers to the learning window of the STDP function. It should be noted that numerous variations of STDP exist, with one of the most

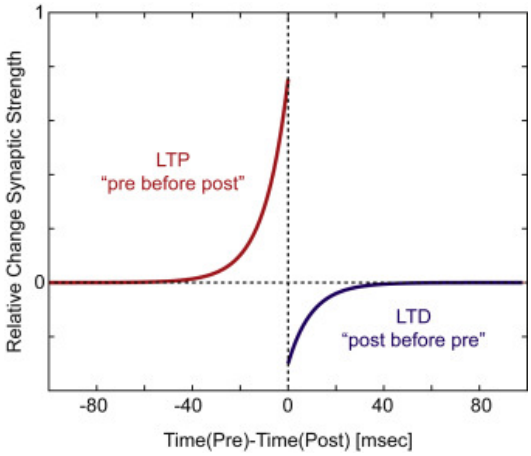

Fig. 4.1 Depiction of spike-timing-dependent plasticity (STDP). If the presynaptic spike occurs before the postsynaptic spike ("pre before post"), the synapse is strengthened (red, LTP, long-term potentiation). If the postsynaptic spike occurs before the presynaptic spike, the synapses are weakened (blue, LTD, long-term depression). Typically, two action potentials need to occur within at most a few tens of milliseconds for STDP to be recruited. [88].

common types being the pair-based variant. This is defined as follows:

$$W_+(x) = \left\{ A_+(w) e^{-|\Delta t|/\tau_+} \right\} \quad \forall t_{post} \in t_{prev} < t_{post} \tag{4.4}$$

$$W_-(x) = \left\{ A_-(w) e^{-|\Delta t|/\tau_-} \right\} \quad \forall t_{prev} \in t_{post} < t_{prev} \tag{4.5}$$

Where $|\Delta t| = \left| t_{post} - t_{prev} \right|$, the time of the postsynaptic spike is designated as t_{post}, while the time of the presynaptic spike is designated as t_{prev}. In most cases, $A_+(w)$ is positive, $A_-(w)$ is negative, and these values may be dependent on the current synaptic weight. $W_+(x)$ is associated with long-term facilitation (LTF), while $W_-(x)$ is associated with long-term depression (LTD).

By introducing $S_j = \sum_f \delta\left(t - t_j^f\right)$ and $S_i = \sum_n \delta\left(t - t_i^n\right)$, where S_j represents the spike train of the presynaptic neuron and S_i represents the spike train of the postsynaptic neuron. The pair-based STDP rule, as presented in previous equations, can be implemented by:

$$\frac{dx_j}{dt} = \sum_f \delta\left(t - t_j^f\right) - \frac{x_j}{\tau_+} \tag{4.6}$$

$$\frac{dy_i}{dt} = \sum_n \delta\left(t - t_i^n\right) - \frac{y_j}{\tau_-} \tag{4.7}$$

In this context, the notation t_j^f represents the spike time of the presynaptic neuron, while t_i^n denotes the spike times of the postsynaptic neuron. The variables x_j and y_i can be interpreted as a trace that each pre- and postsynaptic spike leaves, and respectively corresponds to the $e^{-|\Delta t|/\tau_+}$ and $e^{-|\Delta t|/\tau_-}$ terms to give:

$$\frac{dw_{ij}}{dt} = A_+ w_{ij} x_i(t) \sum_n \delta\left(t - t_i^n\right) + A_- w_{ij} y_i(t) \sum_f \delta\left(t - t_j^f\right) \tag{4.8}$$

Where the first term on the right-hand side denotes the pre-before-post effect, while the second term indicates the post-before-pre effect. It is noteworthy that for independent Poisson inputs, STDP models are related to rate models [95]. One may define it as a rate model as follows:

$$\frac{dw_{ij}}{dt} = \int_0^{+\infty} W(-s)\varepsilon(s)ds \cdot v_j^{prev} + \int_{-\infty}^{+\infty} W(s)ds \cdot v_j^{prev} v_i^{post} \tag{4.9}$$

The first term on the right-hand side is defined by the integral over the 'causal' part of the learning window, also known as the 'pre-before-post' relation. This integral is represented by the function $\int_0^{+\infty} W(-s)\varepsilon(s)ds$, where W is the weighting function and $\varepsilon(s)$ describes the time course of a Postsynaptic Potential (PSP) for $s > 0$. A comparison of Equation 4.2 with Equation 4.9 reveals that there are two terms defining coefficients $a_1(w_{ij}, M)^{prev}$ and $a_2(wij, M)^{corr}$, while the remaining terms are all zero.

Non-volatile memristors are capable of operating as long-term potentiation (LTP) synapses, as the memristance will not undergo alteration following each update. In a fully connected neural network, the number of synapses between two layers is given by the equation $m \cdot n$, where m is the number of neurons in the previous layer and n is the number of neurons in the next layer. Consequently, non-volatile memristors have been employed in crossbar arrays for vector-matrix multiplication, utilising Kirchhoff's current law as illustrated in Figure 1.7.

Gradient backpropagation and STDP are two distinct mechanisms for learning. Gradient backpropagation is a widely adopted technique in the domain of deep neural networks (DNNs), whereas STDP is a prevalent approach in the field of spiking neural networks (SNNs). Both gradient backpropagation and STDP can be implemented using memristors. For gradient backpropagation [110], memristive crossbar arrays can be employed for both training and inference in neural networks [8].

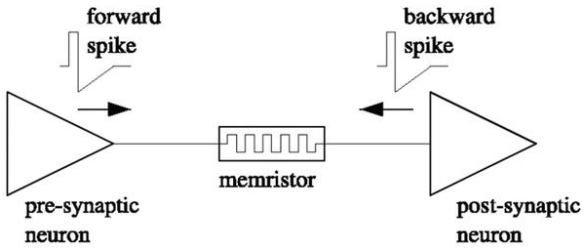

Fig. 4.2 Memristor between presynaptic and postsynaptic neurons [128].

The training process necessitates the utilisation of an external control unit, which introduces a greater degree of overhead than using the memristive crossbar alone. Furthermore, the STDP rule described above can be achieved by utilising memristors [225], and can be verified using SPICE models [393].

As illustrated in Figure 4.2, a memristor is situated between two neurons. The presynaptic and postsynaptic spikes will generate a voltage difference across the memristor, which will then cause a memristance update. Furthermore, the time interval between the presynaptic and postsynaptic spikes will result in varying changes in memristance.

The majority of research into the emulation of plastic synapses relies on non-volatile memristors for long-term potentiation (LTP), with few researchers focusing on the use of volatile memristors to mimic short-term potentiation (STP) synapses. The significance of volatile memristors is further underscored by their role in neural information processing, including functions such as motion detection, speech recognition, and working memory [97].

In contrast to long-term potentiation (LTP), short-term potentiation (STP) is dependent on the spiking activity of the presynaptic neuron. Let define the fraction $P_{release}$ as the amount of neurotransmitter released by the presynaptic neuron. It can then be shown that the synaptic weight is directly related to $P_{release}$, and that both STF and STD can be modelled with the dynamics of $P_{release}$. STF can be defined as:

$$\frac{dP_{release}}{dt} = \frac{P_{release} - P_0}{\tau_F} + f_F(1 - P_{release}) \sum_f \delta\left(t - t^f\right) \tag{4.10}$$

Where t^f is the time for each presynaptic spike, P_0 is the resting value of $P_{release}$, τ_F is a time constant governing the recovery process, and f_F controls the facilitation degree. Similarly,

STD can be defined as the following with D denotes for depression:

$$\frac{dP_{release}}{dt} = \frac{P_{release} - P_0}{\tau_D} + f_D(1 - P_{release}) \sum_f \delta\left(t - t^f\right) \tag{4.11}$$

The transient nature of volatile memristors allows them to emulate the STF or STD, as the memristance change is only retained for a brief period. The measurement of STF and STD is typically conducted through the use of Paired-Pulse Facilitation (PPF) and Paired-Pulse Depression (PPD). The PPF and PPD index is defined as $\frac{A_2}{A_1}$, where A_1 and A_2 are the absolute amplitudes of the EPSC or IPSC resulting from two successive presynaptic spikes.

This index is used to evaluate the strength of PPF or PPD. In 2018, a novel type of volatile memristor was proposed, and the PPF and PPD indexes were subsequently measured [328]. This is a two-terminal single-layered molybdenum disulfide (MoS_2) device, wherein the conductance change is achieved through Joule heating.

An alternative approach is to train the SNN using the supervised SRDP learning rule. This is more hardware-compatible than the backpropagation algorithm (BP) and overcomes the low accuracy of the unsupervised SRDP learning rule, which only considers local optimisation and ignores global errors [279]. The supervised SRDP rule can be described as follows:

$$\Delta_{ij} = \begin{cases} +1 & \text{if } f_{r_j} < f_{t_j} \ \& \ f_{s_i} > \text{ threshold} \\ -1 & \text{if } f_{r_j} < f_{t_j} \ \& \ f_{s_i} < \text{ threshold} \\ -2 & \text{if } f_{r_j} > f_{t_j} \ \& \ f_{s_i} > \text{ threshold} \end{cases} \tag{4.12}$$

In this context, f_s and f_r represent the output frequencies of the sensory and relay neurons, respectively. f_t denotes the teaching frequency, while threshold refers to the threshold of the output frequencies of the sensory neurons. The symbol Δ_{ij} indicates the pulses applied to the positive or negative synapses connecting the i_{th} sensory neuron and the j_{th} relay neuron.

When Δ_{ij} is equal to 1, a pulse is applied on the negative synapse to increase the synaptic weight, whereas when Δ_{ij} is equal to -1, a pulse is applied on the positive synapse to decrease the synaptic weight. In the event that the output frequency of a relay neuron is less than the teaching frequency and the output frequency of the sensory neuron is greater than the threshold, the neuron will learn the input pattern by increasing the synaptic weight $\Delta_{ij} = +1$. Conversely, if the output frequency of the sensory neuron is less than the threshold, the

synaptic weight will be decreased $\Delta_{ij} = -1$.

In the event that the output frequency of the relay neuron is higher than the teaching frequency, this indicates that the neuron has either learned a feature belonging to an alternative class ($f_t = 0$) or has learned an excessive number of features ($f_t > 0$). In such instances, the neuron should erase these features by decreasing the synaptic weight $\Delta_{ij} = -2$.

In order to quantify the training effort, a loss function has been devised based on the discrepancy between the output frequencies of the relay neurons and the teaching frequencies. This can be expressed as follows:

$$l = \frac{\sum_j \left(f_{r_j} - f_{t_j}\right)^2}{\sum_j \left(f_{t_j}\right)^2} \tag{4.13}$$

$$Loss = \frac{\sum_i^n l_i}{n} \tag{4.14}$$

In this context, the variable l represents the error incurred when a single sample is input into the testing set. The term *Loss* denotes the total error after all samples in the testing set have been input. The variable f_{r_j} signifies the output frequency of the j_{th} relay neuron when the input signal originates from the synapses. The variable f_{t_j} represents the teaching frequency generated by the j_{th} relay neuron when the input signal is a teaching signal. The program terminates when the *Loss* value is sufficiently low.

4.2.2 Training Schemes

The discovery of spike-timing-dependent plasticity (STDP) mechanisms and the emergence of nanoscale non-volatile memory (NVM) devices have opened a new avenue towards the realisation of brain-inspired computing over the past decade [405]. Prior research suggests that STDP can be used to train spiking neural networks (SNNs) with resistive random-access memory (RRAM) synapses in-situ, without trading off their parallelism [289]. Furthermore, these devices have demonstrated low energy consumption for state transitions and a highly compact layout footprint [401].

A neuromorphic system-on-a-chip (NeuSoC) architecture has been proposed, comprising of multiple-layer fully connected SNN [163]. In this model, the input layer encodes real-valued inputs into spatiotemporal spike patterns, which are then processed by the subsequent layers using STDP-based unsupervised or semi-supervised learning. Neurons in each layer are

connected to the higher layers via synapses that hold the 'weights' of the SNN.

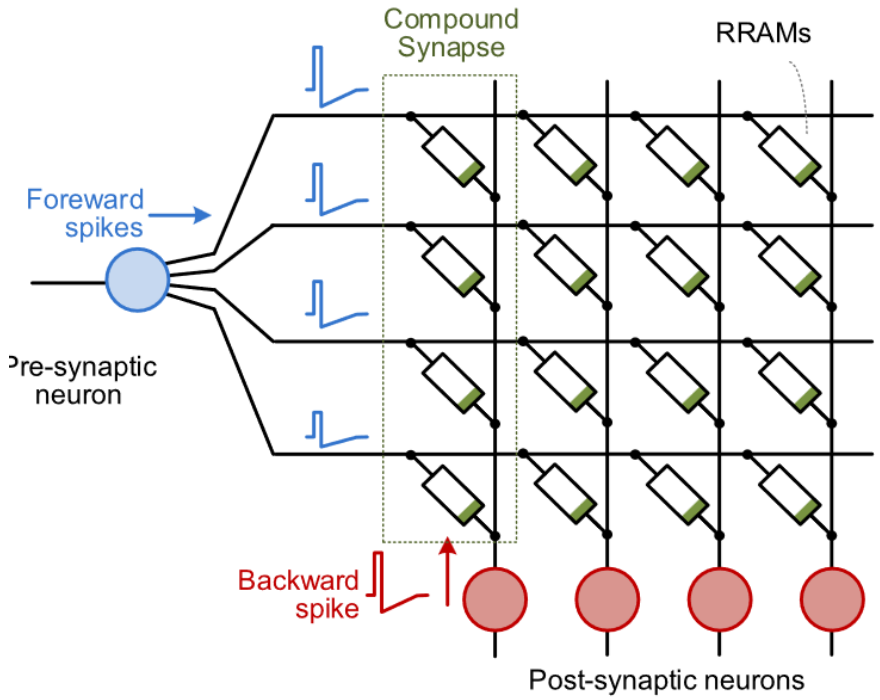

Fig. 4.3 A single layer of spiking neural network with RRAM synapses organized in crossbar architecture. Post-synaptic neuron connects with a pre-synaptic neuron with several RRAM synapses in parallel. Back-propagated spikes after dendritic processing modulate RRAM in-situ under STDP learning rule [385].

As illustrated in Figure 4.3, a RRAM crossbar array is employed to establish synaptic connections between the two neuronal layers. The spiking neurons in the second and subsequent layers implement competitive learning through a winner-take-all shared bus mechanism, whereby the neuron that spikes first for an input pattern inhibits the remaining neurons in the same layer [228].

A combination of STDP and WTA-based learning rules enables the synapses to be locally updated through the interaction of pre- and post-synaptic neuron spikes in Figure 4.2, thereby facilitating network learning in the form of fine-grained weight (or conductance) adaptation in the synapses.

The presented architectural configuration bears resemblance to existing memory architectures, wherein the devices are arranged in a dense two-dimensional array with the input and

output neurons constituting the peripheral circuitry, laid out at a matching pitch with the memory array. This layout forms a repeatable motif that can be scaled to deeper SNN architectures.

The extension of larger 3D IC architectures using through-silicon-via (TSVs) represents a natural pathway for further scaling to very high integration density and network complexity. This can be achieved without resorting to the overhead incurred by asynchronous communication protocols such as the address-event-representation (AER) [145].

It is desirable to have non-volatile analog-like weights in order to achieve effective STDP learning [91]. However, the majority of practical [319], small-sized RRAM devices exhibit abrupt switching behaviour [400], which consequently limits the stable synaptic resolution to 1-bit (or binary, bistable) [331].

Furthermore, the switching probability and switching times of these devices typically depend upon the voltage applied across the device, as well as the duration of the voltage pulse [198]. In order to circumvent the binary resolution of these devices, a compound memristive synapse with multiple bistable devices in parallel was previously proposed as a means of emulating analog weights on average [24].

Theoretical studies have demonstrated that the STDP learning rule facilitates unsupervised local learning in spiking neural networks through the implementation of a Bayesian expectation-maximisation algorithm [263] and hidden Markov models [157]. Furthermore, a nonlinear STDP learning function, such as an exponentially shaped window observed in biological neural systems [341], is essential for ensuring the stability and efficiency of the computing process [320].

As previously stated, emerging memristive nanoscale devices are being considered as a means of enabling the realisation of large-scale neuromorphic hardware. Nanoscale implementations of memristors [176], including phase-change memory (PCM) [155], resistive random-access memory (RRAM) [221], and spin-torque-transfer random-access memory (STT-RAM) [306], have been demonstrated to exhibit switching characteristics analogous to those observed in spike-timing-dependent plasticity (STDP) [202].

Furthermore, these devices enable the highly desired advantage of a small silicon area of $4F^2$ (F is the feature size of the semiconductor fabrication process) [40], ultra-energy-efficient operation of sub-pico-Joule per switching event, CMOS compatibility, and dense crossbar

(or crosspoint) arrays and 3D integration.

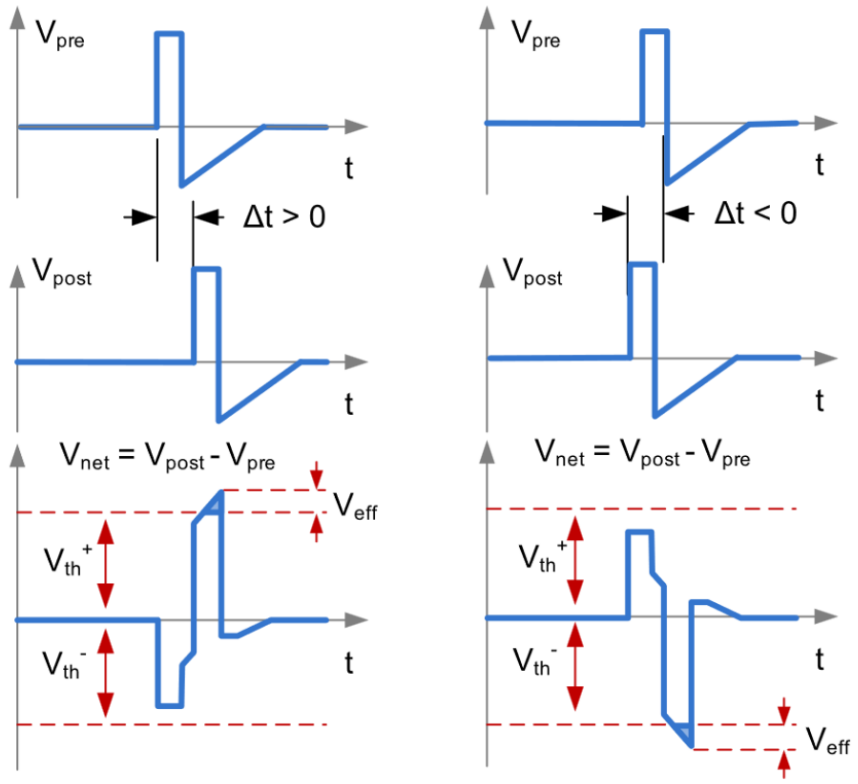

Fig. 4.4 A pair of spikes are applied across a synapse to create relative-timing dependent net potential V_{net}, of which the over-threshold portion V_{eff} may cause the RRAM resistance to switch [384].

In particular, RRAM devices exhibit characteristics that are analogous to those of biological synapses. Furthermore, RRAMs exhibit conductance levels comparable to synaptic strength (or weight), and their conductance can be modified by voltage pulses. Notably, RRAMs are capable of directly implementing spike-timing-dependent plasticity (STDP) through identical pair-wise spikes.

As illustrated in Figure 4.4, the net potential V_{net} created by a pre-synaptic spike and a late arriving post-synaptic spike, with $t > 0$, produces an over-threshold, V_{th}^{+}, portion V_{eff}, which then causes an increase in conductance in a typical RRAM. Conversely, a presynaptic spike and an earlier arriving post-synaptic spike with $t < 0$ produces V_{eff} that crosses the negative threshold, V_{th}^{-}, and thus causes a decrease in the conductance.

A memristor with analogue resistance is an optimal choice for STDP learning. However, experimental studies indicate that the majority of nanoscale implementations of memristive RRAMs exhibit a stochastic process in their filament formation, as well as an abrupt conductance change once the filament is formed [285]. The results demonstrate an abrupt transition from the high resistance state (HRS) to the low resistance state (LRS), as well as notable variations in the switching threshold voltages and HRS/LRS transitions, which tend to be stochastic in nature.

The intrinsic stochastic switching in RRAM therefore limits the stable synaptic resolution to 1 bit, or bistable behaviour. In order to unlock STDP-based learning in SNN hardware, a solution enabling non-linear, especially exponential, STDP learning functions with binary RRAMs is therefore highly desirable.

One proposed approach to enable the complexity of learnable tasks to be scaled up in fully MSNNs by directly applying gradient descent to the nonlinear state evolution of memristive neurons and synapses [413]. The modelling of both neurons and synapses in biological neural networks employs the use of memristors. The MIF neuron model has been devised with the objective of achieving distinct depolarisation, hyperpolarisation and repolarisation voltage phases with a minimal set of circuit elements.

Memristive synapses serve as interconnects between layers of neurons. This allows for the development of dynamical, time-varying memristive neurons, which are capable of learning and achieving significantly enhanced accuracy on data-driven tasks compared to previous reports on MSNNs [262]. By leveraging the analog spiking characteristics inherent to the MIF neuron model, the non-differentiability of spike-based activations is entirely circumvented.

In order to account for the hardware implementation of a fully MSNN, it is necessary to consider the voltage response of the MIF neuron, which must drive the memristive synapses. The synaptic conductances are correspondingly weighted by this response. This can be fully integrated into a crossbar according to the following equation:

$$\begin{array}{cccc} \multicolumn{1}{c}{\mathbf{G}} & & \mathbf{V} & = & \mathbf{I} \\ \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,m} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{bmatrix} & \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{bmatrix} & = & \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_m \end{bmatrix} \end{array} \tag{4.15}$$

where $\boldsymbol{I}$ is the output current vector, $\boldsymbol{V}$ is the input voltage vector, and $\boldsymbol{G}$ is the conductance matrix of the crossbar.

$$I_n = \sum_{i}^{m} v_i \times w_{n,i} \tag{4.16}$$

The output current vector is generated via bitline current summation, as illustrated in equation (4.16), and is directly driven into the input of the subsequent MIF neuron layer. Resistive loading may result in the attenuation of the output current in subsequent stages [364]; however, this can be accounted for through the utilisation of a scaling factor or the implementation of a buffer [363].

$$\tau_{syn}\frac{dI}{dt} = a - I \tag{4.17}$$

$$\tau_{syn}\frac{da}{dt} = W_i \cdot \sum_{n} \delta\left(t - t_i^n\right) \tag{4.18}$$

A considerable number of neural coding studies portray spike trains as a superposition of time-shifted Dirac delta pulses, represented by $\sum_n \delta\left(t - t_i^n\right)$. Consequently, this model of spikes is one idealisation. The spike train is employed to modulate a time-continuous alpha input current I, which has been modelled by the differential equation (4.17) presented in order to ensure compatibility with real, physical systems.

In this series of equations (4.18), a represents an internal state variable, τ_{syn} denotes a time constant that determines the shape of the alpha current, and W_i is the synaptic weight between a presynaptic neuron i and its associated postsynaptic neuron. It is widely accepted that such alpha waveforms correspond to the response from biological neurons in the sensory periphery that respond with graded potentials [75].

$$I^{t+1} = \frac{a^t - I^t}{\tau_{syn}} + I^t \tag{4.19}$$

$$a^{t+1} = a^t - \frac{a^t + \sum_n \delta\left(t - t_i^n\right)}{\tau_{syn}} \tag{4.20}$$

In order to train a network of MIF neurons and synapses using gradient descent, the differential equations representing the MIF circuit dynamics are recast into discrete-time form (4.19, 4.20). This allows the memristive dynamics to be captured in a computational graph that evolves over time, in a manner analogous to that of a recurrent neural network (RNN).

In practice, SPICE-like simulators employ a variety of differential equation solvers, including the backward Euler method and the fourth-order Runge-Kutta method (RK4). In order to ensure compatibility with the BPTT algorithm, the differential equations are solved using the forward Euler method, which provides an explicit representation of the next time step based on present-time dynamics.

The intricate dynamics of the MIF neuronal network are now accounted for in the MSNN, which is unrolled in time in such a way that gradient descent can be used to optimise the memristive synapses as a function of the MIF evolution. The discrete-time solution can be illustrated as a directed, acyclic graph, where time flows in one direction.

In order to train a network, a loss function is calculated using the membrane potential v of the output layer at each step. It should be noted that the adjoint method [45] is normally not adopted, as an intermediate state is required in order to calculate the loss and guide the training process for each time step. The focus is on the spiking output at all time steps, rather than just on the final state of the system, which makes BPTT a more optimal choice.

The predicted MIF neuron is expected to spike most frequently by aiming to increase the membrane potential across time steps, while the incorrect target should be suppressed. Given that the membrane dynamics are continuous, it is possible to train a fully analogue MSNN, as has become the norm in the training of deep SNNs via error backpropagation [262].

The BPTT algorithm employs an iterative application of the chain rule from the output back to the leaf nodes in order to determine the optimal update direction for the network [79], with previous demonstrations have treated w as the device conductance. This approach offers a more biorealistic representation and can be fully integrated using RRAM crossbars.

4.2.3 Conductance Mapping

The approaches described in the preceding sections pertained to memristive neural networks (MNNs) that had been trained through conventional methodologies without consideration of the non-idealities. The primary challenge lies in the discrepancy between the behaviour of conventional SNNs and that of MNNs.

The synapses of the former typically exhibit a consistent response and typically relate inputs to outputs in a linear fashion. In contrast, MNNs utilise synapses implemented using memristors, which can become fixed in a particular state, exhibit varying responses over time,

due retention or random telegraph noise (RTN), and even produce nonlinear outputs, I-V nonlinearity.

A substantial proportion of the literature on addressing memristor nonidealities focuses on modifications at the hardware level. These may entail: Modifications to the device structure are a common approach, as evidenced by the literature [82]; additional circuitry is another avenue of exploration [6, 196]; programming and mapping schemes are also subject to modification [389]; in-situ retraining is a further strategy, as exemplified by the literature [42, 140, 366, 194].

However, these techniques may have some drawbacks. For example, pulse-width modulation [7], may minimise the effects of I-V nonlinearities; however, this comes at the cost of increased clock cycles [32]. Furthermore, hardware-level changes are often technology-specific, thus rendering them difficult to apply to a wide range of device types.

As an alternative, techniques that do not interfere with the hardware, such as committee machines (CMs), may be employed. These allow the performance of MNNs to be improved by combining them and require only the average of the outputs of crossbar arrays [148]. Additionally, modifications to ex situ training have been previously proposed.

Common approaches include adjusting the loss function [418] or introducing noise into the synaptic weights [115] or conductances that implement those synaptic weights [151], thereby enhancing the resilience of MNNs to the effects of non-idealities. It is evident that ex situ training represents a promising methodology for enhancing the feasibility of MNNs.

However, previous approaches have only considered a limited number of non-idealities. For example, injection of noise into the conductances is insufficient in many situations, as the effect of non-idealities such as I-V nonlinearity cannot be represented by the disturbance of the weights alone. Furthermore, conventional weight implementations do not map naturally to resistive crossbar arrays, while training validation schemes are not well suited to the often stochastic nature of MNNs.

This research focuses on enhancing the efficacy of *ex-situ* training of memristive SNNs, while also evaluating the resilience of this approach. To this end, data from a SiO_x memristive device is employed, complemented by insights drawn from existing literature.

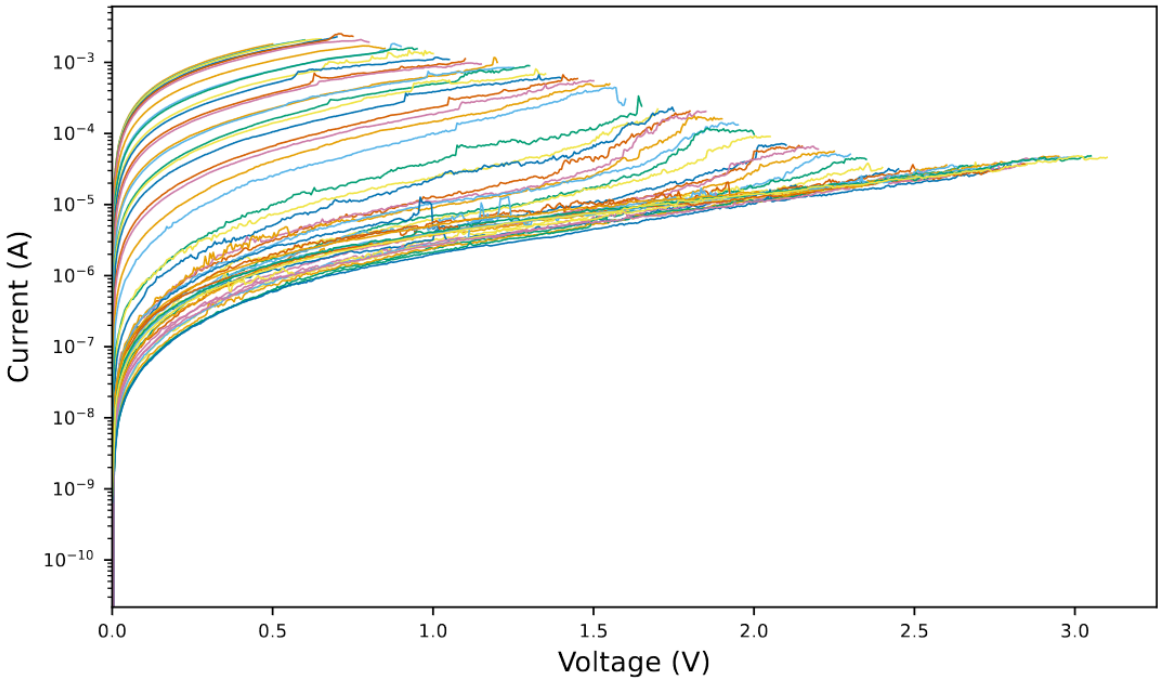

Fig. 4.5 Single sweeps of 53 resistance states of a SiO_x device. Different states were obtained by incrementing maximum voltage by 0.05V. Every seventh state shares the same colour; this does not indicate any other relationship between such states [18].

The SiO_x resistive random-access memory (RRAM) device comprised a $1\mu mSi/SiO_2$ switching layer positioned between a 100nm Mo bottom electrode and a 100nm Au top electrode. Furthermore, a 5nm Ti wetting layer was incorporated between the SiO_x and Au electrodes. Following electroforming and additional validation procedures, positive sweeps were conducted on the SiO_x device, commencing from 0.5V and increasing by 0.05V in each run, thereby generating resistance states that spanned multiple orders of magnitude.

The majority of existing literature on ex-situ training of MNNs assumes a linear relationship between inputs and outputs at the level of individual devices. Non-linearities are only introduced at the level of the activation functions. In particular, outputs are assumed to be a function of the product of a vector of applied inputs and a matrix of weights:

$$y_j = f\left(\sum_{i=1}^{M} x_i, w_{ij}\right) \tag{4.21}$$

Where the outputs, $y_j \in \mathbf{y} \in \mathbb{R}^{1\times N}$ is calculated using the inputs $x_i \in \mathbf{x} \in \mathbb{R}^{1\times M}$, weights $w_{ij} \in \mathbf{w} \in \mathbb{R}^{M\times N}$, and a nonlinear activation function f. Non-Ohmic behaviour manifests itself in individual devices only, therefore these nonlinear effects can be separated from the

other devices. To take I-V nonlinearities into account in ex-situ MNN training, the products in (4.21) can be replaced with a nonlinear function to give:

$$y_j = f\left(\sum_{i=1}^{M} g\left(x_i, w_{ij}\right)\right) \tag{4.22}$$

In order to account for the non-ohmic behaviour of memristors, Function g must be modified to reflect the specific characteristics of the mapping scheme between weights and conductances, as well as the non-ohmic device behaviour model, which is typically dependent on the type of devices employed [149].

The mapping scheme that relates weights and conductances is a crucial aspect to consider when designing memristive neural networks. In typical artificial neural networks, synaptic weights can assume any real value. Conversely, conductances are constrained to be non-negative.

To address this discrepancy, a neural network architecture was devised whereby twice the number of weights are trained, but each is constrained to be non-negative. This enables the association of weights with the conductances of individual devices, thereby creating a more natural mapping and facilitating adaptation to non-idealities. Additionally, it allows for more precise control over power consumption.

The aforementioned mappings are performed in an identical manner in the specific instance of fully connected synaptic layers in artificial neural networks, provided that conventional weight implementation methodologies are utilised. Both the inputs, designated as $x \in \mathbf{x}$, and the outputs, represented by $y \in \mathbf{y}$, are mapped onto voltages V and total output currents I, using the scaling factors k_V and k_I, where k_G is the conductance scaling factor:

$$V = k_V x \tag{4.23}$$

$$y = \frac{I}{k_I} = \frac{I}{k_V k_G} \tag{4.24}$$

$$k_G = \frac{G_{max} - G_{min}}{max|\mathbf{W}|} = \frac{G_{on} - G_{off}}{max|\mathbf{W}|} \tag{4.25}$$

Irrespective of the scheme employed, a minimum of two conductances are required to encode both positive and negative weights. Conductances G_+ and G_- are introduced into the "positive" and "negative" bit lines of the differential pair architecture, respectively. The proportionality of each weight is determined by the difference $G_+ - G_-$, with k_G serving

as the constant of proportionality. This enables the encoding of any real number within a finite range. In a typical differential pair implementation, $G_{min} = G_{off}$ and $G_{max} = G_{on}$. In an alternate proportional mapping scheme, $G_{min} = 0$ and $G_{max} = G_{on}$ to give $k_G = \frac{g_{on}}{max|\mathbf{W}|}$.

The issues associated with proportional mapping schemes, such as the presence of unimplementable regions, are not the only obstacles to be overcome. Conventional differential pair realisation also presents design challenges. For example, infinite conductance combinations will produce the same difference [166], i.e. the same effective conductance.

This means that an arbitrary choice may have to be made of how to perform the mapping between weights and conductances. To illustrate this, consider the encoding of weights $W \in \mathbf{W}$. In this case, pairs of conductances G_+ and G_- may be picked symmetrically around the average value:

$$G\pm = G_{avg} \pm \frac{k_G w}{2} \tag{4.26}$$

$$G_{avg} = \frac{G_{off} + G_{on}}{2} \tag{4.27}$$

While multiple mapping schemes may yield the same conductance difference, some may prove more advantageous than others. The scheme in (4.26) can be beneficial in certain scenarios, as it typically reduces the number of conductances near G_{off} and G_{on}, which are often more challenging to achieve. However, the selection of the mapping scheme can be explicitly linked to specific objectives.

The differential pair architecture can be employed to mitigate the effects of stuck devices. To illustrate, if G_+ is stuck in an undesirable state, G_- can be adjusted to minimise the negative effects. An alternative approach is to employ a simpler scheme that optimises a specific metric, such as power consumption.

This is illustrated in the simulations presented in this chapter, where conventional weights are used. In these simulations, a scheme that minimises power consumption is employed by ensuring that at least one of G_+, G_- is set to G_{off}:

$$G_+ = G_{off} + max\{0, k_G W\} \tag{4.28}$$

$$G_- = G_{off} - min\{0, k_G W\} \tag{4.29}$$

An alternative approach is to utilise two sets of non-negative weights, $W_{ij}^{+} \in \mathbf{W}_{+} \in \mathbb{R}_{\geq 0}^{M \times N}$ and $W_{ij}^{-} \in \mathbf{W}_{+} \in \mathbb{R}_{\geq 0}^{M \times N}$, which are collectively referred to as double weights [161]. This method entails mapping each weight onto a single conductance in the aforementioned "positive" and "negative" bit lines, respectively.

Despite the nonnegativity of each weight, the differential pair architecture allows for encoding the negative contribution of the i_{th} input on the j_{th} output through the introduction of a subtraction operation in hardware. Only the nonlinearity-aware node function in (4.22) requires adjustment to give:

$$y_j = f\left(\sum_{i=1}^{M} g\left(x_i, W_{i,j}^{+}\right) - g\left(x_i, W_{i,j}^{-}\right)\right) \tag{4.30}$$

As all weights in $\mathbf{W} := [W_{+}, W_{-}]$ are non-negative, they can be related to the corresponding conductances in the same way: $W_{\pm} \in [0, max(\mathbf{W})]$ can be linearly mapped onto $G_{\pm} \in [G_{off}, G_{on}]$, thus avoiding the introduction of weight gaps:

$$G_{\pm} = k_G W_{\pm} + G_{off} \tag{4.31}$$

The initial benefit of utilising double weights is that the conductances are more directly exposed to the training algorithm. This enables the selection of combinations that achieve both optimal performance, for example in terms of loss and robustness. To illustrate, if specific non-idealities manifest more at lower conductance values, such as with programming deviations [168], the training may be able to select pairs at higher values, while maintaining the same difference between G_{+} and G_{-}. This allows the minimisation of the negative effects of non-idealities without the need to explicitly specify which conductance pairs should be selected.

When weights W are related to conductances G in a monotonically increasing fashion (in this case, linear), regularisation can be employed to influence the magnitude of both weights and conductances. This provides a means of utilising regularisation as a high-level tool for controlling power consumption. It has been proposed that the *L1* sparsification regulariser [106] should be employed, as this has the potential to not only enhance training, for instance, by preventing overfitting, but also to promote lower conductance values.

In lieu of manually adjusting the mapping scheme, the network designer may elect to prioritize low power consumption to a greater or lesser extent. This may be determined by,

for example, adjusting the regularisation factor in *L1* regularisation and incorporated into the conventional hyperparameter tuning process [83], which is typically performed before deploying SNNs in the real world.

4.2.4 Non-idealities Calibrations

In the simulation presented in this chapter, a combination of experimental data from a SiO_x device, models based on simplified assumptions regarding breaking-down devices, and findings from the literature on programming variability were employed. I-V nonlinearities have been identified as a prevalent method for characterising deviations from ohmic behaviour in memristive devices.

$$\gamma \equiv \frac{G\left(2V_{ref}\right)}{G\left(V_{ref}\right)} \tag{4.32}$$

This approach involves the examination of two points on an I-V curve [192], as previously discussed (4.32). This equation defines conductance linearity, which serves as a means of quantifying nonlinear I-V behaviour. A conductance linearity value of 1 is indicative of ohmic behaviour, while any deviation from this value indicates I-V nonlinearity. While this metric can be useful for describing non-ohmic behaviour at different voltages, it is more challenging to utilise for modelling purposes [330].

$$I = V_{ref} G \left(\frac{V}{V_{ref}}\right)^{log_2 \gamma} \tag{4.33}$$

In order to construct an I-V curve from a given nonlinearity parameter γ, it is possible to assume that the equality in Equation (4.32) holds for all V, rather than just V_{ref}. This would result in a relationship between current and voltage, as shown in (4.32). In this equation, G represents the conductance parameter, which has a specific meaning in the context of V_{ref}, where the device produces the expected ohmic amount of current, which is equal to $V_{ref}G$.

The nonlinearity parameters γ were extracted from experimental data obtained from a SiO_x RRAM device. SiO_x devices are capable of undergoing resistance switching, which is characterised by a typical I-V switching curve. In order to achieve a wide range of resistance states and to analyse I-V nonlinearity, incremental positive sweeps were employed to gradually reset the device from the low-resistance state (LRS) to the high-resistance state (HRS). The low-resistance discrete states display greater linearity and exhibit minimal variability. Con-

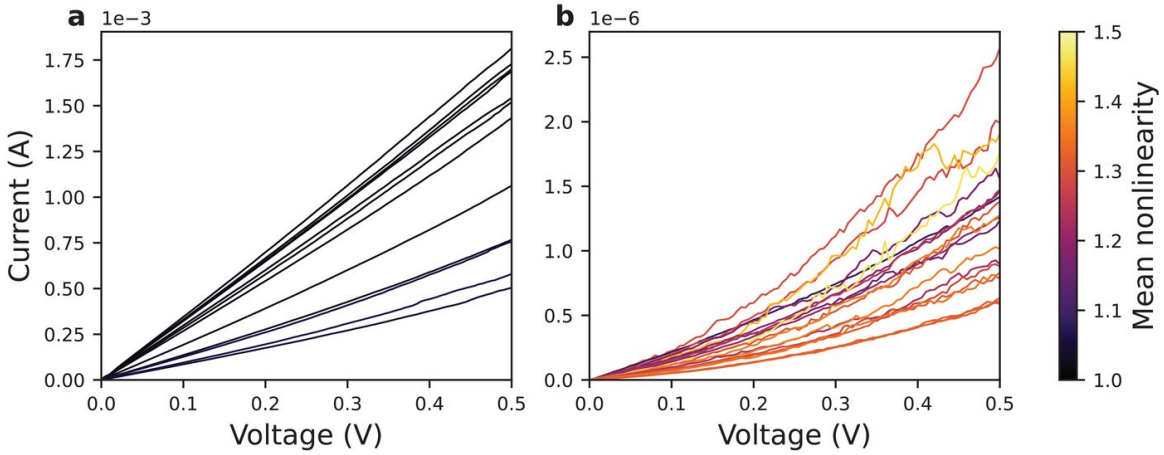

Fig. 4.6 I-V sweeps of a SiOx device are presented for two regions: a) low-resistance region with average resistance ranging from 284.6 Ω to 1003 Ω, and b) high-resistance region with average resistance ranging from 366.2 kΩ to 1.295 MΩ. Only the voltage range from 0.0 V to 0.5 V was considered for all curves. The nonlinearity parameter was calculated by dividing the current at 0.5 V by the current at 0.25 V [149].

versely, the high-resistance states are more nonlinear, and the nonlinearity is less predictable.

The value of V_{ref} from (4.32), was set to 0.25V, which is equivalent to half of the minimum switching voltage. In each case, the devices could be set to any conductance between $G_{off} = 1/R_{off}$ and $G_{on} = 1/R_{on}$. The states for the two groups were selected in such a way that the G_{on}/G_{off} ratio for high-resistance devices would be slightly larger. This approach ensures that any potential higher error rate in this group can be attributed to nonlinearity and its variability, rather than being attributed to limited dynamic range.

In order to guarantee more robust modelling, it was deemed necessary to take uncertainty regarding the degree of nonlinearity experienced by each device into account. Although there is a general tendency for high-resistance states to exhibit greater nonlinearity, the precise degree of nonlinearity at any given resistance state may be challenging to ascertain, as evidenced by the variability observed in the I-V curves depicted in the experimental data presented in Figure 4.6.

Consequently, for each device in either of the groups, the parameter γ was drawn from a truncated normal distribution. This distribution was truncated for γ values of less than 2, with the objective of ensuring realistic device behaviour and numerical stability of the simulations. The mean m_γ and standard deviation s_γ, both of which refer to the underlying

normal distribution prior to truncation, were used to characterise the distribution.

It is not possible to simulate nonidealities such as I-V nonlinearity using conventional noise injection methods that merely disturb the conductance values. Consequently, a forward propagation function must be defined in order to reflect the nonlinear relationship between inputs and outputs.

$$g(x, w_{\pm}) = \frac{\left(k_G w_{\pm} + G_{off}\right) \times (2x)^{log_2 \gamma_{\pm}}}{2k_G} \tag{4.34}$$

In the proposed training function, as outlined in Equation (4.30), the aforementioned I-V nonlinearity is incorporated into the function g by combining Equations (4.25), (4.31), (4.33), using $k_V = 2V_{ref}$ in accordance with the definition in Equation (4.32), resulting in the form presented in Equation (4.34). This function is then implemented using Keras.

In order to conduct simulations involving training, it is essential to utilise an I-V model that incorporates stochasticity. As a specific instance of nonlinear behaviour may be acquired during training, it is unclear how beneficial such a model would be when applied to a different set of devices.

It can be seen, therefore, that the previous approach may not be sufficient, since the experimental I-V measurements were used as a lookup table for computing currents of devices in certain conductance states at certain voltages. There are, however, multiple physical models that could be employed for describing non-ohmic memristor behaviour.

It was decided that the Poole-Frenkel conduction model [149] would be the most appropriate to use, given that the underlying physical mechanism was deemed to be plausible for SiO_x devices. Furthermore, the model demonstrated excellent fit to the experimental I-V curves. Its simple analytical form also allowed for the incorporation of stochasticity by considering the uncertainty in certain parameters.

$$I = cVexp\left(\frac{2e}{k_B T}\sqrt{\frac{eV}{4\pi d\varepsilon}}\right) \tag{4.35}$$

The Poole-Frenkel model postulates that the current through a device can be described by (4.35), where I is the current, V is the voltage, c is a constant (with units of conductance), T is the temperature which is 20 degrees Celsius, d is the effective oxide thickness, and ε is the

permittivity. Consequently, the parameter c and the product $d\varepsilon$ can be fitted.

In order to model the variability of c and $d\varepsilon$, an attempt was made to predict their values on the basis of observable variables. Deviations from any constructed trend would be indicative of the uncertainty in the values of these quantities. It has been demonstrated that memristive devices can behave differently in different resistance states [239], and this phenomenon has often been linked to the conductance quantum $G_0 = 2e^2/h$ [397].

The variables c and $d\varepsilon$ serve to reiterate some of the observations made regarding the I-V behaviour of the SiO_x device, thereby assisting in the assessment of the appropriateness of the Poole-Frenkel model. It was demonstrated that c behaves in a manner analogous to that of conductance, specifically as the reciprocal of resistance [147]. The primary distinction between the lower- and higher-resistance states is that the discrepancies from the trend are markedly more pronounced in the latter.

In (4.35), the product $d\varepsilon$ is indicative of the degree of nonlinearity. Given that this product appears in the denominator of an exponentiated square root, a smaller value of $d\varepsilon$ corresponds to a more nonlinear I-V curve. In LRS, this product is significantly larger. Although this parameter algebraically (4.35) can approximate linear behaviour (i.e. ohmic conduction), the values of $d\varepsilon$ are only plausible for the less conductive states proximate to and below G_0.

For higher-resistance states, this is less discernible. The $d\varepsilon$ values are considerably lower for all such states, yet the trend is not only plateaued but also more erratic. This unpredictability is indicative of the diverse range of colours observed in the curves presented in Figure 4.6. While the significant deviations from the trend line may limit the applicability of this approach in conventional modelling scenarios, the inherent uncertainty is precisely what is required to assess the potential of ex-situ training in addressing unknown behaviours.

The individual data points were integrated into a statistical model that would facilitate the generation of as many data points as required. In order to evaluate the efficacy of the proposed training method, two distinct resistance regions were considered. The low-resistance range was constructed by means of interpolation of the model parameters between the lowest resistance state and five times that resistance. The high-resistance range was constructed by interpolating the model parameters between the highest resistance state and one-fifth of that resistance, to ensure that the dynamic range remained consistent.

The statistical model was employed to ascertain the output current of a given device in the following manner: the initial values of c and $d\varepsilon$ were interpolated from the pertinent fits using the resistance parameter R. Thereafter, c and $d\varepsilon$ were subjected to disturbance via a multivariate normal distribution, which took into account both sets of parameters. The covariance matrix was determined using the residuals of the fits, and the current I was determined using (4.35).

As previously stated, the selection of the output function g in ex-situ training with linearity-nonpreserving nonidealities is contingent upon both the mapping scheme and the intrinsic nature of the nonlinearity. The combination of a linear mapping between inputs and outputs (4.25), a double weights mapping (4.26), and the relationship between current and voltage characterised by c and $d\varepsilon$ (4.35); allows g to be expressed as follows:

$$g\left(x, W_{\pm}\right) = \frac{cx \cdot exp\left(\frac{2e}{kbT}\sqrt{\frac{ek_V x}{4\pi d\varepsilon}}\right)}{k_G} \tag{4.36}$$

$$\begin{bmatrix} c \\ d\varepsilon \end{bmatrix} = exp\left(-ln\left(k_G W_{\pm} + G_{off}\right)\mathbf{m} + \mathbf{b} + \mathbf{E}\right) \tag{4.37}$$

Where $\mathbf{m}$ is the slopes, $\mathbf{b}$ is the intercepts, and the error is drawn from a normal distribution $\mathbf{E} \sim \mathcal{N}_2\left(0, \Sigma\right)$, with 0 mean and standard deviation being the residual covariance matrix.

4.3 Results

4.3.1 Simulation Configurations

4.3.2 Inference and Classification

4.4 Conclusion

Chapter 5

Homeostasis Optimisation

5.1 Introduction

5.1.1 Optimisation Overview

Spike-based optimisation techniques seek to minimise the overall error of a network by optimising the times of individual neuron spikes. As the time at which a neuron spikes is a continuous value, continuous optimisation methods can be employed. However, the problem remains highly non-linear due to the potential for a small change in the input to a neuron (and thus a small change to the neuron's input weights) to push the neuron over its firing threshold, eliciting a spike and drastically changing the neuron's output [103].

The initial algorithm to perform supervised deep learning by optimising spike times was SpikeProp [26]. This algorithm makes the simplifying assumption that each neuron will fire at most one spike during the spiking interval. In the event that multiple spikes are fired, only the first is optimised. Additionally, each connection is composed of numerous synaptic terminals, each with a distinct synaptic delay and connection weight.

The authors demonstrate that their algorithm can solve the XOR problem and performs comparably to backpropagation, which has been optimized both with gradient descent (GD) and Levenberg-Marquardt (LM), on a number of small datasets (the largest of which has 36 input dimensions, six output classes, and 4,435 training examples).

The single-spike optimisation procedure and multiple connection weights per synapse have presented significant challenges in expanding this work to larger datasets. Another work [235] presented two methods to improve the rate of convergence of SpikeProp; however, the

applications remain limited to small datasets.

Another publication put forth an alternative to the SpikeProp algorithm [256], which was designed with the specific intention of accommodating non-leaky integrate-and-fire neurons. The proposed method relaxes the restriction that a connection must be composed of many different discrete-delay elements, and instead employs a more standard network architecture, with one exponential-synapse connection between each pair of neurons. The networks were trained with both one and two hidden layers, achieving 2.45% and 2.86% accuracy on the MNIST dataset, respectively.

One challenge encountered by the author was that the dropout technique, the most commonly employed regularisation method, was ineffective in the network under consideration. This was because it frequently resulted in the complete cessation of neuronal firing. In the absence of an efficacious alternative regularisation method, the networks exhibited considerable generalisation errors, the training error for both networks was almost zero.

While earlier algorithms focused on optimising the initial spike of each neuron, another relaxed this constraint by employing a genetic algorithm to enhance multiple spikes from each neuron [325]. However, they limited their demonstration to relatively modest models comprising fewer than 10 hidden neurons. Genetic algorithms frequently encounter challenges in scaling to problems with numerous parameters, raising concerns about the scalability of this algorithm to datasets such as MNIST or CIFAR-10.

In a similar vein, another publication [190] also optimise over multiple spikes per neuron. The authors disregard the spiking discontinuity that occurs during backpropagation, instead treating the output of a neuron as a linear function of its inputs, which have been filtered by the membrane of the neuron in question. This enables the network to be run in spiking neurons during training, while still performing backpropagation without concern for discontinuities.

Furthermore, the refractory period of the neurons is disregarded, as it is deemed to be relatively brief in comparison to the time interval between spikes, and therefore has a minimal impact on firing rates. In order to enhance the performance of the network, lateral inhibition components are introduced; however, only the first-order derivatives caused by these connections are optimised.

Notwithstanding these simplifications, their method is still capable of learning appropriately on the MNIST task, achieving 1.30% error using standard SGD and 1.23% error using an ADAM optimiser. While it remains to be seen whether this method can generalise to larger datasets in a tractable way, it introduces a number of new ideas for training spiking networks that will hopefully be improved upon by future work.

A different study introduce a novel method for making the spiking process continuous [133]. They introduce a gating function, *g(V)*, of the membrane voltage, *V*, that is greater than zero for voltages approaching the firing threshold and zero otherwise, with unit integral. The region where $g(V) > 0$ is referred to as the active zone.

In contrast to the conventional approach, whereby efferent synapses receive a current $\delta(t - t_k)$ upon the neuron crossing the firing threshold at time t_k, this method allows synapses to continuously receive current based on $g(V)\frac{dV}{dt}$. In the event that the voltage is situated outside the active zone, this term is rendered zero due to the fact that *g(V)* is equal to zero.

Conversely, should the voltage traverse the active zone (and thus the neuron spike), the integral of this term is equal to one. Ultimately, if the voltage enters the active zone but does not exceed the upper threshold (i.e., the firing threshold), the integral of the term will be a positive number between zero and one (this is analogous to a partial spike).

This induced synaptic current is nearly identical to the traditional spiking current $\delta(t - t_k)$ in the extreme cases (i.e., when the neuron is silent or firing at a relatively high rate), but continuous in the intermediate region. The authors may then achieve a gradient through the network at any given point in time by employing backpropagation, and optimise the entire network by utilising backpropagation through time.

The results presented focus on tasks that require a dynamic, temporal representation, such as predictive coding. This represents a distinct focus in comparison to the majority of other spike-based methods, which tend to prioritise static tasks (such as object classification). Consequently, a direct comparison is challenging.

As a consequence of the method rendering the neural nonlinearity differentiable, a multitude of distinct neuron models may be employed. The present paper utilises non-leaky integrate-and-fire and quadratic integrate-and-fire models, with a plethora of alternative models being

equally compatible.

This, in conjunction with the capacity to optimise recurrent spiking networks effectively, renders this a potentially potent methodology for dynamic spiking networks. In the context of networks specialising in static tasks, it is postulated that the necessity for this method to optimise over potentially lengthy time series for each input stimulus would render it unsuitable for training large object classification networks.

To date, spike-based optimisation methods have yet to be applied to larger, deeper architectures such as convolutional neural networks (CNNs). This has precluded the implementation of spike-based training on any datasets that are either larger or more challenging than the MNIST dataset.

One of the challenges lies in the computational requirements of spike-based optimisation methods, which necessitate more computational resources than rate-based methods. This is due to the dynamic nature of the network and the iterative simulation required for each stimulus presentation.

The majority of existing software has been designed for static artificial neural networks (ANNs), and extending it to spiking networks is a complex undertaking. Consequently, researchers have employed rate-based optimisation methods to address larger datasets with spiking networks, which will be discussed next.

Rate-based optimisation methods operate under the simplifying assumption that all neurons are engaged in rate coding. Consequently, these methods are indifferent to the times of individual spikes, focusing instead on the number of spikes occurring within a given time period.

In the majority of cases, these types of methods utilise this simplifying assumption to replace the spiking neural process with a continuous-valued rate approximation. In the case of derivative-based methods, this rate approximation is then differentiated. Derivative-free methods, in contrast, circumvent the necessity of taking the derivative of this rate approximation, instead opting for an optimisation method such as Contrastive Divergence that does not require it.

Finally, function approximation methods approach the problem from a different angle. Rather than assuming a network of spiking neurons and attempting to identify rate approximations to these spiking neurons, they select an arbitrary nonlinearity for training the network, and then employ spiking neurons to approximate this nonlinearity.

5.1.2 Derivative-based Methods

In 2010, a study pioneered the training of a spiking convolutional network [278]. A conventional convolutional neural network (CNN) was trained using the backpropagation algorithm with *tanh* units. In order to transform this into a spiking network, the *tanh* units were substituted with binary threshold units that incorporate a refractory period.

In particular, if the input to a neuron exceeds a specified threshold, it generates a spike, after which the neuron is unable to spike for a designated period. The refractory period causes the firing rates of the neurons to saturate. The authors posit that this emulates the corresponding *sigmoid* functions, namely the *tanh* function, used in the rate model.

However, they do not elucidate the precise mechanism through which the refractory period achieves this emulation. It may be presumed that the rationale is that the *tanh* function is a saturating nonlinearity, and thus having the neuron firing rate saturate makes the firing rate curve more similar to the *tanh* curve.

A more principled approach to the conversion of rate-based convolutional neural networks (CNNs) to spiking CNNs was adopted in 2015 [33]. They exploited the fact that rectified linear units (ReLUs) perfectly model the rate of a non-leaky integrate-and-fire neuron (IF). As ReLUs are currently the standard for deep networks, this allows networks to be trained with the same rate-based ReLUs and then replaced with spiking IF neurons at runtime.

Additionally, the max-pooling layers utilized by the network were replaced with average pooling layers. Average pooling layers only entail the summation and scaling of inputs, thereby enabling them to function effectively with spiking neurons, in a manner that is biologically plausible. In contrast, max pooling lacks a clear spiking analogue.

The process of taking the maximum of the binary spikes occurring at a given point in time (i.e., outputting one if any of the inputs are spiking, and zero otherwise) is not equivalent to taking the maximum across spike rates, as performed by traditional CNNs. Moreover, even taking the maximum across spikes is not biologically plausible, since there is no clear

neuron-level mechanism for taking the maximum across incoming signals.

The same study also removed biases from the network, on the grounds that they would be challenging to implement in a spiking network [33]. However, it is not evident why biases would present a problem for spiking networks, particularly if targeting neuromorphic hardware (much of which supports biases). One advantage of removing biases is that it offers a somewhat more straightforward biological explanation, since the output of each neuron depends solely on the inputs from other neurons in the network.

In a similar approach to that employed previously, a novel method of normalisation for weight magnitudes was utilised [63]. One pivotal design decision in spiking networks is the trade-off between the firing rates of the neurons and the requisite integration time for the network to reach a decision.

Higher firing rates facilitate the expeditious transmission of information, thereby enabling more rapid responses and enhanced accuracy. However, this increased speed comes at the cost of greater energy consumption. On certain neuromorphic platforms, there are imposed limits on the maximum allowable firing rates. An increase in integration time permits the accumulation of greater quantities of information, thereby enhancing accuracy. However, this approach also results in slower responses and an elevated number of spikes per example.

It is therefore necessary to achieve a balance between accuracy, response time and energy efficiency by properly tuning the firing rates and integration time. Given that ReLUs—and, by extension, IF neurons—are scale-invariant, the firing rates of neurons in spiking IF networks can be set at any arbitrary value. In order to alter the firing rate of a neuron while preserving the integrity of the network, it is necessary to offset the increase in firing rate with a corresponding decrease in all connection weights originating from that neuron.

The study present two techniques for normalising network weights, namely model-based and data-based normalisation [63]. Both techniques are founded upon the principle that the input to any given neuron in a single timestep should not exceed the firing threshold of that neuron. Should this occur, it could result in the neuron being required to fire two spikes in the same timestep, which is not supported by the majority of neuromorphic hardware.

In other words, the objective of the normalization process is to guarantee that the instantaneous firing rates of all neurons never exceed the rate of the neuromorphic chip (simulation

rate). In the model-based normalization technique, the maximum sum of positive input weights across all neurons in a layer is identified, and then all neuron inputs in the layer are normalized using this sum.

This guarantees that no neuron can receive an input greater than one (the firing threshold) in a single timestep. The authors discovered that this technique can result in weights that are smaller than necessary. This is because the maximum possible input to a neuron is often never encountered in real data, which in turn leads to longer-than-necessary integration times.

In contrast, data-based normalization is contingent upon the examples within the training data. This ensures that for any given training example, no neuron will receive an input greater than one (the firing threshold). The authors discovered that data-based normalization reduced the integration time required by the network to achieve the same level of accuracy as an unnormalized network with a higher firing threshold. This paper presented the most advanced results to date for spiking networks on the MNIST dataset, achieving a test-set error of 0.88%.

In a study published in 2016 [80], the authors trained deep networks targeting the IBM TrueNorth neuromorphic chip [247]. In contrast to the other networks described herein, this network employs a basic binary threshold neuron, which generates a "spike" in response to a positive input at a given time step and remains quiescent otherwise. This can be considered to be equivalent to a rate-based binary threshold unit.

Furthermore, the network presents each image for a single timestep before resetting all neuron voltages and presenting the next image. As a result of these considerations, the network is classified as rate-based, rather than spiking. The authors trained very large networks, comprising 8 million neurons, on a number of tasks, including CIFAR-10 and SVHN. The results are at the cutting edge of the field when compared to spiking networks.

However, a more pertinent comparison is with rate-based networks, where the sole innovation is the use of binary threshold units, which are not commonly employed in deep networks. The network is capable of running on neuromorphic hardware (TrueNorth), but utilises a considerable number of neurons even for relatively modest datasets such as CIFAR-10. Consequently, it would be challenging to scale up to larger datasets such as ImageNet.

5.1.3 Derivative-free Methods

Derivative-free, rate-based optimisation methods employ rate approximations for the spiking neural process, yet do not necessitate the derivative of the rate approximation. The entirety of the aforementioned methods are founded upon the utilisation of the Contrastive Divergence (CD) algorithm [119] for the training of deep networks comprising stacked RBMs.

In a study published in 2013 [268], a spiking network of LIF neurons using RBMs was developed. In order to train the network, the researchers employ the Siegert approximation to a noisy spiking LIF neuron as the rate neuron model. The Siegert approximation is an equation that describes the mean firing rate of an LIF neuron when presented with a number of input spike trains, where the spikes are distributed according to a Poisson process with a constant firing rate.

Subsequently, the output rates are normalised by the maximum firing rate (1/tref) in order to generate firing probabilities, and the hidden units are sampled from a binary distribution based on these probabilities. The authors employ this model to simulate the variability introduced by the use of spikes.

Consequently, spiking LIF neurons are introduced in place of the rate neurons, and the network is trained using Poisson input trains based on pixel intensities, resulting in an error rate of 5.91% on MNIST. A study in 2015 [326] further extend this work by demonstrating the method's resilience to reduced weight precision, achieving an error rate of 5.06% on MNIST with 11-bit fixed point weights running on the SpiNNaker platform.

In a similar vein, a 2013 model was also developed spiking RBMs using LIF neurons [261]. Their approach entails fitting the mean firing rate of a noisy LIF neuron to a sigmoid curve and subsequently training using this sigmoid curve. This restricts the range of possible parameters for the LIF neuron, as they must permit the firing rate to be accurately approximated by a sigmoid function. However, it permits the RBMs to be trained with conventional sigmoid activation functions, thereby facilitating a more straightforward and efficient training process.

They put forth an online variant of CD for spiking networks, based on an STDP learning rule. For each training image, two phases are conducted. The initial phase occurs immediately upon presentation of the image, wherein connections are updated based on the data (stimulus). Subsequently, following a designated period of time, the recurrent connections become active,

thereby driving the network towards its reconstruction distribution.

This has an effect that is similar to that of Gibbs sampling in the CD algorithm. During this period, the gating signal on the derivative is reversed, resulting in a negative impact on the weights based on the reconstruction. This constitutes one complete cycle of the algorithm, which is repeated for each training image over numerous iterations through the training data. This approach bears resemblance to CD, hence the authors have designated it as "event-based CD." Employing this technique, the researchers achieved an 8.1% test-set error on MNIST in spiking neurons.

5.1.4 Function-approximation and Noise

Approximation methods for functions diverge considerably from the previously discussed approaches. In contrast to the approach of utilising a single spiking neuron to represent each node in an ANN, function approximation methods employ multiple spiking neurons for each node. These spiking neurons are employed to approximate the nonlinear function (e.g., the sigmoid function) that is being computed by the node.

There is only one documented instance of this method being employed [74]. The approximation of the sigmoid function computed by each node is achieved through the use of three spiking LIF neurons. The parameters for each neuron are randomly assigned, with the most significant aspect being the uniform distribution of the random bias current. This enables each of the three neurons to target a distinct portion of the sigmoid curve, in addition to the bias current assigned to the node during the training process.

Once the random parameters have been set, a scalar weighting is calculated for each neuron, determining the extent to which it contributes to the sigmoid function. This is the standard method for function approximation, but applied to a limited number of neurons in order to approximate a sigmoid function. This method enables a spiking network to approximate a rate-based network in a highly general manner.

One disadvantage is that it requires a greater number of neurons than nodes in the network, which makes it more costly in terms of neural resources than the aforementioned methods. The efficacy of this approach is contingent upon the ability of the neurons to accurately represent the node function within the operational domain. Sigmoid functions can be readily represented with LIF neurons. However, ReLUs are considerably flatter and do not undergo squashing, necessitating a significantly larger range of output values for accurate representa-

tion. This renders them less compatible with this method.

The transmission of discrete, constant-amplitude spikes, as observed throughout much of the human central nervous system, represents a higher-variance method of information transmission than the simple transmission of a scalar value, such as a voltage. From a rate-based perspective, the increased variance can be conceptualised as 'noise' surrounding the firing rate, which can be considered the 'signal'. The firing rate is defined as the mean value of a spike train signal.

If a spike train is passed through a low-pass filter, the result can be viewed as an estimate of the mean with additional time-varying noise superimposed. The quantity of noise is contingent upon the filter employed; the elimination of a greater proportion of high frequencies will yield a superior estimate with diminished noise, yet if the fundamental firing rate signal is also time-varying, then its higher frequencies will also be removed. In a spiking network that is receiving a sequence of images and outputting a response for each in real-time, this results in a longer response time for each image.

It is observed that a considerable number of rate-based spiking neural networks do not take into account the variance caused by spikes. Amongst those that do, two distinct approaches have been identified. The first involves modelling the impact of spike noise on the mean neural firing rates, with these revised mean rates then being employed during the training process. The second approach entails incorporating stochastic elements into the model during training, with the objective of ensuring that the probability distribution of the training output rates reflects, to some extent, the variance resulting from the spikes employed during testing.

An exemplar of this initial approach is the utilisation of the Siegert model to account for the effects of incoming spikes on the LIF neuron firing rate [268]. This approach is predicated on the assumption that neuronal inputs are uncorrelated and that spikes are governed by a Poisson process.

The Siegert model then provides the mean firing rate of the LIF neuron, given the mean firing rates of all inputs and their respective connection weights. One disadvantage of this approach is that the Siegert equation is complex, containing an integral with no closed-form solution. The integrand of this integral contains both the exponential and error (erf) functions.

Computing this function on a GPU (or even a CPU) would be challenging and time-consuming in comparison to the functions typically employed in neural networks. Additionally, this approach necessitates the calculation of two linear functions of the input neuron rates, one to determine the overall input mean and the other to ascertain the variance. Consequently, it requires twice the number of matrix-vector operations as a network without the Siegert model.

The second approach, which involves incorporating stochastic elements into a model of spiking noise, has frequently been employed in a range of neural network contexts. However, this has not typically been done with the specific objective of running the final model in spiking neurons.

A foundational example is that of Boltzmann machines, which employ binary sampling on probabilistic units [120]. This enables units to communicate binary values with one another, which bears resemblance to spikes. However, if one considers each unit to be attempting to transmit its underlying firing probability (which is analogous to a firing rate), this represents a particularly severe form of noise.

Subsequent work significantly reduced the amount of noise by employing rectified linear activation functions with Gaussian noise on the neuron input [260]. An additional illustration of the incorporation of stochastic elements into a model is the denoising autoencoder [355]. This approach entails the introduction of noise into the model inputs, followed by an attempt to reproduce the original, noise-free version of the input.

The addition of noise to model inputs has frequently been employed as a means of enhancing the model's capacity to handle a diverse range of inputs, effectively constituting a form of data augmentation. Other forms of data augmentation, such as translation, rotation, or deformation of images, can also be regarded as a structured form of noise on model inputs, distinct from Gaussian noise.

Nevertheless, the incorporation of noise into hidden units, whether at the input or output level, remains a relatively uncommon practice in the context of nonlinear networks [284]. Despite extensive research, there seems to be a dearth of examples where this approach has been employed to model the inherent variability associated with spikes, with the aim of translating a rate model into a spiking neuron framework.

5.2 Methodology

5.2.1 Neural-engine Framework

While it is possible to modify the mapping scheme in order to achieve specific behaviours, this is a highly detailed and complex approach that requires a fundamental understanding of the physics of memristors. Therefore, even when training is conducted externally, the network designer must consider not only the conventional training hyperparameters, such as learning rates or the number of epochs, but also the physical implementation of MNNs, as this will affect the network performance.

In order to make MNNs a feasible proposition in the real world, it may be necessary to adopt more high-level approaches that require minimal understanding of the hardware. This is similar to how the Compute Unified Device Architecture (CUDA) hides the complexity of graphics processing units (GPUs) behind high-level software abstractions [72].

The original Neural Engineering Framework is a theory that connects neural-level computations with higher-level algorithmic descriptions [73]. Specifically, it describes how neurons can perform vector computations, which can then be built upon to create increasingly complex systems [74]. It has three main tenets: representation, how neurons represent a real-world value; transformation, how neurons compute (static) functions on represented values; and dynamics, how neurons can implement dynamical systems or functions.

The Neural Engineering Framework is a scheme that was founded upon the principle of population coding [300]. In particular, it employs a methodology whereby linear least-squares techniques are utilised to identify linear decoders for a population of neurons. This concept can be succinctly encapsulated as nonlinear encoding, linear decoding.

The fundamental premise of the NEF is that a "population" of neurons can be conceptualised as a representation of a specific value. This could be either a value in the real world or a purely internal value. In the core NEF, each population represents a vector $\mathbf{x}$, but this can be extended to representing things like scalar- and vector-fields, too.

Each neural population is equipped with a set of encoders, designated as $\mathbf{E}$, which facilitate the mapping from the state space, a d-dimensional representation of the values that the population can represent, to the neuron space, a n-dimensional representation of neural

activities. The mapping is nonlinear.

$$a_j(\mathbf{x}) = G_j\left(\alpha_j \sum_i E_{ji} x_i + \beta_j\right) \tag{5.1}$$

In this context, the vector value represented by $\boldsymbol{x}$ denotes the encoding of the population, while a_j signifies the activity of neuron j, $G_j(\cdot)$ represents a rate-based neural nonlinearity that transforms input currents into firing rates. E is a matrix of dimensions $n \times d$, comprising encoders, while α_j and β_j are scalar gain and bias terms, respectively. It is common practice to model neurons in the NEF as LIF neurons. In such cases, $G_j(\cdot)$ is given by Equation 2.17 for all neurons.

In order to decode information from the population, it is necessary to identify a suitable linear decoder D. When this is multiplied by the neural activities $\boldsymbol{a(x)}$, the resulting value $\boldsymbol{x}$ represents the state-space value. In practice, the choice of decoder is typically based on the minimisation of the squared error $\|D\mathbf{a}(\mathbf{x}) - \mathbf{x}\|_2^2$.

In an ideal scenario, the objective would be to minimise this across all possible values of $\boldsymbol{x}$. However, in practice, the space of $\boldsymbol{x}$ is sampled at m points, which are referred to as evaluation points. Given an $m \times d$ matrix of evaluation points, $\boldsymbol{X}$, and the corresponding neural activities at each evaluation point, $\boldsymbol{A}$, the least-squares optimal decoders can be found as follows:

$$D = (A^\intercal A)^{-1} A^\intercal X \tag{5.2}$$

In practice, some neurons may exhibit similar activity patterns, rendering this an ill-conditioned problem. Furthermore, it is common practice to impose a penalty on large decoders in order to mitigate the impact of variability in the spiking neurons on the decoded value. To address these issues, regularisation is typically employed.

Given that the number of neurons in the population n is typically much larger than the number of dimensions d in the encoded vector $\boldsymbol{x}$, it follows that the neural population not only contains information about $\boldsymbol{x}$, but also about possible functions of $\boldsymbol{x}$. The decoding of a function $f(\boldsymbol{x})$ from the population can be achieved by identifying decoders Df that minimise the squared error $\left\|D_f \mathbf{a}(\mathbf{x}) - \mathbf{x}\right\|_2^2$.

This results in a similar least-squares optimisation problem as previously described, but with the matrix of evaluation points X replaced by a matrix of targets Y. This method can be used to decode arbitrary functions from the population. The accuracy with which a given function can be decoded depends on the characteristics of the function (e.g. higher frequency changes are more difficult to decode), the tuning curves of the neurons, and the relationship between these two factors.

One notable aspect of the encoding model is its rate-based structure. The decoders are identified through the utilisation of the rate response function of the neurons. This does not imply that NEF models employ traditional rate coding. Indeed, firing rates can vary with considerable rapidity, and one can construct models, for example, an oscillator, where neurons fire in rapid bursts, the timing of which is of consequence.

It is not possible for NEF networks to exploit synchronisation between neurons. Any correlations between neural firing will merely result in increased variability (noise) in the decoded output. Furthermore, NEF networks function equally well or better when rate-based neurons are used instead of spiking neurons. This indicates that these networks utilise a fundamentally rate code, albeit one that can change quickly.

A significant feature of NEF networks is the heterogeneity of neurons with regard to their encoders and biases, which results in differential responses to input stimuli. This heterogeneity enables neurons to not only encode different aspects of their input signals a capability that can also be demonstrated by noisy homogeneous neurons [134], but also to compute nonlinear transformations of these signals.

The precise nature of this heterogeneity has a substantial impact on both the signals that can be represented and the transformations that can be computed. It is not always the case that representation and transformation are mutually exclusive; for instance, a population of neurons that are all tuned to the sum of two inputs is unable to represent the individual signals, but is capable of computing a transformation such as the square of their sum. There has been limited examination of the types of heterogeneity that are most effective for computing specific functions.

5.2.2 Programming Variabilities

One of the most common non-idealities observed in memristive devices is the phenomenon of devices becoming stuck. This topic was previously explored in depth in the preceding

chapter. In this chapter, a similar model is employed, in which devices may become stuck at G_{off} or G_{on}. For both types of simulations, a range of probability were used, indicating that any individual may become stuck in that state. Although this is a simple model, it is not data-specific, and thus could be combined with the nonidealities that were modelled using experimental data, specifically SiO_x nonlinearities.

A more realistic probabilistic model for describing stuck device behaviour in crossbar array can be introduced. This is achieved by randomly drawing from a sample of all memristors and setting the conductance to the closest achievable state. Although this method is relatively robust given the high number of devices, it is discrete in nature and therefore more difficult to apply during SNN training, where gradients need to be computed.

Furthermore, the method is limited in data in certain conductance regions, which may result in training that fits the weights to the behaviour of a single instance of a crossbar array. Consequently, a more continuous approach to picking the states at which the devices may get stuck in was adopted.

It is possible to display previously encountered data; in this case, all potentiation and depression cycles are shown for some of the devices. The majority of memristors are capable of achieving a high conductance range, with only a small proportion exhibiting limited conductance or remaining in a fixed state. In this chapter, the simulations were conducted with G_{off} and G_{on} defined as the median of minimum and maximum conductances, respectively.

Any device whose maximum range (i.e., the difference between the highest and lowest conductances achieved) was less than half the median range (Gon - Goff) was classified as stuck. A further simplifying assumption was made that any such device would be treated as fully stuck. This overestimates the effect of the variability because in reality some of the 'stuck' devices may still be tweaked, albeit within a narrower range.

The challenge in constructing a probabilistic model in this case is the generation of a probability density function (PDF) that accurately describes the conductance values at which devices may become stuck. The selection of devices that may become stuck is a relatively straightforward process.

These devices can be chosen randomly, with the proportion of devices that fit the previously defined criteria for stuck behaviour being the determining factor. For instance, 10% of the

devices may be deemed to exhibit the necessary characteristics. However, the conversion of the discrete mean conductance values of these devices into a probability distribution that can be applied to a wider range of situations represents a more challenging aspect of this process.

In order to produce a probabilistic model, it was decided that kernel density estimation (KDE) would be employed. KDE is a method of producing a probability density function (PDF) given a sequence of randomly distributed variables. Each point is usually approximated with a random distribution, which are then summed together.

As these distributions are typically identical, the only choices that have to be made are the type of distributions to be used, or the width of those distributions, which is more commonly referred to as "bandwidth" [347] in the context of KDE. In this chapter, it was decided to employ truncated normal distributions, truncated at zero to circumvent the issue of negative conductance.

The underlying normal distributions were set to have a mean equal to the mean conductance of faulty devices, with the objective of estimating the standard deviation of the underlying distributions. In order to achieve this, Scott's rule [305] was utilised. As a consequence of the clipping of conductance values below zero, a bias is introduced, whereby the probability density is underestimated in the vicinity of the boundary, even if the PDF is re-scaled after truncation [317].

To address this issue, a reflection method can be employed. Rather than performing re-normalisation, a second distribution was introduced, representing the reflection of the original normal distribution around zero. The negative part of this distribution was then clipped [150]. This approach ensures that the two truncated PDFs sum to one, while providing a more reliable estimate of the probability density near zero.

Furthermore, device-to-device (D2D) variability, which arises from inaccuracies during device programming, was also taken into account. As discussed in the preceding chapter, during the mapping of weights onto conductances, one may ultimately arrive at values that differ from the desired outcome.

Lognormal distribution is a commonly utilized approach to model these discrepancies. For instance, it has been demonstrated that resistance deviations adhere to this trend, with the

relative (and consequently, the absolute) magnitude of the deviations being more pronounced in the high-resistance state (HRS) compared to the low-resistance state (LRS).

5.2.3 Architecture Modifications

5.2.4 Biosignal Applications

5.3 Results

5.4 Conclusion

Chapter 6

Summary

6.1 Contributions

6.2 Open Questions

6.3 Future Works

References

[1] Abbott, L. F. (1999). Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain research bulletin*, 50(5-6):303–304.

[2] Adler, D., Shur, M., Silver, M., and Ovshinsky, S. (1980). Threshold switching in chalcogenide-glass thin films. *Journal of Applied Physics*, 51(6):3289–3309.

[3] Agrawal, A., Lee, C., and Roy, K. (2019). X-changr: Changing memristive crossbar mapping for mitigating line-resistance induced accuracy degradation in deep neural networks. *arXiv preprint arXiv:1907.00285*.

[4] Ambrogio, S., Balatti, S., Cubeta, A., Calderoni, A., Ramaswamy, N., and Ielmini, D. (2014a). Statistical fluctuations in hfo x resistive-switching memory: part i-set/reset variability. *IEEE Transactions on electron devices*, 61(8):2912–2919.

[5] Ambrogio, S., Balatti, S., Cubeta, A., Calderoni, A., Ramaswamy, N., and Ielmini, D. (2014b). Statistical fluctuations in hfo x resistive-switching memory: Part ii—random telegraph noise. *IEEE Transactions on Electron Devices*, 61(8):2920–2927.

[6] Ambrogio, S., Narayanan, P., Tsai, H., Shelby, R. M., Boybat, I., Di Nolfo, C., Sidler, S., Giordano, M., Bodini, M., Farinha, N. C., et al. (2018). Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature*, 558(7708):60–67.

[7] Amirsoleimani, A., Alibart, F., Yon, V., Xu, J., Pazhouhandeh, M. R., Ecoffey, S., Beilliard, Y., Genov, R., and Drouin, D. (2020). In-memory vector-matrix multiplication in monolithic complementary metal–oxide–semiconductor-memristor integrated circuits: Design choices, challenges, and perspectives. *Advanced Intelligent Systems*, 2(11):2000115.

[8] Ankit, A., Hajj, I. E., Chalamalasetti, S. R., Ndu, G., Foltin, M., Williams, R. S., Faraboschi, P., Hwu, W.-m. W., Strachan, J. P., Roy, K., et al. (2019). Puma: A programmable ultra-efficient memristor-based accelerator for machine learning inference. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 715–731.

[9] Asanuma, S., Akoh, H., Yamada, H., and Sawa, A. (2009). Relationship between resistive switching characteristics and band diagrams of ti/pr 1- x ca x mno 3 junctions. *Physical Review B*, 80(23):235113.

[10] Azghadi, M. R., Lammie, C., Eshraghian, J. K., Payvand, M., Donati, E., Linares-Barranco, B., and Indiveri, G. (2020). Hardware implementation of deep network accelerators towards healthcare and biomedical applications. *IEEE Transactions on Biomedical Circuits and Systems*, 14(6):1138–1159.

[11] Azghadi, M. R., Linares-Barranco, B., Abbott, D., and Leong, P. H. (2016). A hybrid cmos-memristor neuromorphic synapse. *IEEE transactions on biomedical circuits and systems*, 11(2):434–445.

[12] Azghadi, M. R., Moradi, S., Fasnacht, D. B., Ozdas, M. S., and Indiveri, G. (2015). Programmable spike-timing-dependent plasticity learning circuits in neuromorphic vlsi architectures. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 12(2):1–18.

[13] Ba, J. and Caruana, R. (2014). Do deep nets really need to be deep? *Advances in neural information processing systems*, 27.

[14] Bajo, J. and Corchado, J. M. (2018). Neural networks in distributed computing and artificial intelligence. *Neurocomputing*, 272(C):1–2.

[15] Banerjee, R., Ray, S., Basu, N., Batabyal, A., and Barua, A. (1987). Degradation of tin-doped indium-oxide film in hydrogen and argon plasma. *Journal of applied physics*, 62(3):912–916.

[16] Bao, L., Zhu, J., Yu, Z., Jia, R., Cai, Q., Wang, Z., Xu, L., Wu, Y., Yang, Y., Cai, Y., et al. (2019). Dual-gated mos2 neuristor for neuromorphic computing. *ACS applied materials & interfaces*, 11(44):41482–41489.

[17] Bardeen, J. (1947). Surface states and rectification at a metal semi-conductor contact. *Physical review*, 71(10):717.

[18] Barmpatsalos, N. and Mehonic, A. (2021). SiOx memristor structure - Gradual RESET conductance modulation.

[19] Bauer, F. C., Muir, D. R., and Indiveri, G. (2019). Real-time ultra-low power ecg anomaly detection using an event-driven neuromorphic processor. *IEEE transactions on biomedical circuits and systems*, 13(6):1575–1582.

[20] Bear, M. F. and Malenka, R. C. (1994). Synaptic plasticity: Ltp and ltd. *Current opinion in neurobiology*, 4(3):389–399.

[21] Behrenbeck, J., Tayeb, Z., Bhiri, C., Richter, C., Rhodes, O., Kasabov, N., Espinosa-Ramos, J. I., Furber, S., Cheng, G., and Conradt, J. (2019). Classification and regression of spatio-temporal signals using neucube and its realization on spinnaker neuromorphic hardware. *Journal of neural engineering*, 16(2):026014.

[22] Bellec, G., Scherr, F., Hajek, E., Salaj, D., Subramoney, A., Legenstein, R., and Maass, W. (2019). Eligibility traces provide a data-inspired alternative to backpropagation through time.

[23] Bien, N., Rajpurkar, P., Ball, R. L., Irvin, J., Park, A., Jones, E., Bereket, M., Patel, B. N., Yeom, K. W., Shpanskaya, K., et al. (2018). Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of mrnet. *PLoS medicine*, 15(11):e1002699.

[24] Bill, J. and Legenstein, R. (2014). A compound memristive synapse model for statistical learning through stdp in spiking neural networks. *Frontiers in neuroscience*, 8:412.

[25] Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

[26] Bohte, S. M., Kok, J. N., and La Poutre, H. (2002). Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37.

[27] Boikov, Y. A., Goltsman, B., Yarmarkin, V., and Lemanov, V. (2002). Near-electrode model of transient currents in (ba, sr) tio 3 thin film capacitor structures. *Applied physics letters*, 80(21):4003–4005.

[28] Bromley, A. G. (1982). Charles babbage's analytical engine, 1838. *Annals of the History of Computing*, 4(3):196–217.

[29] Buckwell, M., Montesi, L., Hudziak, S., Mehonic, A., and Kenyon, A. J. (2015). Conductance tomography of conductive filaments in intrinsic silicon-rich silica rram. *Nanoscale*, 7(43):18030–18035.

[30] Burnstock, G. (2004). Cotransmission. *Current opinion in pharmacology*, 4(1):47–52.

[31] Burr, G. W., Shelby, R. M., Sidler, S., Di Nolfo, C., Jang, J., Boybat, I., Shenoy, R. S., Narayanan, P., Virwani, K., Giacometti, E. U., et al. (2015). Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Transactions on Electron Devices*, 62(11):3498–3507.

[32] Cai, F., Correll, J. M., Lee, S. H., Lim, Y., Bothra, V., Zhang, Z., Flynn, M. P., and Lu, W. D. (2019). A fully integrated reprogrammable memristor–cmos system for efficient multiply–accumulate operations. *Nature Electronics*, 2(7):290–299.

[33] Cao, Y., Chen, Y., and Khosla, D. (2015). Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113:54–66.

[34] Ceolini, E., Frenkel, C., Shrestha, S. B., Taverni, G., Khacef, L., Payvand, M., and Donati, E. (2020). Hand-gesture recognition based on emg and event-based camera sensor fusion: A benchmark in neuromorphic computing. *Frontiers in Neuroscience*, 14:637.

[35] Cha, E., Woo, J., Lee, D., Lee, S., Song, J., Koo, Y., Lee, J., Park, C. G., Yang, M. Y., Kamiya, K., et al. (2013). Nanoscale (10nm) 3d vertical reram and nbo 2 threshold selector with tin electrode. In *2013 IEEE International Electron Devices Meeting*, pages 10–5. IEEE.

[36] Chai, Z., Freitas, P., Zhang, W., Hatem, F., Zhang, J. F., Marsland, J., Govoreanu, B., Goux, L., and Kar, G. S. (2018). Impact of rtn on pattern recognition accuracy of rram-based synaptic neural network. *IEEE Electron Device Letters*, 39(11):1652–1655.

[37] Chang, S., Chae, S., Lee, S., Liu, C., Noh, T., Lee, J., Kahng, B., Jang, J., Kim, M., Kim, D.-W., et al. (2008). Effects of heat dissipation on unipolar resistance switching in pt/ ni o/ pt capacitors. *Applied Physics Letters*, 92(18):183507.

[38] Chang, T., Jo, S.-H., and Lu, W. (2011). Short-term memory to long-term memory transition in a nanoscale memristor. *ACS nano*, 5(9):7669–7676.

[39] Chase, S. M. and Young, E. D. (2006). Spike-timing codes enhance the representation of multiple simultaneous sound-localization cues in the inferior colliculus. *Journal of Neuroscience*, 26(15):3889–3898.

[40] Chen, A. (2016). A review of emerging non-volatile memory (nvm) technologies and applications. *Solid-State Electronics*, 125:25–38.

[41] Chen, L., Bang, W., Park, Y.-J., Ryan, E. T., King, S., and Kim, C.-U. (2010). Observation of space charge limited current by cu ion drift in porous low-k/cu interconnects. *Applied Physics Letters*, 96(9).

[42] Chen, L., Li, J., Chen, Y., Deng, Q., Shen, J., Liang, X., and Jiang, L. (2017). Accelerator-friendly neural-network training: Learning variations and defects in rram crossbar. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 19–24. IEEE.

[43] Chen, L., Zaharia, M., and Zou, J. (2023). How is chatgpt's behavior changing over time? *arXiv preprint arXiv:2307.09009*.

[44] Chen, P.-Y., Peng, X., and Yu, S. (2018a). Neurosim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(12):3067–3080.

[45] Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018b). Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

[46] Chen, W.-H., Li, K.-X., Lin, W.-Y., Hsu, K.-H., Li, P.-Y., Yang, C.-H., Xue, C.-X., Yang, E.-Y., Chen, Y.-K., Chang, Y.-S., et al. (2018c). A 65nm 1mb nonvolatile computing-in-memory reram macro with sub-16ns multiply-and-accumulate for binary dnn ai edge processors. In *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 494–496. IEEE.

[47] Cheney, N., Schrimpf, M., and Kreiman, G. (2017). On the robustness of convolutional neural networks to internal architecture and weight perturbations. *arXiv preprint arXiv:1703.08245*.

[48] Chicca, E. and Indiveri, G. (2020). A recipe for creating ideal hybrid memristive-cmos neuromorphic processing systems. *Applied Physics Letters*, 116(12):120501.

[49] Chua, L. (1971). Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, 18(5):507–519.

[50] Chua, L. (2019). Resistance switching memories are memristors. In *Handbook of memristor networks*, pages 197–230. Springer.

[51] Chua, L., Sbitnev, V., and Kim, H. (2012). Hodgkin–huxley axon is made of memristors. *International Journal of Bifurcation and Chaos*, 22(03):1230011.

[52] Chua, L. O. and Kang, S. M. (1976). Memristive devices and systems. *Proceedings of the IEEE*, 64(2):209–223.

[53] Ciocci, P., Lemineur, J.-F., Noël, J.-M., Combellas, C., and Kanoufi, F. (2021). Differentiating electrochemically active regions of indium tin oxide electrodes for hydrogen evolution and reductive decomposition reactions. an in situ optical microscopy approach. *Electrochimica Acta*, 386:138498.

[54] Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., and Bengio, Y. (2016). Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.

[55] Cowley, A. and Sze, S. (1965). Surface states and barrier height of metal-semiconductor systems. *Journal of Applied Physics*, 36(10):3212–3220.

[56] Cox, H. R., Buckwell, M., Ng, W. H., Mannion, D. J., Mehonic, A., Shearing, P. R., Fearn, S., and Kenyon, A. J. (2021). A nanoscale analysis method to reveal oxygen exchange between environment, oxide, and electrodes in reram devices. *APL Materials*, 9(11).

[57] Dalgaty, T., Payvand, M., Moro, F., Ly, D. R., Pebay-Peyroula, F., Casas, J., Indiveri, G., and Vianello, E. (2019). Hybrid neuromorphic circuits exploiting non-conventional properties of rram for massively parallel local plasticity mechanisms. *APL Materials*, 7(8):081125.

[58] Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27.

[59] Dayan, P. and Abbott, L. F. (2005). *Theoretical neuroscience: computational and mathematical modeling of neural systems*. MIT press.

[60] Deal, B. E. (1974). The current understanding of charges in the thermally oxidized silicon structure. *Journal of the Electrochemical Society*, 121(6):198C.

[61] Del Valle, J., Salev, P., Kalcheim, Y., and Schuller, I. K. (2020). A caloritronics-based mott neuristor. *Scientific reports*, 10(1):4292.

[62] Di Ventra, M., Pershin, Y. V., and Chua, L. O. (2009). Circuit elements with memory: memristors, memcapacitors, and meminductors. *Proceedings of the IEEE*, 97(10):1717–1724.

[63] Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., and Pfeiffer, M. (2015). Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International joint conference on neural networks (IJCNN)*, pages 1–8. ieee.

[64] DiMaria, D. and Cartier, E. (1995). Mechanism for stress-induced leakage currents in thin silicon dioxide films. *Journal of Applied physics*, 78(6):3883–3894.

[65] Donati, E., Payvand, M., Risi, N., Krause, R., and Indiveri, G. (2019). Discrimination of emg signals using a neuromorphic implementation of a spiking neural network. *IEEE transactions on biomedical circuits and systems*, 13(5):795–803.

[66] Du, C. (2017). *Metal Oxide Memristors with Internal Dynamics for Neuromorphic Applications*. PhD thesis.

[67] Duan, Q., Jing, Z., Zou, X., Wang, Y., Yang, K., Zhang, T., Wu, S., Huang, R., and Yang, Y. (2020). Spiking neurons with spatiotemporal dynamics and gain modulation for monolithically integrated memristive neural networks. *Nature communications*, 11(1):3399.

[68] Einstein, A. (1905). Über einem heuristics standpunkt zur produktion und transformation des lichts.

[69] El Kamel, F., Gonon, P., Ortéga, L., Jomni, F., and Yangui, B. (2006). Space charge limited transient currents and oxygen vacancy mobility in amorphous ba ti o 3 thin films. *Journal of applied physics*, 99(9):094107.

[70] El Kamel, F., Gonon, P., Yangui, B., and Jomni, F. (2007). Ionic and electronic defects in a-batio3 thin films studied by transient and steady state conductivity measurements. *physica status solidi c*, 4(3):1242–1245.

[71] El-Sayed, A.-M., Watkins, M. B., Shluger, A. L., and Afanas'ev, V. V. (2013). Identification of intrinsic electron trapping sites in bulk amorphous silica from ab initio calculations. *Microelectronic engineering*, 109:68–71.

[72] El Zein, A., McCreath, E., Rendell, A., and Smola, A. (2008). Performance evaluation of the nvidia geforce 8800 gtx gpu for machine learning. In *International Conference on Computational Science*, pages 466–475. Springer.

[73] Eliasmith, C. and Anderson, C. H. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press.

[74] Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., and Rasmussen, D. (2012). A large-scale model of the functioning brain. *science*, 338(6111):1202–1205.

[75] Eshraghian, J. K., Baek, S., Kim, J.-H., Iannella, N., Cho, K., Goo, Y. S., Iu, H. H., Kang, S.-M., and Eshraghian, K. (2018). Formulation and implementation of nonlinear integral equations to model neural dynamics within the vertebrate retina. *International Journal of Neural Systems*, 28(07):1850004.

[76] Eshraghian, J. K., Cho, K., and Kang, S. M. (2021). A 3-d reconfigurable rram crossbar inference engine. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

[77] Eshraghian, J. K., Kang, S.-M., Baek, S., Orchard, G., Iu, H. H.-C., and Lei, W. (2019). Analog weights in reram dnn accelerators. In *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 267–271. IEEE.

[78] Eshraghian, J. K., Wang, X., and Lu, W. D. (2022). Memristor-based binarized spiking neural networks: Challenges and applications. *IEEE Nanotechnology Magazine*, 16(2):14–23.

[79] Eshraghian, J. K., Ward, M., Neftci, E. O., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D. S., and Lu, W. D. (2023). Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*.

[80] Esser, S. K., Appuswamy, R., Merolla, P., Arthur, J. V., and Modha, D. S. (2015). Backpropagation for energy-efficient neuromorphic computing. *Advances in neural information processing systems*, 28.

[81] Fadley, C. S. (2010). X-ray photoelectron spectroscopy: Progress and perspectives. *Journal of Electron Spectroscopy and Related Phenomena*, 178:2–32.

[82] Fang, Y., Yu, Z., Wang, Z., Zhang, T., Yang, Y., Cai, Y., and Huang, R. (2018). Improvement of hfo x-based rram device variation by inserting ald tin buffer layer. *IEEE Electron Device Letters*, 39(6):819–822.

[83] Feurer, M. and Hutter, F. (2019). Hyperparameter optimization. *Automated machine learning: Methods, systems, challenges*, pages 3–33.

[84] Fischetti, M. (2011). Computers vs. brains. *Scientific American*, 305(5):104–104.

[85] Frenkel, C., Lefebvre, M., Legat, J.-D., and Bol, D. (2018). A 0.086-mm^2 12.7-pj/sop 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm cmos. *IEEE transactions on biomedical circuits and systems*, 13(1):145–158.

[86] Frenkel, C., Legat, J.-D., and Bol, D. (2019). Morphic: A 65-nm 738k-synapse/mm^2 quad-core binary-weight digital neuromorphic processor with stochastic spike-driven online learning. *IEEE transactions on biomedical circuits and systems*, 13(5):999–1010.

[87] Frenkel, J. (1938). On pre-breakdown phenomena in insulators and electronic semiconductors. *Physical Review*, 54(8):647.

[88] Fröhlich, F. (2016). Chapter 4 - synaptic plasticity. In Fröhlich, F., editor, *Network Neuroscience*, pages 47–58. Academic Press, San Diego.

[89] Fujii, T., Kawasaki, M., Sawa, A., Akoh, H., Kawazoe, Y., and Tokura, Y. (2005). Hysteretic current–voltage characteristics and resistance switching at an epitaxial oxide schottky junction srruo3/ srti0. 99nb0. 01o3. *Applied Physics Letters*, 86(1).

[90] Funahashi, K.-i. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806.

[91] Gaba, S., Sheridan, P., Zhou, J., Choi, S., and Lu, W. (2013). Stochastic memristive devices for computing and neuromorphic applications. *Nanoscale*, 5(13):5872–5878.

[92] Gao, D. Z., El-Sayed, A.-M., and Shluger, A. L. (2016). A mechanism for frenkel defect creation in amorphous sio2 facilitated by electron injection. *Nanotechnology*, 27(50):505207.

[93] Gaol, D., Zhang, G. L., Yin, X., Li, B., Schlichtmann, U., and Zhuo, C. (2021). Reliable memristor-based neuromorphic design using variation-and defect-aware training. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9. IEEE.

[94] Geng, S., Zhang, S., and Onishi, H. (2002). Xps applications in thin films research. *Materials Technology*, 17(4):234–240.

[95] Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

[96] Gerstner, W., Kreiter, A. K., Markram, H., and Herz, A. V. (1997). Neural codes: firing rates and beyond. *Proceedings of the National Academy of Sciences*, 94(24):12740–12741.

[97] Ghanbari, A., Malyshev, A., Volgushev, M., and Stevenson, I. H. (2017). Estimating short-term synaptic plasticity from pre-and postsynaptic spiking. *PLoS computational biology*, 13(9):e1005738.

[98] Ghibaudo, G., Riess, P., Bruyere, S., DeSalvo, B., Jahan, C., Scarpa, A., Pananakakis, G., and Vincent, E. (1999). Emerging oxide degradation mechanisms: stress induced leakage current (silc) and quasi-breakdown (qb). *Microelectronic engineering*, 49(1-2):41–50.

[99] Goodman, D. F. and Brette, R. (2008). Brian: a simulator for spiking neural networks in python. *Frontiers in neuroinformatics*, page 5.

[100] Goux, L. and Valov, I. (2016). Electrochemical processes and device improvement in conductive bridge ram cells. *physica status solidi (a)*, 213(2):274–288.

[101] Guerguiev, J., Lillicrap, T. P., and Richards, B. A. (2017). Towards deep learning with segregated dendrites. *Elife*, 6:e22901.

[102] Gupta, S. and Babu, M. R. (2011). Performance analysis of gpu compared to single-core and multi-core cpu for natural language applications. *Advanced Computer Science and Applications*, 2.

[103] Gütig, R. (2014). To spike, or when to spike? *Current opinion in neurobiology*, 25:134–139.

[104] Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A., and Dally, W. J. (2016). Eie: Efficient inference engine on compressed deep neural network. *ACM SIGARCH Computer Architecture News*, 44(3):243–254.

[105] Han, S., Mao, H., and Dally, W. J. (2015a). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

[106] Han, S., Pool, J., Tran, J., and Dally, W. (2015b). Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

[107] Hansen, M., Ziegler, M., Kolberg, L., Soni, R., Dirkmann, S., Mussenbrock, T., and Kohlstedt, H. (2015). A double barrier memristive device. *Scientific reports*, 5(1):13753.

[108] Hao, S., Ji, X., Zhong, S., Pang, K. Y., Lim, K. G., Chong, T. C., and Zhao, R. (2020). A monolayer leaky integrate-and-fire neuron for 2d memristive neuromorphic networks. *Advanced Electronic Materials*, 6(4):1901335.

[109] Hardy, N. F. and Buonomano, D. V. (2018). Encoding time in feedforward trajectories of a recurrent neural network model. *Neural computation*, 30(2):378–396.

[110] Hasan, R., Taha, T. M., and Yakopcic, C. (2017). On-chip training of memristor crossbar based multi-layer neural networks. *Microelectronics journal*, 66:31–40.

[111] Hasan, R., Taha, T. M., Yakopcic, C., and Mountain, D. J. (2016). High throughput neural network based embedded streaming multicore processors. In *2016 IEEE International Conference on Rebooting Computing (ICRC)*, pages 1–8. IEEE.

[112] Hassan, A. M., Khalaf, A. F., Sayed, K. S., Li, H. H., and Chen, Y. (2018). Real-time cardiac arrhythmia classification using memristor neuromorphic computing system. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2567–2570. IEEE.

[113] Hayakawa, Y., Himeno, A., Yasuhara, R., Boullart, W., Vecchio, E., Vandeweyer, T., Witters, T., Crotti, D., Jurczak, M., Fujii, S., et al. (2015). Highly reliable tao x reram with centralized filament for 28-nm embedded application. In *2015 Symposium on VLSI Technology (VLSI Technology)*, pages T14–T15. IEEE.

[114] Hazan, H., Saunders, D. J., Khan, H., Patel, D., Sanghavi, D. T., Siegelmann, H. T., and Kozma, R. (2018). Bindsnet: A machine learning-oriented spiking neural networks library in python. *Frontiers in neuroinformatics*, 12:89.

[115] He, Z., Lin, J., Ewetz, R., Yuan, J.-S., and Fan, D. (2019). Noise injection adaption: End-to-end reram crossbar non-ideal effect adaption for neural network mapping. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6.

[116] Hebb, D. (2002). The organization of behavior. 1949. *New York Wiely*, 2(7):8.

[117] Hemanth, D. J., Vijila, C. K. S., Selvakumar, A. I., and Anitha, J. (2014). Performance improved iteration-free artificial neural networks for abnormal magnetic resonance brain image classification. *Neurocomputing*, 130:98–107.

[118] Hertz, H. (1887). On the influence of ultraviolet light on electrical discharge. *Annals of Physics*, 267(8):983–1000.

[119] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

[120] Hinton, G. E. and Sejnowski, T. J. (1983). Optimal perceptual inference. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, volume 448, pages 448–453. Citeseer.

[121] Hochstetter, J., Zhu, R., Loeffler, A., Diaz-Alvarez, A., Nakayama, T., and Kuncic, Z. (2021). Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nature Communications*, 12(1):4008.

[122] Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500.

[123] Hofstein, S. R. (1967). Proton and sodium transport in sio 2 films. *IEEE Transactions on Electron Devices*, 14(11):749–759.

[124] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

[125] Houng, M. P., Wang, Y. H., and Chang, W. J. (1999). Current transport mechanism in trapped oxides: A generalized trap-assisted tunneling model. *Journal of applied physics*, 86(3):1488–1491.

[126] Hu, M., Strachan, J. P., Li, Z., Stanley, R., et al. (2016). Dot-product engine as computing memory to accelerate machine learning algorithms. In *2016 17th International Symposium on Quality Electronic Design (ISQED)*, pages 374–379. IEEE.

[127] Hu, S., Wu, S., Jia, W., Yu, Q., Deng, L., Fu, Y. Q., Liu, Y., and Chen, T. P. (2014). Review of nanostructured resistive switching memristor and its applications. *Nanoscience and Nanotechnology Letters*, 6(9):729–757.

[128] Huang, A., Zhang, X., Li, R., and Chi, Y. (2018). Memristor neural network design. *Memristor and Memristive Neural Networks*, pages 1–35.

[129] Huang, C., Li, K., Tu, G., and Wang, W. (2003). The electrochemical behavior of tin-doped indium oxide during reduction in 0.3 m hydrochloric acid. *Electrochimica Acta*, 48(24):3599–3605.

[130] Huang, J., Stathopoulos, S., Serb, A., and Prodromakis, T. (2022). Neuropack: An algorithm-level python-based simulator for memristor-empowered neuro-inspired computing. *Frontiers in Nanotechnology*, 4:851856.

[131] Huang, L., Diao, J., Teng, S., Li, Z., Wang, W., Liu, S., Li, M., and Liu, H. (2021). A method for obtaining highly robust memristor based binarized convolutional neural network. In *INTERNATIONAL CONFERENCE ON WIRELESS COMMUNICATIONS, NETWORKING AND APPLICATIONS*, pages 813–822. Springer.

[132] Huang, P., Li, Z., Dong, Z., Han, R., Zhou, Z., Zhu, D., Liu, L., Liu, X., and Kang, J. (2019). Binary resistive-switching-device-based electronic synapse with spike-rate-dependent plasticity for online learning. *ACS Applied Electronic Materials*, 1(6):845–853.

[133] Huh, D. and Sejnowski, T. J. (2018). Gradient descent for spiking neural networks. *Advances in neural information processing systems*, 31.

[134] Hunsberger, E., Scott, M., and Eliasmith, C. (2014). The competing benefits of noise and heterogeneity in neural coding. *Neural computation*, 26(8):1600–1623.

[135] Ielmini, D., Nardi, F., and Cagli, C. (2010). Resistance-dependent amplitude of random telegraph-signal noise in resistive switching memories. *Applied Physics Letters*, 96(5):053503.

[136] Ielmini, D., Spinelli, A. S., Rigamonti, M. A., and Lacaita, A. L. (2000). Modeling of silc based on electron and hole tunneling. ii. steady-state. *IEEE Transactions on Electron Devices*, 47(6):1266–1272.

[137] Indiveri, G. (2021). Introducing 'neuromorphic computing and engineering'. *Neuromorphic Computing and Engineering*, 1(1):010401.

[138] Indiveri, G. and Liu, S.-C. (2015). Memory and information processing in neuromorphic systems. *Proceedings of the IEEE*, 103(8):1379–1397.

[139] Jain, A. K., Mao, J., and Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, 29(3):31–44.

[140] Jain, S. and Raghunathan, A. (2019). Cxdnn: Hardware-software compensation methods for deep neural networks on resistive crossbar systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(6):1–23.

[141] Jeong, D. S., Schroeder, H., and Waser, R. (2007). Coexistence of bipolar and unipolar resistive switching behaviors in a pt/ tio2/ pt stack. *Electrochemical and solid-state letters*, 10(8):G51.

[142] Ji, S., Xu, W., Yang, M., and Yu, K. (2012). 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231.

[143] Jiménez-Molinos, F., Palma, A., Gámiz, F., Banqueri, J., and López-Villanueva, J. (2001). Physical model for trap-assisted inelastic tunneling in metal-oxide-semiconductor structures. *Journal of Applied Physics*, 90(7):3396–3404.

[144] Jo, S. H., Chang, T., Ebong, I., Bhadviya, B. B., Mazumder, P., and Lu, W. (2010). Nanoscale memristor device as synapse in neuromorphic systems. *Nano letters*, 10(4):1297–1301.

[145] Jo, S. H., Kim, K.-H., and Lu, W. (2009). Programmable resistance switching in nanoscale two-terminal devices. *Nano letters*, 9(1):496–500.

[146] Johnston, D., Magee, J. C., Colbert, C. M., and Christie, B. R. (1996). Active properties of neuronal dendrites. *Annual review of neuroscience*, 19(1):165–186.

[147] Joksas, D. (2022). *Memristive crossbars as hardware accelerators: modelling, design and new uses*. PhD thesis, UCL (University College London).

[148] Joksas, D., Freitas, P., Chai, Z., Ng, W. H., Buckwell, M., Li, C., Zhang, W., Xia, Q., Kenyon, A., and Mehonic, A. (2020). Committee machines—a universal method to deal with non-idealities in memristor-based neural networks. *Nature communications*, 11(1):4273.

[149] Joksas, D., Wang, E., Barmpatsalos, N., Ng, W. H., Kenyon, A. J., Constantinides, G. A., and Mehonic, A. (2022). Nonideality-aware training for accurate and robust low-power memristive neural networks. *Advanced Science*, 9(17):2105784.

[150] Jones, M. C. (1993). Simple boundary correction for kernel density estimation. *Statistics and computing*, 3:135–146.

[151] Joshi, V., Le Gallo, M., Haefeli, S., Boybat, I., Nandakumar, S. R., Piveteau, C., Dazzi, M., Rajendran, B., Sebastian, A., and Eleftheriou, E. (2020). Accurate deep neural network inference using computational phase-change memory. *Nature communications*, 11(1):2473.

[152] Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., et al. (2017). In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12.

[153] Kaiser, J., Mostafa, H., and Neftci, E. (2020). Synaptic plasticity dynamics for deep continuous local learning (decolle). *Frontiers in Neuroscience*, 14:424.

[154] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., Mack, S., et al. (2000). *Principles of neural science*, volume 4. McGraw-hill New York.

[155] Kang, D.-H., Jun, H.-G., Ryoo, K.-C., Jeong, H., and Sohn, H. (2015). Emulation of spike-timing dependent plasticity in nano-scale phase change memory. *Neurocomputing*, 155:153–158.

[156] Kang, S. M., Choi, D., Eshraghian, J. K., Zhou, P., Kim, J., Kong, B.-S., Zhu, X., Demirkol, A. S., Ascoli, A., Tetzlaff, R., et al. (2021). How to build a memristive integrate-and-fire model for spiking neuronal signal generation. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(12):4837–4850.

[157] Kappel, D., Nessler, B., and Maass, W. (2014). Stdp installs in winner-take-all circuits an online approximation to hidden markov model learning. *PLoS computational biology*, 10(3):e1003511.

[158] Karpathy, A. (2011). Lessons learned from manually classifying cifar-10. *Published online at http://karpathy. github. io/2011/04/27/manually-classifying-cifar10*.

[159] Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

[160] Kasabov, N. K. (2014). Neucube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data. *Neural Networks*, 52:62–76.

[161] Kendall, J., Pantone, R., Manickavasagam, K., Bengio, Y., and Scellier, B. (2020). Training end-to-end analog neural networks with equilibrium propagation. *arXiv preprint arXiv:2006.01981*.

[162] Kenyon, A. J., Munde, M. S., Ng, W. H., Buckwell, M., Joksas, D., and Mehonic, A. (2019). The interplay between structure and function in redox-based resistance switching. *Faraday discussions*, 213:151–163.

[163] Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., and Masquelier, T. (2018). Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67.

[164] Kiani, F., Yin, J., Wang, Z., Yang, J. J., and Xia, Q. (2021). A fully hardware-based memristive multilayer neural network. *Science advances*, 7(48):eabj4801.

[165] Kim, D., Won, Y., Cha, J., Yoon, S., Choi, J., and Kang, S. (2015). Exploiting compression-induced internal fragmentation for power-off recovery in ssd. *IEEE Transactions on Computers*, 65(6):1720–1733.

[166] Kim, H., Mahmoodi, M., Nili, H., and Strukov, D. B. (2021). 4k-memristor analog-grade passive crossbar circuit. *Nature communications*, 12(1):5198.

[167] Kim, K. M., Jeong, D. S., and Hwang, C. S. (2011). Nanofilamentary resistive switching in binary oxide system; a review on the present status and outlook. *Nanotechnology*, 22(25):254002.

[168] Kim, K. M., Yang, J. J., Strachan, J. P., Grafals, E. M., Ge, N., Melendez, N. D., Li, Z., and Williams, R. S. (2016). Voltage divider effect for the improvement of variability and endurance of taox memristor. *Scientific reports*, 6(1):20085.

[169] Kline, D. M. and Berardi, V. L. (2005). Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing & Applications*, 14:310–318.

[170] Knight, B. W. (1972). Dynamics of encoding in a population of neurons. *The Journal of general physiology*, 59(6):734–766.

[171] Koch, C. (2004). *Biophysics of computation: information processing in single neurons*. Oxford university press.

[172] Kozicki, M. N., Mitkova, M., and Valov, I. (2016). Electrochemical metallization memories. *Resistive switching: from fundamentals of nanoionic redox processes to memristive device applications*, pages 483–514.

[173] Krestinskaya, O., Salama, K. N., and James, A. P. (2018). Learning in memristive neural network architectures using analog backpropagation circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66(2):719–732.

[174] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

[175] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

[176] Krzysteczko, P., Münchenberger, J., Schäfers, M., Reiss, G., and Thomas, A. (2012). The memristive magnetic tunnel junction as a nanoscopic synapse-neuron system. *Advanced Materials*, 24(6):762–766.

[177] Ku, Y., Tompkins, W., and Xue, Q. (1992). Artificial neural network for ecg arrhythmia monitoring. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, volume 2, pages 987–992. IEEE.

[178] Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (1998). Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, 9(1):112–147.

[179] Lammie, C. and Azghadi, M. R. (2019). Stochastic computing for low-power and high-speed deep learning on fpga. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

[180] Lammie, C., Krestinskaya, O., James, A., and Azghadi, M. R. (2019a). Variation-aware binarized memristive networks. In *2019 26th IEEE international conference on electronics, circuits and systems (ICECS)*, pages 490–493. IEEE.

[181] Lammie, C., Olsen, A., Carrick, T., and Azghadi, M. R. (2019b). Low-power and high-speed deep fpga inference engines for weed classification at the edge. *IEEE Access*, 7:51171–51184.

[182] Lammie, C., Xiang, W., and Azghadi, M. R. (2020). Training progressively binarizing deep networks using fpgas. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

[183] Lammie, C., Xiang, W., Linares-Barranco, B., and Azghadi, M. R. (2022). Memtorch: An open-source simulation framework for memristive deep learning systems. *Neurocomputing*, 485:124–133.

[184] Lampert, M. A. and Schilling, R. B. (1970). Current injection in solids: The regional approximation method. In *Semiconductors and semimetals*, volume 6, pages 1–96. Elsevier.

[185] Lapicque, L. É. (1907). Louis lapicque. *J. physiol*, 9:620–635.

[186] Lapique, L. (1907). Researches quantatives sur l'excitation electrique des nerfs traitee comme une polarization. *Journal of Physiology, Pathology and Genetics*, 9:620–635.

[187] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[188] Lee, H., Chen, Y., Chen, P., Gu, P., Hsu, Y., Wang, S., Liu, W., Tsai, C., Sheu, S., Chiang, P., et al. (2010). Evidence and solution of over-reset problem for hfo x based resistive memory with sub-ns switching speed and high endurance. In *2010 International Electron Devices Meeting*, pages 19–7. IEEE.

[189] Lee, H.-S., Park, S., and Lee, D.-H. (2013). Rmss: an efficient recovery management scheme on nand flash memory based solid state disk. *IEEE Transactions on Consumer Electronics*, 59(1):107–112.

[190] Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10:508.

[191] Lenat, D. B. (1983). Eurisko: a program that learns new heuristics and domain concepts: the nature of heuristics iii: program design and results. *Artificial intelligence*, 21(1-2):61–98.

[192] Lentz, F., Roesgen, B., Rana, V., Wouters, D. J., and Waser, R. (2013). Current compliance-dependent nonlinearity in tiox reram. *IEEE electron device letters*, 34(8):996–998.

[193] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867.

[194] Li, B., Yan, B., Liu, C., and Li, H. (2019). Build reliable and efficient neuromorphic design with memristor technology. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, pages 224–229.

[195] Li, C., Han, L., Jiang, H., Jang, M.-H., Lin, P., Wu, Q., Barnell, M., Yang, J. J., Xin, H. L., and Xia, Q. (2017a). Three-dimensional crossbar arrays of self-rectifying si/sio2/si memristors. *Nature communications*, 8(1):15666.

[196] Li, C., Hu, M., Li, Y., Jiang, H., Ge, N., Montgomery, E., Zhang, J., Song, W., Dávila, N., Graves, C. E., et al. (2018). Analogue signal and image processing with large memristor crossbars. *Nature electronics*, 1(1):52–59.

[197] Li, J., Xu, H., Sun, S.-Y., Liu, S., Li, N., Li, Q., Liu, H., and Li, Z. (2020). Enhanced spiking neural network with forgetting phenomenon based on electronic synaptic devices. *Neurocomputing*, 408:21–30.

[198] Li, Q., Khiat, A., Salaoru, I., Xu, H., and Prodromakis, T. (2014). Stochastic switching of tio 2-based memristive devices with identical initial memory states. *Nanoscale research letters*, 9:1–5.

[199] Li, Y. (2018). Analog computing using 1t1r crossbar arrays.

[200] Li, Y., Lei, Y., Shen, B., and Sun, J. (2015). Visible-light-accelerated oxygen vacancy migration in strontium titanate. *Scientific reports*, 5(1):14576.

[201] Li, Y., Long, S., Liu, Q., Lv, H., and Liu, M. (2017b). Resistive switching performance improvement via modulating nanoscale conductive filament, involving the application of two-dimensional layered materials. *Small*, 13(35):1604306.

[202] Li, Y., Zhong, Y., Xu, L., Zhang, J., Xu, X., Sun, H., and Miao, X. (2013). Ultrafast synaptic events in a chalcogenide memristor. *Scientific reports*, 3(1):1619.

[203] Lifshitz, N. and Smolinsky, G. (1989). Detection of water-related charge in electronic dielectrics. *Applied physics letters*, 55(4):408–410.

[204] Lillicrap, T. P., Cownden, D., Tweed, D. B., and Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):13276.

[205] Lim, H., Kornijcuk, V., Seok, J. Y., Kim, S. K., Kim, I., Hwang, C. S., and Jeong, D. S. (2015). Reliability of neuronal information conveyed by unreliable neuristor-based leaky integrate-and-fire neurons: a model study. *Scientific reports*, 5(1):9776.

[206] Lin, C.-Y., Chen, J., Chen, P.-H., Chang, T.-C., Wu, Y., Eshraghian, J. K., Moon, J., Yoo, S., Wang, Y.-H., Chen, W.-C., et al. (2020). Adaptive synaptic memory via lithium ion modulation in rram devices. *Small*, 16(42):2003964.

[207] Liu, C., Hu, M., Strachan, J. P., and Li, H. (2017). Rescuing memristor-based neuromorphic design with high defects. In *Proceedings of the 54th Annual Design Automation Conference 2017*, pages 1–6.

[208] Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528.

[209] Liu, J.-C., Hsu, C.-W., Wang, I.-T., and Hou, T.-H. (2015a). Categorization of multilevel-cell storage-class memory: an rram example. *IEEE Transactions on Electron Devices*, 62(8):2510–2516.

[210] Liu, L., Yellinek, S., Valdinger, I., Donval, A., and Mandler, D. (2015b). Important implications of the electrochemical reduction of ito. *Electrochimica Acta*, 176:1374–1381.

[211] Liu, Q., Long, S., Lv, H., Wang, W., Niu, J., Huo, Z., Chen, J., and Liu, M. (2010). Controllable growth of nanoscale conductive filaments in solid-electrolyte-based reram by using a metal nanocrystal covered bottom electrode. *ACS nano*, 4(10):6162–6168.

[212] Loeffler, A., Zhu, R., Hochstetter, J., Diaz-Alvarez, A., Nakayama, T., Shine, J. M., and Kuncic, Z. (2021). Modularity and multitasking in neuro-memristive reservoir networks. *Neuromorphic Computing and Engineering*, 1(1):014003.

[213] Lukoševičius, M. and Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149.

[214] Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671.

[215] Madams, C., Morgan, D., and Howes, M. (1974). Migration of gold atoms through thin silicon oxide films. *Journal of Applied Physics*, 45(11):5088–5090.

[216] Mainen, Z. F. and Sejnowski, T. J. (1995). Reliability of spike timing in neocortical neurons. *Science*, 268(5216):1503–1506.

[217] Majumdar, A., Cadambi, S., Becchi, M., Chakradhar, S. T., and Graf, H. P. (2012). A massively parallel, energy efficient programmable accelerator for learning and classification. *ACM Transactions on Architecture and Code Optimization (TACO)*, 9(1):1–30.

[218] Malik, P. (2013). Governing big data: principles and practices. *IBM Journal of Research and Development*, 57(3/4):1–1.

[219] Manceau, J.-P., Bruyere, S., Jeannot, S., Sylvestre, A., and Gonon, P. (2007a). Current instability, permittivity variation with frequency, and their relationship in capacitor. *IEEE transactions on device and materials reliability*, 7(2):315–323.

[220] Manceau, J.-P., Bruyere, S., Jeannot, S., Sylvestre, A., and Gonon, P. (2007b). Metal-insulator-metal capacitors' current instability improvement using dielectric stacks to prevent oxygen vacancies formation. *Applied Physics Letters*, 91(13).

[221] Mandal, S., El-Amin, A., Alexander, K., Rajendran, B., and Jha, R. (2014). Novel synaptic memory device for neuromorphic computing. *Scientific reports*, 4(1):5333.

[222] Mannion, D. J. (2022). *Current transient phenomena in silicon oxide resistance switching oxides: characterisation and computational applications*. PhD thesis, UCL (University College London).

[223] Mannocci, P., Farronato, M., Lepri, N., Cattaneo, L., Glukhov, A., Sun, Z., and Ielmini, D. (2023). In-memory computing with emerging memory devices: Status and outlook. *APL Machine Learning*, 1(1).

[224] Many, A. and Rakavy, G. (1962). Theory of transient space-charge-limited currents in solids in the presence of trapping. *Physical Review*, 126(6):1980.

[225] Maranhão, G. and Guimarães, J. G. (2021). Low-power hybrid memristor-cmos spiking neuromorphic stdp learning system. *IET Circuits, Devices & Systems*, 15(3):237–250.

[226] Marder, E. and Goaillard, J.-M. (2006). Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience*, 7(7):563–574.

[227] Margaritondo, G. (1988). 100 years of photoemission. *Physics Today*, 41(4):66–72.

[228] Markram, H., Gerstner, W., and Sjöström, P. J. (2012). Spike-timing-dependent plasticity: a comprehensive overview. *Frontiers in synaptic neuroscience*, 4:2.

[229] Markram, H., Muller, E., Ramaswamy, S., Reimann, M. W., Abdellah, M., Sanchez, C. A., Ailamaki, A., Alonso-Nanclares, L., Antille, N., Arsever, S., et al. (2015). Reconstruction and simulation of neocortical microcircuitry. *Cell*, 163(2):456–492.

[230] Martens, J. et al. (2010). Deep learning via hessian-free optimization. In *Icml*, volume 27, pages 735–742.

[231] Marvin, M. and Seymour, A. P. (1969). Perceptrons. *Cambridge, MA: MIT Press*, 6:318–362.

[232] Matsugu, M., Mori, K., Mitari, Y., and Kaneda, Y. (2003). Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, 16(5-6):555–559.

[233] McCarthy, J., Minsky, M. L., Rochester, N., and Shannon, C. E. (2006). A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4):12–12.

[234] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.

[235] McKennoch, S., Liu, D., and Bushnell, L. G. (2006). Fast modifications of the spikeprop algorithm. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 3970–3977. IEEE.

[236] McKinney, S. M., Sieniek, M., Godbole, V., Godwin, J., Antropova, N., Ashrafian, H., Back, T., Chesus, M., Corrado, G. S., Darzi, A., et al. (2020). International evaluation of an ai system for breast cancer screening. *Nature*, 577(7788):89–94.

[237] Mead, C. (1990). Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636.

[238] Mead, C. and Ismail, M. (1989). *Analog VLSI implementation of neural systems*, volume 80. Springer Science & Business Media.

[239] Mehonic, A., Buckwell, M., Montesi, L., Garnett, L., Hudziak, S., Fearn, S., Chater, R., McPhail, D., and Kenyon, A. J. (2015). Structural changes and conductance thresholds in metal-free intrinsic siox resistive random access memory. *Journal of Applied Physics*, 117(12).

[240] Mehonic, A., Cueff, S., Wojdak, M., Hudziak, S., Jambois, O., Labbé, C., Garrido, B., Rizk, R., and Kenyon, A. J. (2012). Resistive switching in silicon suboxide films. *Journal of Applied Physics*, 111(7):074507.

[241] Mehonic, A., Joksas, D., Ng, W. H., Buckwell, M., and Kenyon, A. J. (2019). Simulation of inference accuracy using realistic rram devices. *Frontiers in neuroscience*, 13:593.

[242] Mehonic, A., Munde, M. S., Ng, W., Buckwell, M., Montesi, L., Bosman, M., Shluger, A., and Kenyon, A. (2017). Intrinsic resistance switching in amorphous silicon oxide for high performance siox reram devices. *Microelectronic Engineering*, 178:98–103.

[243] Mehonic, A., Shluger, A. L., Gao, D., Valov, I., Miranda, E., Ielmini, D., Bricalli, A., Ambrosi, E., Li, C., Yang, J. J., et al. (2018). Silicon oxide (siox): A promising material for resistance switching? *Advanced materials*, 30(43):1801187.

[244] Mel, B. W. (1994). Information processing in dendritic trees. *Neural computation*, 6(6):1031–1085.

[245] Menke, T., Meuffels, P., Dittmann, R., Szot, K., and Waser, R. (2009). Separation of bulk and interface contributions to electroforming and resistive switching behavior of epitaxial fe-doped srtio 3.

[246] Merced-Grafals, E. J., Dávila, N., Ge, N., Williams, R. S., and Strachan, J. P. (2016). Repeatable, accurate, and high speed multi-level programming of memristor 1t1r arrays for power efficient analog computing applications. *Nanotechnology*, 27(36):365202.

[247] Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., Jackson, B. L., Imam, N., Guo, C., Nakamura, Y., et al. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673.

[248] Meyer, R., Liedtke, R., and Waser, R. (2005). Oxygen vacancy migration and time-dependent leakage current behavior of ba0. 3sr0. 7tio3 thin films. *Applied Physics Letters*, 86(11).

[249] Mikolov, T., Kombrink, S., Burget, L., Černockỳ, J., and Khudanpur, S. (2011). Extensions of recurrent neural network language model. In *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5528–5531. IEEE.

[250] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

[251] Mok, A. (2023). Chatgpt could cost over 700,000 usd per day to operate. *Microsoft is reportedly trying to make it cheaper*, 2.

[252] Molas, G. and Nowak, E. (2021). Advances in emerging memory technologies: From data storage to artificial intelligence. *Applied Sciences*, 11(23):11254.

[253] Molter, T. W. and Nugent, M. A. (2016). The generalized metastable switch memristor model. In *CNNA 2016; 15th International workshop on cellular nanoscale networks and their applications*, pages 1–2. VDE.

[254] Moon, K., Lim, S., Park, J., Sung, C., Oh, S., Woo, J., Lee, J., and Hwang, H. (2019). Rram-based synapse devices for neuromorphic systems. *Faraday discussions*, 213:421–451.

[255] Moore, G. E. (2006). Lithography and the future of moore's law. *IEEE Solid-State Circuits Society Newsletter*, 11(3):37–42.

[256] Mostafa, H. (2017). Supervised learning based on temporal coding in spiking neural networks. *IEEE transactions on neural networks and learning systems*, 29(7):3227–3235.

[257] Mulaosmanovic, H., Breyer, E. T., Dünkel, S., Beyer, S., Mikolajick, T., and Slesazeck, S. (2021). Ferroelectric field-effect transistors based on hfo2: a review. *Nanotechnology*, 32(50):502002.

[258] Munde, M., Mehonic, A., Ng, W., Buckwell, M., Montesi, L., Bosman, M., Shluger, A., and Kenyon, A. (2017). Intrinsic resistance switching in amorphous silicon suboxides: the role of columnar microstructure. *Scientific reports*, 7(1):9274.

[259] Nageswaran, J. M., Dutt, N., Krichmar, J. L., Nicolau, A., and Veidenbaum, A. V. (2009). A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors. *Neural networks*, 22(5-6):791–800.

[260] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

[261] Neftci, E., Das, S., Pedroni, B., Kreutz-Delgado, K., and Cauwenberghs, G. (2014). Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in neuroscience*, 7:272.

[262] Neftci, E. O., Mostafa, H., and Zenke, F. (2019). Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63.

[263] Nessler, B., Pfeiffer, M., Buesing, L., and Maass, W. (2013). Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLoS computational biology*, 9(4):e1003037.

[264] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A. Y., et al. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 7. Granada, Spain.

[265] Noble, D. (1962). A modification of the hodgkin—huxley equations applicable to purkinje fibre action and pacemaker potentials. *The Journal of physiology*, 160(2):317.

[266] Nordling, C., Sokolowski, E., and Siegbahn, K. (1957). Precision method for obtaining absolute values of atomic binding energies. *Physical Review*, 105(5):1676.

[267] Nurse, E., Mashford, B. S., Yepes, A. J., Kiral-Kornek, I., Harrer, S., and Freestone, D. R. (2016). Decoding eeg and lfp signals using deep learning: heading truenorth. In *Proceedings of the ACM international conference on computing frontiers*, pages 259–266.

[268] O'Connor, P., Neil, D., Liu, S.-C., Delbruck, T., and Pfeiffer, M. (2013). Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in neuroscience*, 7:178.

[269] Ohno-Machado, L. and Bialek, D. (1998). Diagnosing breast cancer from fnas: variable relevance in neural network and logistic regression models. In *MEDINFO'98*, pages 537–540. IOS Press.

[270] Oswald, S. (2006). X-ray photoelectron spectroscopy in analysis of surfaces. *Encyclopedia of Analytical Chemistry: Applications, Theory and Instrumentation*.

[271] Pal, A. and Singh, D. (2010). Handwritten english character recognition using neural network. *International Journal of Computer Science & Communication*, 1(2):141–144.

[272] Payvand, M., Demirag, Y., Dalgaty, T., Vianello, E., and Indiveri, G. (2020). Analog weight updates with compliance current modulation of binary rerams for on-chip learning. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

[273] Payvand, M. and Indiveri, G. (2019). Spike-based plasticity circuits for always-on on-line learning in neuromorphic systems. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

[274] Pehle, C.-G. and Egholm Pedersen, J. (2021). Norse-a deep learning library for spiking neural networks. *Zenodo*.

[275] Peláez, J. I., Doña, J. M., Fornari, J. F., and Serra, G. (2014). Ischemia classification via ecg using mlp neural networks. *International Journal of Computational Intelligence Systems*, 7(2):344–352.

[276] Peng, C.-N., Wang, C.-W., Chan, T.-C., Chang, W.-Y., Wang, Y.-C., Tsai, H.-W., Wu, W.-W., Chen, L.-J., and Chueh, Y.-L. (2012). Resistive switching of au/zno/au resistive memory: an in situ observation of conductive bridge formation. *Nanoscale research letters*, 7:1–6.

[277] Pérez, E., Grossi, A., Zambelli, C., Olivo, P., and Wenger, C. (2016). Impact of the incremental programming algorithm on the filament conduction in hfo 2-based rram arrays. *IEEE Journal of the Electron Devices Society*, 5(1):64–68.

[278] Perez-Carrasco, J.-A., Serrano, C., Acha, B., Serrano-Gotarredona, T., and Linares-Barranco, B. (2010). Spike-based convolutional network for real-time processing. In *2010 20th International Conference on Pattern Recognition*, pages 3085–3088. IEEE.

[279] Pfeiffer, M. and Pfeil, T. (2018). Deep learning with spiking neurons: opportunities and challenges. *Frontiers in neuroscience*, page 774.

[280] Pi, S., Li, C., Jiang, H., Xia, W., Xin, H., Yang, J. J., and Xia, Q. (2019). Memristor crossbar arrays with 6-nm half-pitch and 2-nm critical dimension. *Nature nanotechnology*, 14(1):35–39.

[281] Pickett, M. D., Medeiros-Ribeiro, G., and Williams, R. S. (2013). A scalable neuristor built with mott memristors. *Nature materials*, 12(2):114–117.

[282] Polsky, A., Mel, B. W., and Schiller, J. (2004). Computational subunits in thin dendrites of pyramidal cells. *Nature neuroscience*, 7(6):621–627.

[283] Pomerat, J., Segev, A., and Datta, R. (2019). On neural network activation functions and optimizers in relation to polynomial regression. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 6183–6185. IEEE.

[284] Poole, B., Sohl-Dickstein, J., and Ganguli, S. (2014). Analyzing noise in autoencoders and deep networks. *arXiv preprint arXiv:1406.1831*.

[285] Prakash, A. and Hwang, H. (2016). Multilevel cell storage and resistance variability in resistive random access memory. *Physical Sciences Reviews*, 1(6):20160010.

[286] Puglisi, F. M. and Pavan, P. (2016). Guidelines for a reliable analysis of random telegraph noise in electronic devices. *IEEE Transactions on Instrumentation and Measurement*, 65(6):1435–1442.

[287] Qiao, N., Mostafa, H., Corradi, F., Osswald, M., Stefanini, F., Sumislawska, D., and Indiveri, G. (2015). A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses. *Frontiers in neuroscience*, 9:141.

[288] Querlioz, D., Bichler, O., Dollfus, P., and Gamrat, C. (2013). Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE transactions on nanotechnology*, 12(3):288–295.

[289] Querlioz, D., Zhao, W., Dollfus, P., Klein, J.-O., Bichler, O., and Gamrat, C. (2012). Bioinspired networks with nanoscale memristive devices that combine the unsupervised and supervised learning approaches. In *Proceedings of the 2012 IEEE/ACM International Symposium on Nanoscale Architectures*, pages 203–210.

[290] Raghunandan, A. and Shilpa, D. (2019). Design of high-speed hybrid full adders using finfet 18nm technology. In *2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, pages 410–415. IEEE.

[291] Rahimi Azghadi, M., Chen, Y.-C., Eshraghian, J. K., Chen, J., Lin, C.-Y., Amirsoleimani, A., Mehonic, A., Kenyon, A. J., Fowler, B., Lee, J. C., et al. (2020). Complementary metal-oxide semiconductor and memristive hardware for neuromorphic computing. *Advanced Intelligent Systems*, 2(5):1900189.

[292] Rasch, M. J., Moreda, D., Gokmen, T., Le Gallo, M., Carta, F., Goldberg, C., El Maghraoui, K., Sebastian, A., and Narayanan, V. (2021). A flexible and fast pytorch toolkit for simulating training and inference on analog crossbar arrays. In *2021 IEEE 3rd international conference on artificial intelligence circuits and systems (AICAS)*, pages 1–4. IEEE.

[293] Regonini, D. (2008). *Anodised TiO2 nanotubes: synthesis, growth mechanism and thermal stability*. PhD thesis, University of Bath Bath, UK.

[294] Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

[295] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

[296] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252.

[297] Sacramento, J., Ponte Costa, R., Bengio, Y., and Senn, W. (2018). Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in neural information processing systems*, 31.

[298] Saha, S. and Krupanidhi, S. (2001). Transient analysis in al-doped barium strontium titanate thin films grown by pulsed laser deposition. *Journal of Applied Physics*, 90(3):1250–1254.

[299] Saïghi, S., Mayr, C. G., Serrano-Gotarredona, T., Schmidt, H., Lecerf, G., Tomas, J., Grollier, J., Boyn, S., Vincent, A. F., Querlioz, D., et al. (2015). Plasticity in memristive devices for spiking neural networks. *Frontiers in neuroscience*, 9:51.

[300] Salinas, E. and Abbott, L. (1994). Vector reconstruction from firing rates. *Journal of computational neuroscience*, 1(1):89–107.

[301] Sawa, A., Fujii, T., Kawasaki, M., and Tokura, Y. (2004). Hysteretic current–voltage characteristics and resistance switching at a rectifying ti/ pr0. 7ca0. 3mno3 interface. *Applied Physics Letters*, 85(18):4073–4075.

[302] Schemmel, J., Fieres, J., and Meier, K. (2008). Wafer-scale integration of analog neural networks. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 431–438. IEEE.

[303] Schlaf, R., Murata, H., and Kafafi, Z. (2001). Work function measurements on indium tin oxide films. *Journal of Electron Spectroscopy and Related Phenomena*, 120(1-3):149–154.

[304] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

[305] Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

[306] Sengupta, A., Al Azim, Z., Fong, X., and Roy, K. (2015). Spin-orbit torque induced spike-timing dependent plasticity. *Applied Physics Letters*, 106(9):093704.

[307] Senthilkumar, M., Mathiyarasu, J., Joseph, J., Phani, K., and Yegnaraman, V. (2008). Electrochemical instability of indium tin oxide (ito) glass in acidic ph range during cathodic polarization. *Materials Chemistry and Physics*, 108(2-3):403–407.

[308] Seong, D.-j., Jo, M., Lee, D., and Hwang, H. (2007). Hpha effect on reversible resistive switching of pt/ nb-doped srtio3 schottky junction for nonvolatile memory application. *Electrochemical and solid-state letters*, 10(6):H168.

[309] Serrano-Gotarredona, T., Masquelier, T., Prodromakis, T., Indiveri, G., and Linares-Barranco, B. (2013). Stdp and stdp variations with memristors for spiking neuromorphic learning systems. *Frontiers in neuroscience*, 7:2.

[310] Shannon, C. E. (1938). A symbolic analysis of relay and switching circuits. *Electrical Engineering*, 57(12):713–723.

[311] Sharifshazileh, M., Burelo, K., Fedele, T., Sarnthein, J., and Indiveri, G. (2019). A neuromorphic device for detecting high-frequency oscillations in human ieeg. In *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 69–72. IEEE.

[312] Shatz, C. J. (1992). The developing brain. *Scientific American*, 267(3):60–67.

[313] Shen, W., Kumar, S., and Kumar, S. (2021). Experimentally calibrated electro-thermal modeling of temperature dynamics in memristors. *Applied Physics Letters*, 118(10).

[314] Sheridan, P. M., Cai, F., Du, C., Ma, W., Zhang, Z., and Lu, W. D. (2017). Sparse coding with memristor networks. *Nature nanotechnology*, 12(8):784–789.

[315] Sidler, S., Boybat, I., Shelby, R. M., Narayanan, P., Jang, J., Fumarola, A., Moon, K., Leblebici, Y., Hwang, H., and Burr, G. W. (2016). Large-scale neural networks implemented with non-volatile memory as the synaptic weight element: Impact of conductance response. In *2016 46th European Solid-State Device Research Conference (ESSDERC)*, pages 440–443. Ieee.

[316] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.

[317] Silverman, B. W. (2018). *Density estimation for statistics and data analysis*. Routledge.

[318] Snow, E., Grove, A., Deal, B., and Sah, C. (1965). Ion transport phenomena in insulating films. *Journal of Applied Physics*, 36(5):1664–1673.

[319] Soni, R., Meuffels, P., Staikov, G., Weng, R., Kügeler, C., Petraru, A., Hambe, M., Waser, R., and Kohlstedt, H. (2011). On the stochastic nature of resistive switching in cu doped ge0. 3se0. 7 based memory devices. *Journal of applied physics*, 110(5).

[320] Sprekeler, H., Michaelis, C., and Wiskott, L. (2007). Slowness: An objective for spike-timing–dependent plasticity? *PLoS Computational Biology*, 3(6):e112.

[321] Squire, L., Berg, D., Bloom, F. E., Du Lac, S., Ghosh, A., and Spitzer, N. C. (2012). *Fundamental neuroscience*. Academic press.

[322] Stein, R. and Hodgkin, A. L. (1967). The frequency of nerve action potentials generated by applied currents. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 167(1006):64–86.

[323] Stocks, N. (2001). Information transmission in parallel threshold arrays: Suprathreshold stochastic resonance. *Physical Review E*, 63(4):041114.

[324] Stone, J. V. (2019). *Artificial intelligence engines: a tutorial introduction to the mathematics of deep learning*. Sebtel Press Warszawa, Poland.

[325] Stromatias, E. and Marsland, J. S. (2015). Supervised learning in spiking neural networks with limited precision: Snn/lp. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.

[326] Stromatias, E., Neil, D., Pfeiffer, M., Galluppi, F., Furber, S. B., and Liu, S.-C. (2015). Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms. *Frontiers in neuroscience*, 9:222.

[327] Strukov, D. B., Snider, G. S., Stewart, D. R., and Williams, R. S. (2008). The missing memristor found. *nature*, 453(7191):80–83.

[328] Sun, L., Zhang, Y., Hwang, G., Jiang, J., Kim, D., Eshete, Y. A., Zhao, R., and Yang, H. (2018). Synaptic computation enabled by joule heating of single-layered semiconductors for sound localization. *Nano letters*, 18(5):3229–3234.

[329] Sun, Z.-L., Wang, H., Lau, W.-S., Seet, G., and Wang, D. (2014). Application of bw-elm model on traffic sign recognition. *Neurocomputing*, 128:153–159.

[330] Sung, C., Lim, S., Kim, H., Kim, T., Moon, K., Song, J., Kim, J.-J., and Hwang, H. (2018). Effect of conductance linearity and multi-level cell characteristics of taox-based synapse device on pattern recognition accuracy of neuromorphic system. *Nanotechnology*, 29(11):115203.

[331] Suri, M., Querlioz, D., Bichler, O., Palma, G., Vianello, E., Vuillaume, D., Gamrat, C., and DeSalvo, B. (2013). Bio-inspired stochastic computing using binary cbram synapses. *IEEE Transactions on Electron Devices*, 60(7):2402–2409.

[332] Sze, S. M., Li, Y., and Ng, K. K. (2021). *Physics of semiconductor devices*. John wiley & sons.

[333] Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

[334] Taha, T. M., Hasan, R., Yakopcic, C., and McLean, M. R. (2013). Exploring the design space of specialized multicore neural processors. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[335] Tal, D. and Schwartz, E. L. (1997). Computing with the leaky integrate-and-fire neuron: logarithmic computation and multiplication. *Neural computation*, 9(2):305–318.

[336] Tang, Z., Chen, Y., Ye, S., Hu, R., Wang, H., He, J., Huang, Q., and Chang, S. (2020). Fully memristive spiking-neuron learning framework and its applications on pattern recognition and edge detection. *Neurocomputing*, 403:80–87.

[337] Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., and Maida, A. (2019). Deep learning in spiking neural networks. *Neural networks*, 111:47–63.

[338] Teeter, C., Iyer, R., Menon, V., Gouwens, N., Feng, D., Berg, J., Szafer, A., Cain, N., Zeng, H., Hawrylycz, M., et al. (2018). Generalized leaky integrate-and-fire models classify multiple neuron types. *Nature communications*, 9(1):709.

[339] Tien, N.-W. and Kerschensteiner, D. (2018). Homeostatic plasticity in neural development. *Neural development*, 13(1):1–7.

[340] Torralba, A., Fergus, R., and Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970.

[341] Toyoizumi, T., Pfister, J.-P., Aihara, K., and Gerstner, W. (2004). Spike-timing dependent plasticity and mutual information maximization for a spiking neuron model. *Advances in neural information processing systems*, 17.

[342] Tran, R., Li, X.-G., Montoya, J. H., Winston, D., Persson, K. A., and Ong, S. P. (2019). Anisotropic work function of elemental crystals. *Surface Science*, 687:48–55.

[343] Truong, S. N. and Min, K.-S. (2014). New memristor-based crossbar array architecture with 50-% area reduction and 48-% power saving for matrix-vector multiplication of analog neuromorphic computing. *JSTS: Journal of Semiconductor Technology and Science*, 14(3):356–363.

[344] Tuller, H. L. and Bishop, S. R. (2011). Point defects in oxides: tailoring materials through defect engineering. *Annual Review of Materials Research*, 41(1):369–398.

[345] Turing, A. (2004). On computable numbers, with an application to the entscheidungs problem, 1936. *The essential Turing: seminal writings in computing, logic, philosophy, artificial intelligence, and artificial life, plus the secrets of Enigma*, page 58.

[346] Turing, A. M. et al. (1936). On computable numbers, with an application to the entscheidungsproblem. *J. of Math*, 58(345-363):5.

[347] Turlach, B. A. (1993). Bandwidth selection in kernel density estimation: a rewiew. Technical report, Humboldt Universitaet Berlin.

[348] Turrigiano, G. G. (1999). Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. *Trends in neurosciences*, 22(5):221–227.

[349] Valentian, A., Rummens, F., Vianello, E., Mesquida, T., de Boissac, C. L.-M., Bichler, O., and Reita, C. (2019). Fully integrated spiking neural network with analog neurons and rram synapses. In *2019 IEEE International Electron Devices Meeting (IEDM)*, pages 14–3. IEEE.

[350] Vanheusden, K., Karna, S., Pugh, R., Warren, W., Fleetwood, D., Devine, R., and Edwards, A. (1998a). Thermally activated electron capture by mobile protons in sio 2 thin films. *Applied physics letters*, 72(1):28–30.

[351] Vanheusden, K., Warren, W., Devine, R., Fleetwood, D., Draper, B., and Schwank, J. (1999). A non-volatile mosfet memory device based on mobile protons in sio2 thin films. *Journal of non-crystalline solids*, 254(1-3):1–10.

[352] Vanheusden, K., Warren, W., Fleetwood, D., Schwank, J., Shaneyfelt, M., Draper, B., Winokur, P., Devine, R., Archer, L., Brown, G., et al. (1998b). Chemical kinetics of mobile-proton generation and annihilation in sio 2 thin films. *Applied physics letters*, 73(5):674–676.

[353] Vanka, S., Shin, H., Davidson, B. A., Liu, C., and Zou, K. (2022). Hydrogen atom doping—a versatile method for modulated interface resistive switching. *Advanced Electronic Materials*, 8(10):2200353.

[354] Vieluf, S., El Atrache, R., Cantley, S., Jackson, M., Clark, J., Sheehan, T., Bosl, W. J., Zhang, B., and Loddenkemper, T. (2022). Seizure-related differences in biosignal 24-h modulation patterns. *Scientific reports*, 12(1):1–9.

[355] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103.

[356] Von Neumann, J. (1993). First draft of a report on the edvac. *IEEE Annals of the History of Computing*, 15(4):27–75.

[357] Wakiya, N., Tajiri, N., Kiguchi, T., Mizutani, N., Cross, J. S., and Shinozaki, K. (2006). Activation energy of oxygen vacancy diffusion of yttria-stabilized-zirconia thin film determined from dc current measurements below 150 c. *Japanese journal of applied physics*, 45(6L):L525.

[358] Wan, W., Kubendran, R., Gao, B., Joshi, S., Raina, P., Wu, H., Cauwenberghs, G., and Wong, H. P. (2020). A voltage-mode sensing scheme with differential-row weight mapping for energy-efficient rram-based in-memory computing. In *2020 IEEE Symposium on VLSI Technology*, pages 1–2. IEEE.

[359] Wang, B. (2022). Tsmc investment report.

[360] Wang, J., Hu, S., Zhan, X., Yu, Q., Liu, Z., Chen, T. P., Yin, Y., Hosaka, S., and Liu, Y. (2018a). Handwritten-digit recognition by hybrid convolutional neural network based on hfo2 memristive spiking-neuron. *Scientific reports*, 8(1):12546.

[361] Wang, J. and Trolier-McKinstry, S. (2006). Oxygen vacancy motion in er-doped barium strontium titanate thin films. *Applied physics letters*, 89(17).

[362] Wang, X., Lin, X., and Dang, X. (2020a). Supervised learning in spiking neural networks: A review of algorithms and evaluations. *Neural Networks*, 125:258–280.

[363] Wang, X.-Y., Dong, C.-T., Zhou, P.-F., Nandi, S. K., Nath, S. K., Elliman, R. G., Iu, H. H.-C., Kang, S.-M., and Eshraghian, J. K. (2022). Low-variance memristor-based multi-level ternary combinational logic. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 69(6):2423–2434.

[364] Wang, X.-Y., Zhou, P.-F., Eshraghian, J. K., Lin, C.-Y., Iu, H. H.-C., Chang, T.-C., and Kang, S.-M. (2020b). High-density memristor-cmos ternary logic family. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(1):264–274.

[365] Wang, Y., Chen, W. P., Cheng, K. C., Chan, H. L. W., and Choy, C. L. (2003). Optical degradation of indium tin oxide thin films induced by hydrogen-related room temperature reduction. *Japanese journal of applied physics*, 42(5B):L546.

[366] Wang, Y., Wu, S., Tian, L., and Shi, L. (2020c). Ssm: a high-performance scheme for in situ training of imprecise memristor neural networks. *Neurocomputing*, 407:270–280.

[367] Wang, Z., Joshi, S., Savel'Ev, S., Song, W., Midya, R., Li, Y., Rao, M., Yan, P., Asapu, S., Zhuo, Y., et al. (2018b). Fully memristive neural networks for pattern classification with unsupervised learning. *Nature Electronics*, 1(2):137–145.

[368] Wang, Z., Joshi, S., Savel'ev, S. E., Jiang, H., Midya, R., Lin, P., Hu, M., Ge, N., Strachan, J. P., Li, Z., et al. (2017). Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nature materials*, 16(1):101–108.

[369] Warren, W., Fleetwood, D., Schwank, J., Shaneyfelt, M., Draper, B., Winokur, P., Knoll, M., Vanheusden, K., Devine, R., Archer, L., et al. (1997). Protonic nonvolatile field effect transistor memories in si/sio/sub 2//si structures. *IEEE Transactions on Nuclear Science*, 44(6):1789–1798.

[370] Waser, R. and Aono, M. (2010). Nanoionics-based resistive switching memories. In *Nanoscience and technology: A collection of reviews from nature journals*, pages 158–165. World Scientific.

[371] Waser, R., Baiatu, T., and Härdtl, K.-H. (1990a). Dc electrical degradation of perovskite-type titanates: I, ceramics. *Journal of the American Ceramic Society*, 73(6):1645–1653.

[372] Waser, R., Baiatu, T., and Härdtl, K.-H. (1990b). Dc electrical degradation of perovskite-type titanates: Ii, single crystals. *Journal of the American Ceramic Society*, 73(6):1654–1662.

[373] Waser, R., Dittmann, R., Staikov, G., and Szot, K. (2009). Redox-based resistive switching memories–nanoionic mechanisms, prospects, and challenges. *Advanced materials*, 21(25-26):2632–2663.

[374] Wei, Z., Kanzawa, Y., Arita, K., Katoh, Y., Kawai, K., Muraoka, S., Mitani, S., Fujii, S., Katayama, K., Iijima, M., et al. (2008). Highly reliable taox reram and direct evidence of redox reaction mechanism. In *2008 IEEE International Electron Devices Meeting*, pages 1–4. IEEE.

[375] Wen, S. and Zeng, Z. (2012). Dynamics analysis of a class of memristor-based recurrent networks with time-varying delays in the presence of strong external stimuli. *Neural processing letters*, 35(1):47–59.

[376] Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

[377] Widrow, B. and Lehr, M. A. (1990). 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415–1442.

[378] Wijesinghe, P., Ankit, A., Sengupta, A., and Roy, K. (2018). An all-memristor deep spiking neural computing system: A step toward realizing the low-power stochastic brain. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(5):345–358.

[379] Winokur, P., Boesch, H., McGarrity, J., and McLean, F. (1977). Field-and time-dependent radiation effects at the sio2/si interface of hardened mos capacitors. *IEEE Transactions on Nuclear Science*, 24(6):2113–2118.

[380] Woo, J., Moon, K., Song, J., Lee, S., Kwak, M., Park, J., and Hwang, H. (2016). Improved synaptic behavior under identical pulses using alo x/hfo 2 bilayer rram array for neuromorphic systems. *IEEE Electron Device Letters*, 37(8):994–997.

[381] Wronski, C. and Carlson, D. (1977). Surface states and barrier heights of metal-amorphous silicon schottky barriers. *Solid State Communications*, 23(7):421–424.

[382] Wu, W., Wu, H., Gao, B., Yao, P., Zhang, X., Peng, X., Yu, S., and Qian, H. (2018). A methodology to improve linearity of analog rram for neuromorphic computing. In *2018 IEEE symposium on VLSI technology*, pages 103–104. IEEE.

[383] Wu, X., Mei, S., Bosman, M., Raghavan, N., Zhang, X., Cha, D., Li, K., and Pey, K. L. (2015). Evolution of filament formation in ni/hfo2/siox/si-based rram devices. *Advanced Electronic Materials*, 1(11):1500130.

[384] Wu, X. and Saxena, V. (2017). Enabling bio-plausible multi-level stdp using cmos neurons with dendrites and bistable rrams. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 3522–3526. IEEE.

[385] Wu, X. and Saxena, V. (2018). Dendritic-inspired processing enables bio-plausible stdp in compound binary synapses. *IEEE Transactions on Nanotechnology*, 18:149–159.

[386] Wu, Y., Yu, S., Wong, H.-S. P., Chen, Y.-S., Lee, H.-Y., Wang, S.-M., Gu, P.-Y., Chen, F., and Tsai, M.-J. (2012). Alox-based resistive switching device with gradual resistance modulation for neuromorphic device application. In *2012 4th IEEE International Memory Workshop*, pages 1–4. IEEE.

[387] Wunderlich, T., Kungl, A. F., Müller, E., Hartel, A., Stradmann, Y., Aamir, S. A., Grübl, A., Heimbrecht, A., Schreiber, K., Stöckel, D., et al. (2019). Demonstrating advantages of neuromorphic computation: a pilot study. *Frontiers in neuroscience*, 13:260.

[388] Xia, L., Gu, P., Li, B., Tang, T., Yin, X., Huangfu, W., Yu, S., Cao, Y., Wang, Y., and Yang, H. (2016). Technological exploration of rram crossbar array for matrix-vector multiplication. *Journal of Computer Science and Technology*, 31(1):3–19.

[389] Xia, L., Huangfu, W., Tang, T., Yin, X., Chakrabarty, K., Xie, Y., Wang, Y., and Yang, H. (2017a). Stuck-at fault tolerance in rram computing systems. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 8(1):102–115.

[390] Xia, L., Li, B., Tang, T., Gu, P., Chen, P.-Y., Yu, S., Cao, Y., Wang, Y., Xie, Y., and Yang, H. (2017b). Mnsim: Simulation platform for memristor-based neuromorphic computing system. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(5):1009–1022.

[391] Xia, Q. and Yang, J. J. (2019). Memristive crossbar arrays for brain-inspired computing. *Nature materials*, 18(4):309–323.

[392] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR.

[393] Yakopcic, C., Taha, T. M., Subramanyam, G., and Pino, R. E. (2013). Generalized memristive device spice model and its application in circuit design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(8):1201–1214.

[394] Yang, J. J., Strukov, D. B., and Stewart, D. R. (2013). Memristive devices for computing. *Nature nanotechnology*, 8(1):13–24.

[395] Yao, P., Wu, H., Gao, B., Tang, J., Zhang, Q., Zhang, W., Yang, J. J., and Qian, H. (2020). Fully hardware-implemented memristor convolutional neural network. *Nature*, 577(7792):641–646.

[396] Ye, N., Cao, L., Yang, L., Zhang, Z., Fang, Z., Gu, Q., and Yang, G.-Z. (2023). Improving the robustness of analog deep neural networks through a bayes-optimized noise injection approach. *Communications Engineering*, 2(1):25.

[397] Yi, W., Savel'Ev, S. E., Medeiros-Ribeiro, G., Miao, F., Zhang, M.-X., Yang, J. J., Bratkovsky, A. M., and Williams, R. S. (2016). Quantized conductance coincides with state instability and excess noise in tantalum oxide memristors. *Nature communications*, 7(1):11142.

[398] Yon, E., Ko, W., and Kuper, A. (1966). Sodium distribution in thermal oxide on silicon by radiochemical and mos analysis. *IEEE Transactions on Electron Devices*, (2):276–280.

[399] Yu, S., Chen, P.-Y., Cao, Y., Xia, L., Wang, Y., and Wu, H. (2015). Scaling-up resistive synaptic arrays for neuro-inspired architecture: Challenges and prospect. In *2015 IEEE International Electron Devices Meeting (IEDM)*, pages 17–3. IEEE.

[400] Yu, S., Guan, X., and Wong, H.-S. P. (2012). On the switching parameter variation of metal oxide rram—part ii: Model corroboration and device design strategy. *IEEE Transactions on Electron Devices*, 59(4):1183–1188.

[401] Yu, S., Wu, Y., Jeyasingh, R., Kuzum, D., and Wong, H.-S. P. (2011). An electronic synapse device based on metal oxide resistive switching memory for neuromorphic computation. *IEEE Transactions on Electron Devices*, 58(8):2729–2737.

[402] Yun, J.-B., Kim, S., Seo, S., Lee, M.-J., Kim, D.-C., Ahn, S.-E., Park, Y., Kim, J., and Shin, H. (2007). Random and localized resistive switching observation in pt/nio/pt. *physica status solidi (RRL)–Rapid Research Letters*, 1(6):280–282.

[403] Zafar, S., Jagannathan, H., Edge, L. F., and Gupta, D. (2011). Measurement of oxygen diffusion in nanometer scale hfo2 gate dielectric films. *Applied Physics Letters*, 98(15).

[404] Zafar, S., Jones, R. E., Jiang, B., White, B., Chu, P., Taylor, D., and Gillespie, S. (1998). Oxygen vacancy mobility determined from current measurements in thin ba 0.5 sr 0.5 tio 3 films. *Applied physics letters*, 73(2):175–177.

[405] Zamarreño-Ramos, C., Camuñas-Mesa, L. A., Pérez-Carrasco, J. A., Masquelier, T., Serrano-Gotarredona, T., and Linares-Barranco, B. (2011). On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex. *Frontiers in neuroscience*, 5:26.

[406] Zbontar, J., LeCun, Y., et al. (2016). Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318.

[407] Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31.

[408] Zhang, X., Liu, S., Zhao, X., Wu, F., Wu, Q., Wang, W., Cao, R., Fang, Y., Lv, H., Long, S., et al. (2017). Emulating short-term and long-term plasticity of bio-synapse based on cu/a-si/pt memristor. *IEEE Electron Device Letters*, 38(9):1208–1211.

[409] Zhang, Y. and Ge, S. S. (2005). Design and analysis of a general recurrent neural network model for time-varying matrix inversion. *IEEE Transactions on Neural Networks*, 16(6):1477–1490.

[410] Zhang, Y., Wang, Z., Zhu, J., Yang, Y., Rao, M., Song, W., Zhuo, Y., Zhang, X., Cui, M., Shen, L., et al. (2020). Brain-inspired computing with memristors: Challenges in devices, circuits, and systems. *Applied Physics Reviews*, 7(1).

[411] Zhao, W., Chellappa, R., Phillips, P. J., and Rosenfeld, A. (2003). Face recognition: A literature survey. *ACM computing surveys (CSUR)*, 35(4):399–458.

[412] Zhong, N., Shima, H., and Akinaga, H. (2010). Transient current study on pt/tio2-x/pt capacitor. *Japanese Journal of Applied Physics*, 49(4S):04DJ15.

[413] Zhou, P. (2022). *Memristive Spiking Neural Network for Neuromorphic Computing*. University of California, Santa Cruz.

[414] Zhou, P. and Abbaszadeh, S. (2020). Towards real-time machine learning for anomaly detection. In *2020 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, pages 1–3. IEEE.

[415] Zhou, P., Choi, D.-U., Eshraghian, J. K., and Kang, S.-M. (2022). A fully memristive spiking neural network with unsupervised learning. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 634–638. IEEE.

[416] Zhu, R., Loeffler, A., Hochstetter, J., Diaz-Alvarez, A., Nakayama, T., Stieg, A., Gimzewski, J., Lizier, J., and Kuncic, Z. (2021). Mnist classification using neuromorphic nanowire networks. In *International Conference on Neuromorphic Systems 2021*, pages 1–4.

[417] Zhu, X., Wang, Q., and Lu, W. D. (2020a). Memristor networks for real-time neural activity analysis. *Nature communications*, 11(1):2439.

[418] Zhu, Y., Zhang, G. L., Wang, T., Li, B., Shi, Y., Ho, T.-Y., and Schlichtmann, U. (2020b). Statistical training for neuromorphic computing using memristor-based crossbars considering process variations and noise. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1590–1593. IEEE.

[419] Zucker, R. S. and Regehr, W. G. (2002). Short-term synaptic plasticity. *Annual review of physiology*, 64(1):355–405.