IET Cyber-Physical Systems: Theory & Applications

Research Article

Discrete-level memristive circuits for HTM-based spatiotemporal data classification system

ISSN 2398-3396
Received on 28th March 2017
Revised 15th July 2017
Accepted on 14th September 2017
E-First on 2nd November 2017
doi: 10.1049/iet-cps.2017.0053
www.ietdl.org

Aidana Irmanova[1] ✉, Timur Ibrayev[1], Alex Pappachen James[1]

[1]Bioinspired Microelectronic Systems Laboratory, School of Engineering, Nazarbayev University, Astana, Akmola, Kazakhstan

✉ *E-mail: aidana.irmanova@nu.edu.kz*

Abstract: The authors propose a discrete-level memristive memory design for analogue data processing in hardware implementations of hierarchical temporal memory (HTM). In this study, memristors were set to ternary and quaternary states in a sub-cell by application of different write voltage levels through a resistive network configuration. Simulations of the proposed circuit show that the highest number of discrete output levels of the memory was achieved using quaternary logic. However overall, using the same number of sub-cells and ternary logic exhibits the lowest relative error rate. For data classification purposes, the proposed discrete-level memristive cells are incorporated into the TM of HTM architecture, and its hardware circuit is presented for pattern recognition. They report improved results of face recognition using AR, ORL and UFI databases, and TIMIT database for speech recognition. These results are compared with the earlier design of HTM having only the spatial pooler (SP). Accuracy of the HTM architecture incorporating both SP and TM with discrete-level memristive cells for face recognition increased from 76.5 to 83.5% for AR database and speech recognition accuracy is improved from 73.3 to 93.3%.

1 Introduction

The growth and improvement of brain-inspired computing techniques for solving complex tasks such as data classification requires high-speed processing and high-level accuracy of its implementation. To fulfil these requirements, brain-inspired computing products are diverging into two directions. First is taking the system level approach, by developing software solutions, and focusing on improving the efficiency of algorithms. This method is commonplace and incorporable into existing systems, but highly dependent on computing and memory resources which is being addressed by shared pool of configurable computing resources or cloud computing. However, moving the computation part to the cloud server increases the requirements on robustness, bandwidth, and availability of communication networks of the system.

The other direction of implementing brain-inspired computing techniques would be mapping the algorithms directly to the hardware of computers. Hardware level data processing in such systems require the development of decision-making logic units [1] and storage units that can be realised in digital or analogue domains. In the case of digital data processing, variety of data storage is available and programming flexibility with greater precision is possible, which can justify the choice of digital data processing. Nevertheless, analogue mode of computation can offer a higher implementation density on silicon and require less power [2]. Moreover, analogue data processing complements neuromorphic engineering, making it possible to emulate bio-inspired models of machine learning in very large-scale integration (VLSI) technology. The first step in building analogue data processing system is developing analogue storage units [3], which is still an open research problem. Most of the approaches taken toward designing the analogue memory heavily rely on the device design, and minor changes in the device manufacturing significantly affect the accuracy of technology [4–6].

In this paper, we use memristors as the key element in developing analogue memory. In the past, memristors were used as a switching element, storing binary values [7] in crossbar array for image and audio processing [8, 9] and was incorporated in hybrid complimentary metal–oxide–semiconductor (CMOS) structures for designing a multilevel resistive random-access memory (ReRAM) [10]. Even though the exploitation of memristors in the design of memory technologies is not novel, presented work proposes a new design of the memory cell with programmable memristive sub-cells that can store from binary to multilevel or continuous analogue values. Taking into account that the output of analogue sensors that serve as an input source to the memory is a voltage signal, the programmability feature of the memory is set to be voltage driven. In this paper, we analyse the functionality of the proposed memory, calculating its relative error rate, power dissipation, and area for application in the brain-inspired algorithm – hierarchical temporal memory (HTM)-based data classification.

This paper is organised into six sections. Section 2 provides background information on analogue data storage implementations and HTM. Section 3 is divided into three sections that explain the design of the proposed memory cell focusing on the number of stored values, setting the resistive states of sub-cells and simulation results of the memory with relative error analyses. In Section 4, the hardware implementation of HTM based on the combination of memristive crossbar circuits for spatial pooler (SP) [11] with proposed discrete-level memristive cells is described and analysed for image and speech recognition. In Section 5, the discussion of discrete-level programmable memristive memory design for on-chip HTM pattern recognition is provided.

2 Background

One of the early designs that exploited analogue memory in data classification tasks was adaptive linear neuron or later adaptive linear element which memory element consisted of memristors [12]. A memistor (memory resistor) is a three-terminal device, made from an electroplating cell, which had thousands of possible analogue storage levels [13]. The main limitation of the technology was its non-scaling property, which made it redundant for further VLSI applications.

Another three-terminal device that was widely used to implement analogue memory cells is a floating gate transistor. Analogue output of floating gate memory devices heavily relied on accurate control of charge injections and operated in the high range of voltage levels [14]; therefore, are error prone in discrimination of data levels. In [15], floating gate devices were used for building analogue data storage of analogue neural networks.

Multilevel flash cells [16] and phase-change memory (PRAM) [17] are another set of attempts to build analogue memory.

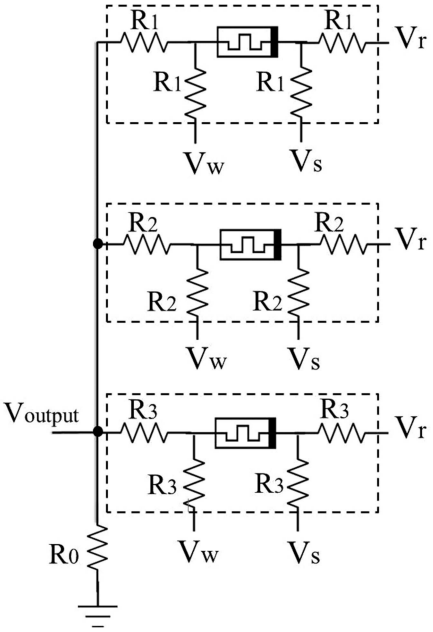

Fig. 1 *Circuit design of the proposed discrete analogue memory cell*

Multilevel flash is also a floating gate device that uses multiple levels per cell that can store more than 1 bit using the same number of transistors. The use of flash memory in neural network architecture was described in [18, 19]. PRAM is a type of non-volatile random-access memory that exploits behaviour of chalcogenide glass [20]. PRAM's switching time and inherent scalability make it appealing element for building analogue data storage, but its temperature sensitivity presents a notable issue [21], which should be taken into account during fabrication. PRAM for brain-like associative learning is described in [22].

A different approach to realise multilevel memory is using a memristor, a two-terminal electrical component, the value of which is the linear relation between the charge and flux [10, 23, 24]. Memristor has multilevel resistance property that can be controlled by the voltage or current applied across the device as a function of time and can be programmed to any resistance level between a maximum possible resistance and minimum resistance [25]. Materials used for fabrication of different types of memristors can be found in [26]. It is expected that scalable and efficient hardware solutions for mimicking the cognitive processing of human brains will be possible with the use of memristors [27]. One of such promising candidates for hardware implementation of cognitive processing algorithms is an HTM, which is a machine learning algorithm (LA) developed at Numenta Inc. [28]. It was designed as a model for emulating the algorithmic operations of the neocortex, a part of the human brain that is responsible for classification, learning, and decision making [29]. The algorithm is developed considering various aspects of neuroscience including modularity and hierarchy of information processing within the cortex, as well as memorisation and processing functionality of synapses. These concepts are explained in detail in the book 'On intelligence' by Hawkins [29].

The current model of HTM is a result of significant developments in its first two implementation models. The first implementation is called Zeta1 and primarily based on the Bayesian belief propagation [28]. Following Zeta1 model, the nodes have to be connected in a tree-shaped manner, with the bottom layer sensing data from the sensory input [28]. The second implementation was based on the cortical LA, which is closer to the current model of HTM, introducing the concepts of SP and temporal poolers [30]. This model is focused on learning and prediction making aspects of HTM. Despite the fact, that current HTM is based on these two models, with consistent improvements and modifications HTM is now expanding into brain and machine intelligence (BAMI) [31]. BAMI is constantly updated with concepts of machine intelligence and its influence on shaping HTM functionality of data processing [31]. According to the current version of the book, there are three major features of the HTM framework: (i) HTM should mainly be seen as a memory system, (ii) such memory system is primarily a memory of time-varying inputs, and (iii) the HTM regions are arranged and connected in a hierarchical manner [31].

3 Proposed memory cell

Memristors can store a minimum of two distinct logic states, as it is able to switch between two distinct resistance levels R_{on} and R_{off}, low and high resistances, respectively [32]. Memristors combine the behaviour of the non-linear resistor with the additional ability to memorise its resistive state. While there were attempts to quantise the resistance levels written to memristors, it was practically difficult to implement in circuit arrays and increased the complexity of the device. Therefore, to store values beyond binary and ternary states, we combine memristors programmed to ternary and quaternary values in a resistive divider configuration and store the information in total resistance of memristors connected in parallel. Circuit configuration of the proposed memory combining three memristors is shown in Fig. 1. Each memristor is connected to its resistive network and represents a memristive sub-cell together. The state of the memristive sub-cell is programmed by applying different voltage levels through its resistive network connected to write (V_{w1}, V_{w2}, V_{w3}) and reset (V_s) ports. Stored value of the cell can be read through (V_{output}) port applying low-voltage level through V_r read ports of the cell.

3.1 Number of stored values

In general, the memory cell with n sub-cells having unique resistive network $\forall R_1 \neq R_2 \neq \cdots \neq R_n$ can store $L = m^n$ discrete values, where m is the number of different values of resistance a sub-cell can store.

Fig. 1 shows the proposed memory cell with three memristive sub-cells that can provide up to 27 (=3^3) or 64 (=4^3) levels depending on whether ternary or quaternary logic ($m = 3$ or $m = 4$) sub-cells store. When the resistance values of the resistive networks of the sub-cells are equal, $\forall R_1 = R_2 = \cdots = R_n$ the states of each memristor of the cell represent non-positional code, and the number of m states combinations is equal to the binomial coefficient, which is 10 or 20 for 3 sub-cell memory

$$L = \frac{(n+m-1)!}{m!(n-1)!} \tag{1}$$

For programming the memristor in a sub-cell to desired level and overall the cell to multiple levels, reset port V_s and the write port V_w are used. While, to read the current state of the cell V_r port should be activated. The write signal V_w is connected to the positive terminal of memristors, and serve to change the resistive state of the sub-cell, while the reset signal V_s is connected to the negative terminal of the device and is used to erase the previous state of the sub-cell. Read signal should be applied to V_r port for reading the L combinations of memory states at the output V_{output} of the circuit.

The reset signal V_s is constantly at high logic state for each reset operation. While the write signal V_w in each of the sub-cells can take any of the three or four logic voltage levels represented as [0, 1, 2, 3]. The read signal V_r is driven to read V_r port during the read operation and set to logic low (V_r port gets grounded) in the write and reset operations.

3.2 Setting the resistive states of a memristor

The real-time programming of the memristor requires the application of a voltage pulse, which is greater than its threshold switching voltage. The range of writing voltage can be selected according to the memristor model, so the highest positive values are used for writing highest resistance levels, and highest negative value is used for the reset operation. Throughout this paper, we

Table 1 Used values for write pattern and resistive network of the sub-cells

Write voltage V_w, V				Wire resistances, Ω					
V_{w1}	V_{w2}	V_{w3}	V_{w4}	R_1	R_2	R_3	R_4	R_5	R_6
0	1.5	2	3	20	60	120	180	240	300

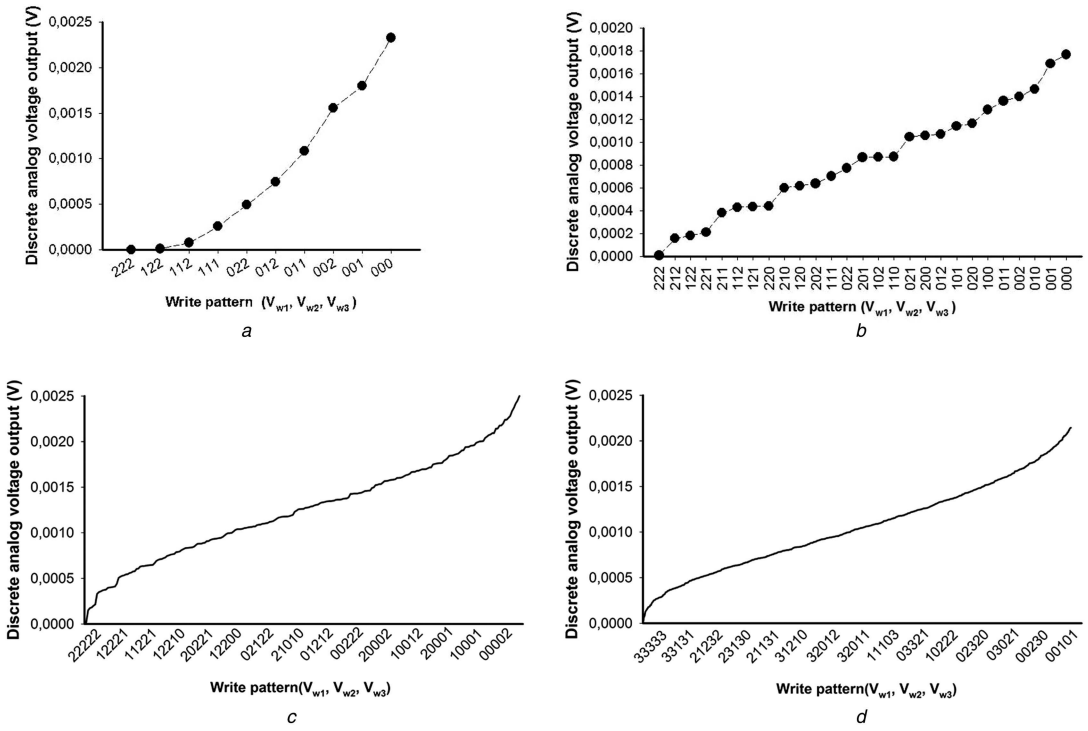

Fig. 2 *Discrete analogue output of*
(a) 10 level memory cell, ***(b)*** 27 level memory cell, ***(c)*** 243 level memory cell, ***(d)*** 1024 level memory cell

have used modified Pickett memristor model [33, 34] and kept the voltage in the range of −3 and +3 V.

3.2.1 Read operation: Values of used write V_w voltage levels and resistance values of resistive networks are provided in Table 1. Each sub-cell can store three or four different values indicating a ternary or quaternary logic system with the possible $V_{w1} = V_{gnd}$, V_{w2}, V_{w3}, and V_{w4} voltage levels (Fig. 1). The read voltages V_r of 0.1 V that is much lower than V_{max} is applied for 50 ns to enable the read operation. The reset V_s and the write ports V_w are connected to the ground in this operation. The output voltage V_{output} is indicative of the effective discrete analogue level of the memory cell

$$V_{output} = V_r \frac{\sum_{i=1}^{s}(1/R_{i1}k_i)}{\left(\sum_{i=1}^{s}(1/R_{i1})(1-(1/k_i))+(1/R_0)\right)} \quad (2)$$

where s are number of sub-cells, $n \in \{1, 2, \ldots, 6\}$, M_n is the memristance value of the nth sub-cell, $k_n = 2 + (R_{n1}/R_{n2})$, $R_{n1} = R_n + ((M_nR_n)/(2R_n + M_n))$, and $R_{n2} = R_n + (R_n^2/(2R_n + M_n))$.

3.2.2 Write operation: To get the same set of resistance values of memristor for corresponding write voltage values at every write operation (V_{w1} for logical state 0, V_{w2} for logical state 1, V_{w3} for logical state 2, and V_{w4} for logical state 3), it is necessary to reset the memristor to its initial resistance value, before applying write voltage signal V_w. In terms of speed, it will double up the writing period of the memory cell, but resetting the memristor is critical for the functionality of the proposed memory cell design. Writing and resetting operations are conducted in nano-scale time period, which makes it negligible prolongation of writing operation.

In the proposed design, resetting is implemented through V_s port (Fig. 1) which is located at the negative terminal of the memristor. The amplitude of V_s signal is V_{max} (3 V used for simulations) and period of 100 ns precede every write operation to keep the initial state of the sub-cell same and low. During the reset operation read (V_w) and write ports (V_w) are grounded, while after resetting the sub-cell, reset ports V_s get grounded and V_w are activated. To write the desired value of logic 0, logic 1, logic 2, or logic 3 the corresponding 0, 1.5, 2, or 3 V, voltage levels are driven into the write line for the same 100 ns period. In case of logic 0 the value of the sub-cell stays the same as after resetting, whereas other write signals change the resistance value of the memristor.

3.3 Simulations and results

The simulation of the circuit was conducted with modified Pickett memristor model [33]. To achieve 27 and 64 levels in 3 sub-cell memory, the values of resistors were set to $R_1 = 20\,\Omega$, $R_2 = 60\,\Omega$, and $R_3 = 180\,\Omega$, to get 81 and 256 levels in 4 sub-cell memory, the values of resistors were $R_1 = 20\,\Omega$, $R_2 = 60\,\Omega$, $R_3 = 120\,\Omega$, and $R_4 = 180\,\Omega$, for 243 and 1024 discrete levels in 5 sub-cell memory the resistances were set to $R_1 = 20\,\Omega$, $R_2 = 60\,\Omega$, $R_3 = 120\,\Omega$, $R_4 = 180\,\Omega$, and $R_5 = 240\,\Omega$, using ternary and quaternary logics, respectively.

To achieve 10 and 20, 15 and 35, 21 and 56 levels in three, four, five sub-cell memory cells the resistances of the resistive network should be set as $R_1 = R_2 = \cdots = R_n = 20\,\Omega$. These semiconductor resistors account for the wire resistances and are used to differentiate the voltage levels between the sub-cells.

Fig. 2*a* shows the output voltage levels resulting from simulation of three sub-cell memory as shown in Fig. 1, that is, ten distinct discrete levels, with the resistors R_1, R_2, and R_3 set to the same value of 20 Ω. Changing the resistance values of the sub-cells, i.e. $\forall R_1 \neq R_2 \neq R_3$ results in changing the voltage drop across the potential divider configuration within each sub-cell as shown in Fig. 1. This change in output voltage of the sub-cells leads to differentiated voltage levels at the output of the proposed memory cell.

Fig. 2*b* shows an example of the increased number of discrete voltage levels to 27, when the resistor values are set to $R_1 = 20\,\Omega$, $R_2 = 60\,\Omega$, and $R_3 = 180\,\Omega$. In Fig. 2*b*, it can be noted that some of the levels such as at 112 and 121, 121 and 220, 201 and 102, 102 and 110, and others have very close discrete voltage levels that can reduce the discrimination between the output voltage levels. On the other hand, Fig. 2*a* shows that there is only one pair of closest values that are 222 and 122.

Fig. 3 shows the histogram of differences between the subsequent levels as a relative percentage change in output voltage. The histogram indicates that the increase in the number of levels decreases the output voltage differences between subsequent levels. This relative change means that a cell with fewer levels can be more robust in operation for the given set of programming voltage levels (i.e. [0, 3 V]).

Table 2, on the other hand, provides different perspective with the analysis on sensitivity of the output states relative to the changes in write signal pattern. By introducing 5% change in write signal V_w pattern in L level memory cells it was concluded that, overall, memory cell programmed to quaternary logic is more prone to erroneous outputs than a cell programmed to ternary logic states. For instance, 243 (=3^5) level cell as shown in Fig. 2*c* exhibits the highest degree of tolerance to changes in write signal. Even though increasing the resistive states in the same five sub-cell memory achieve the highest number of levels $L = 1024(= 4^5)$, which plot is illustrated in Fig. 2*d*, the average relative error increases significantly from 6.01 to 20.64%.

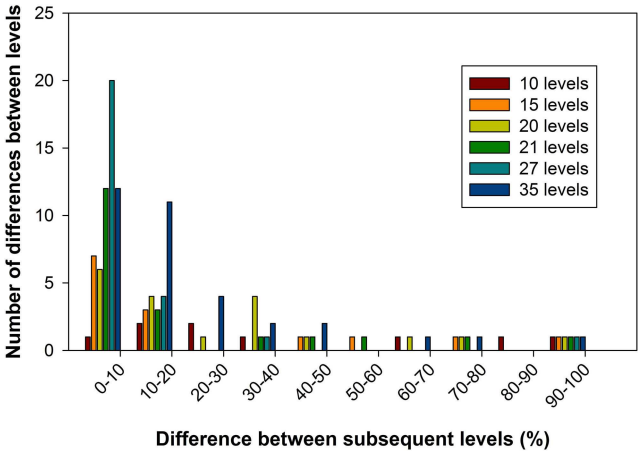

Fig. 3 *Histogram of relative percentage difference between the subsequent levels*

Furthermore, it was noted that using five or more sub-cells in given range of used values for resistive network and write voltage, result in increased or decreased number of discrete output levels than it was expected, i.e. a memory cell with five memristors having four states according to (1) have to achieve 56 levels, while simulation results show 59, a memory cell with 6 memristors having 3 and 4 states expected to achieve 729 and 4096 levels, and simulation results indicate 26 and 64 discrete output voltage levels, respectively. This partially can be explained with modelling issues of the memristors which require the proposed circuit to be configured differently in terms of resistive network values and applied input voltage levels.

3.4 Discussion

The memristors due to its small area, fast read times, low leakage currents, and its ease of programming makes it useful as a promising non-volatile memory device. The circuit structure and architecture used for data storage differ widely from the architecture to the ways of memristor's switching property is used. The most convenient structure for memristive memory cells is assumed to be crossbars with memristors switching from high to low resistance levels, storing binary values. However, there were attempts of implementing analogue memory, but due to the instability of the physical device, it is argued that one memristor is not sufficient for storing analogue information [35]. Another set of works on implementing multilevel memory propose separate reading and writing circuitries along with memristors to store multilevel (discrete) values [36, 37]. Examples of previous implementations of the multilevel memory cells are provided in Table 3. The main difference of the proposed memory cell compared with the previously implemented architectures is in its simplistic approach to write, reset, read, and store the information purely based on memristors. The structure of the voltage divider provides with different V_{output} levels across R_0 (Fig. 1), resulting from the V_r input voltage applied to memristors connected in parallel during the read operation.

The output levels of the memory cell and the difference between its possible output levels heavily rely on configuration of the voltage divider structure, which is the resistance values of the writing, resetting, and reading ports of the cell and applied input voltage values. In this paper, we provided experimental results

Table 2 Relative error for 5% change in write signal pattern of N memristors in a cell

		Three states			Four states			
		$R_1 = R_2 = \cdots = R_n$		$R_1 \neq R_2 \neq \cdots \neq R_n$	$R_1 = R_2 = \cdots = R_n$		$R_1 \neq R_2 \neq \cdots \neq R_n$	
n	L	Average error	L	Average error	L	Average error	L	Average error
3	10	14.98 ± 5.49	27	8.55 ± 3.33	20	21.11 ± 635	64	11.58 ± 3.83
4	15	11.16 ± 1.67	81	6.65 ± 1.20	35	19.65 ± 5.81	256	9.40 ± 1.14
5	21	12.22 ± 174	243	6.01 ± 0.33	59	117.46 ± 45.39	1024	20.64 ± 17.06
6	28	10.92 ± 2.04	26	7.62 ± 1.03	84	531.09 ± 558.64	64	35.75 ± 29.96

Table 3 Descriptions of memristive multilevel memory cells

Design for memristive multilevel memory	Description
a multilevel memristor-CMOS memory cell as an ReRAM [10]	design of the memory cell is based on CMOS switches that controls reading and writing operations. The core element of the cell is the memristor configurable to ternary logic
a read-monitored write circuit for 1T1M multilevel memristor memories [38]	authors claim that the proposed architecture is sneak path free and provides with fast read and write speeds with the exponential drift model of memristor. The structure of 1 transistor per memristor that provides for 2 bit storage in each data cell (memristor) are used for building a multilevel memory
memristor-based multilevel memory [36]	authors propose to use array of the reference resistors to write the resistance level to memristors. For reading and writing operations, they provide with different circuitries based on CMOS switches and pulse width modulators
design considerations for variation tolerant multilevel CMOS/nano memristor memory [37]	in this paper, crossbar array of memristors described for storing multilevel data. The comparison of using diode and transistors for reading and writing the information in terms of power consumption and overall multilevel memory limitation analysis are provided

from simulations of the memory cell that used resistors from 20 to 300 Ω and input voltage values from −3 to 3 V. The overall error of distinguishing output levels provided in Table 2 was calculated for the Pickett model presented in [34]. However, the parasitics and internal properties of physical memristors can vary from device to device depending on its material, size, and dimensions [39, 40]. Therefore, the design parameters such as V_r, V_s, and V_w can be adjusted to reduce the calculated errors during the chip fabrication and testing stages. Area and power consumption of the proposed memory will stay low irrespective of the errors in model-based simulations (see reference values in Table 4) due to the memristor's nano-scale and low-current leakage property.

4 Application in HTM

One of the options which can incorporate the proposed architecture of discrete-level analogue memory cell is the hardware of HTM for pattern recognition applications. HTM is the combination of its two building units, namely SP and TM. SP generates sparse-distributed representations of input data and practically can be used for feature extraction and pattern recognition applications on its own, whereas the functionality of TM is in providing the system with learning ability from input patterns and making predictions based on the temporal changes in the given stream of input data [30, 41, 42].

HTM was initially designed as a software algorithmic tool [30] and, among other research works [43–45] have shown the feasibility of HTM as a learning mechanism. Nevertheless, there were only few steps taken toward hardware implementation of this machine LA. For instance, Melis *et al.* [46] discuss the VLSI architecture required for HTM hardware implementation of digital application-specific integrated circuit, Zyarah [41] proposes design for hardware realisation of digital HTM using field programmable gate array, Fan *et al.* [47] present architecture for implementing HTM in the hardware as computing blocks using spin-neurons and memristive crossbar network and James et al. [11] and Ibrayev *et al.* [48] proposed circuit design of HTM SP which is based on a combination of memristor devices and CMOS technology. Thus, a hardware implementation of the HTM remains as a problem which still needs to be researched.

4.1 Spatial pooler

The SP [49] consists of initialisation, overlap computation, inhibition, and learning stages. An encoder converts any analogue input signal to an encoded digital form. The initialisation of the HTM regions is done by selecting a fixed number of columns. A dendrite segment connects each column with a set of potential synapses. These potential synapses and its permanence values are initialised randomly. The permanence values whose values are above a permanence threshold will be connected. The total number of connected synapses to active input bits forms the active synapses. The active synapses are multiplied with the frequency of activeness of the column relative to its neighbourhood which is referred to as boosting factor. The columns with highest activations are assigned as active with a given inhibition radius. Hebbian-style learning rule can be iteratively applied to update the permanence values for learning the input pattern. We denote x_j as the location Hebbian-style learning of the topologically arranged jth input neuron. The location of output neurons are denoted as y_i corresponding to the ith mini-column arranged topologically in output space. The hypercube of input space having a centre x_i^c with edge length γ represents the synapse for the ith SP. The potential input connections proportional–integral (PI)(i) to the input space for the ith mini-column are represented as

$$\mathrm{PI}(i) = \left\{ j \,|\, i(x_j;\, x_i^c,\, \gamma) \text{ and } (z_{ij} < \rho) \right\} \tag{3}$$

where $i(x_j;\, x_i^c,\, \gamma) = 1 \quad \forall x_j \in (x_i^c,\, \gamma)$, and $z_{ij} \sim U(0,\, 1)$, z is selected randomly from the uniform distribution U ranging in [0,1]. ρ is the fraction of inputs that are potential connections within the hypercube. The set of connected synapse can be represented using a binary matrix $\boldsymbol{B}$

$$\boldsymbol{B}_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{S}_{ij} \geq \theta_c \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Here, θ_c determines the percentage of the potential synapse that is connected. For example, $\theta_c = 0.5$ indicates that 50% of the potential synapse is connected. The matrix $\boldsymbol{S}_{ij} \in [0,\, 1]$ represents the synaptic permanence from the jth input to the ith SP mini-column such that

$$\boldsymbol{S}_{ij} = \begin{cases} U(0,\, 1) & \text{if } j \in \mathrm{PI}(i) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The local inhibition mechanism makes the neighbouring SP mini-columns to inhibit each other. The neighbourhood $\boldsymbol{N}_i$ of the ith SP mini-column is defined as

$$\boldsymbol{N}_i = \left\{ j \,|\, \| \, y_i - y_j \, \| < \phi, \quad i \neq j \right\} \tag{6}$$

where $\| \, y_i - y_j \, \|$ represents the Euclidean distance between the mini-columns at i and j. The parameter ϕ controls the inhibition radius and increases with increase in the receptive field size. The parameter ϕ is calculated as the multiplication between the average number of connected input span of all the SP mini-columns and the number of mini-columns per inputs. ϕ is set equal to γ if the dimensions of SP inputs and mini-columns are same. The input overlap is the feed-forward input to each mini-column and is the measure to determine the activation of SP mini-columns for a given input pattern $\boldsymbol{Z}$

$$o_i = \beta_i \sum_j \boldsymbol{B}_{ij} \boldsymbol{Z}_j \tag{7}$$

The excitability of each SP mini-column is controlled by training or selecting an appropriate value of β_i, generally known in HTM literature as boosting factor.

The SP mini-column becomes active provided two conditions are satisfied, i.e. input overlap is greater than a stimulus threshold θ_s and is among the top s percentile (prctile) of its neighbourhood. The active column selections are represented as

$$\alpha_i = 1, \text{ if } \quad (o_i \geq \mathrm{prctile}(\boldsymbol{NO}(i),\, 1 - s)) \text{ and } (o_i \geq \theta_s) \tag{8}$$

where $\boldsymbol{NO}(i) = \left\{ o_j \,|\, j \in \boldsymbol{N}(i) \right\}$, with s as the target activation density. This activation rule implements the k-winners-take-all computation within a local neighbourhood. Ideally, the parameter k can be adjusted to regulate the desired number of winning columns [50]. However, in the studies [11, 48], the inhibition phase is implemented by the winner takes all circuit; as a result, the value of the desired activity level is limited to 1.

The proposed discrete analogue memory cells were incorporated into the HTM architecture presented in [11]. Fig. 4 illustrates the overall system design, the main blocks of which are

Table 4 Reset, write, and read time of the memory and power dissipation calculation for each operation

Memory cell with n number of memristors	Reset	Write	Read
time, ns	100	100	50
power dissipation for $n = 3$ and $R_1 = R_2 = \cdots = R_n$, W	2.52×10^{-4}	1.56×10^{-1}	12.21×10^{-17}
power dissipation for $n = 4$ and $R_1 = R_2 = \cdots = R_n$, W	3.41×10^{-4}	2.15×10^{-1}	6.64×10^{-17}
power dissipation for $n = 3$ and $R_1 \neq R_2 \neq \cdots \neq R_n$, W	2.82×10^{-4}	5.90×10^{-2}	5.32×10^{-12}
power dissipation for $n = 4$ and $R_1 \neq R_2 \neq \cdots \neq R_n$, W	4.37×10^{-4}	6.81×10^{-2}	4.59×10^{-12}

SP and TM that is in this work constructed by the proposed analogue memory cells.

HTM SP is responsible for feature extraction of every input image and provides a binary output with 1's representing important spatial features and 0's representing unimportant spatial features [11, 48] such that collection of bits indicated as important spatial features then represent eyes, nose, and face contour of an individual shown on the image. During the training phase, such feature-extracted images are moved to TM. By focusing on important and unimportant spatial features, TM learns the facial images of each input class. After the system is trained and during the testing phase, extracted features of the testing image are moved directly to the pattern matcher and compared with features memorised by TM. As a result, the pattern matcher calculates the similarity scores between the testing image and all the classes memorised by the system. Finally, the best match and the class of the testing image are determined based on the maximum similarity score.

4.2 Temporal memory

The output from the active columns of the SP is fed to TM [51]. The cells in the active columns that are in the predictive state are activated, and if there is none in predictive state then activate all the cells. The activated cells provide the context to the prior input. In each layer, calculate the total synaptic connection between active cells and dendrite segments of all cells, such that if the total synaptic connections exceed a threshold assign corresponding dendrite segment to be active. Moreover, the cells linked to those dendrite segments are set active. The group of cells in predictive state form the predication pattern for the given layer. It is required to update the permanence values of potential synapse based on the occurrence of activation of dendrite segment. These modifications are assigned to be temporary. Some of the feed-forward inputs could change the cell state to active removing the temporary assignments. The changes of the cell state from predictive state to inactive require undoing the permanence changes marked as temporary.

Formally, the predictive state of the cells in TM [51] can be represented as

$$\pi_{ij}^{t} = \begin{cases} 1 & \text{if } \exists_d \| \widetilde{D}_{ij}^{d} \cdot \boldsymbol{A}^{t} \|_1 > \theta \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $\boldsymbol{A}^t$ is the activation matrix of size $N \times M$ representing N columns and M neurons per column with elements a_{ij}^t of $\boldsymbol{A}^t$ representing the activation state of the ith cell and jth column at the time point t. D_{ij}^d is the permanence of the dth segment of the ith cell in the jth column and $\widetilde{D}_{ij}^d$ corresponds to connected synapses. The segment activation threshold is represented as θ. The activation state

$$a_{ij}^{t} = \begin{cases} 1 & \text{if } j \in W^t \text{ and } \pi_{ij}^{t-1} = 1 \\ 1 & \text{if } j \in W^t \text{ and } \sum_i \pi_{ij}^{t-1} = 0 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where W^t represents the top $s1$ percentage of column that has most synaptic inputs. Typically $s1$ is set to 1–2%. Hebbs-like learning can be introduced to the dendrite segment activations. The reinforcement of the depolarised cell in a segment that subsequently becomes active can be done using

$$\Delta D_{ij}^{d} = \delta p \dot{\boldsymbol{D}}_{ij}^{d} \cdot A^{t-1} - \delta n \dot{\boldsymbol{D}}_{ij}^{d} \cdot (1 - \boldsymbol{A}^{t-1}) \tag{11}$$

where $\dot{\boldsymbol{D}}_{ij}^{d}$ is a binary matrix representing only positive elements of D_{ij}^d and the small values δn and δp are for introducing negative and positive reinforcements for inactive synapse and active synapse in dendrite segments, respectively. The long-term depression can be introduced by including a small decay to the active segments that did not become active using

$$\Delta D_{ij}^{d} = \delta \tilde{n} \dot{D}_{ij}^{d} \tag{12}$$

where $a_{ij}^t = 0$ and $\| \widetilde{D}_{ij}^{d} \cdot \boldsymbol{A}^{t-1} \|_1 > \theta$, with $\delta \tilde{n} \ll \delta n$.

Within HTM, the proposed discrete analogue memory cells are utilised in the design of TM, which, instead of saving all the feature-extracted images as it was done in the previous design of HTM incorporating only SP [11], creates a class map – a single image containing all the common and important temporal features of the images belonging to a single class in the training set. Such design reduces memory requirements and processing time, since pattern matching is performed only between the testing image and the class map of each of the trained classes.

The class map formation is realised by making TM learn by *focusing* on both important and unimportant spatial features and by *reflecting* how features change with time by changing the weight of memory cells, which are now used to represent temporal features. The difference is that spatial features extracted by SP are local to each particular image, whereas temporal features are the features that are common to all training images of a single class. To illustrate, while three different images of a single individual may have difference in the spatial feature representing face contour, e.g. due to spatial variations, the temporal feature, after being fully trained, ideally will represent the exact face contour of that particular individual. Then, the outputs of SP will be referred as feature-extracted (binary) images.

Focus, for the established learning mechanism, is achieved by placing TM circuitry after SP, so that the inputs are not the original training images, but feature-extracted images provided by the SP. Since each of the outputs of SP is binary in nature, such arrangement allows TM to differentiate important and unimportant spatial features.

Reflection is achieved by changing the weights of the TM cells according to the importance of the corresponding input bit within the entire training sequence. If the input bit of feature-extracted image is 1, meaning that it represents an important spatial feature, then the weight of the corresponding TM cell increases by *positive weight update* ($+\Delta$) value. On the contrary, if the input bit of the feature-extracted image is 0, meaning that it represents an unimportant spatial feature, then the weight of the corresponding TM cell decreases by *negative weight update* ($-\Delta$) value. Hence, the output of the TM is not binary, but analogue having discrete levels, and the weight difference between two consecutive levels is one Δ.

Fig. 5, as an example, illustrates the formation of the class map for the first class by fetching feature-extracted binary images belonging to the first class to TM. All of the TM cells, initially having the same weight, eventually become distinguishable at the end of the training sequence and reflect temporal variations (features) of the input training sequence.

However, such learning mechanism requires TM to be multi-valued. Thus, the proposed discrete analogue memory cells is a plausible candidate, on which TM can be built. This will allow the weights to take values not only of 1 and 0, as feature-extracted images of SP do, but can be changed according to the weight update value, which is $\pm\Delta$. Fig. 6 illustrates the design of TM required to memorise single class map and utilising proposed analogue memristive memory cells. The total number of the required memory cells correspond to the product of the number of memorised classes and the number of bits of a single class map. For example, for 13 class maps, each having dimensions of $120 \times 160\,\text{bits}^2$, the required number of memory cells is $13 \times 19,200$. For storing the 19,200 pixels, we can use memory cells based on four memristors with each memristor having the ability to store four logical states such that the resistive network is set as $R_1 \neq R_2 \neq \cdots \neq R_n$. The memristor used for the simulations [34] has an on-chip area of $5 \times 5\,\mu\text{m}^2$ resulting in the overall on-chip area requirement to be $1.92\,\mu\text{m}^2$ for storing one class map. As regards the power consumptions of the storage, reference values are provided in Table 4.

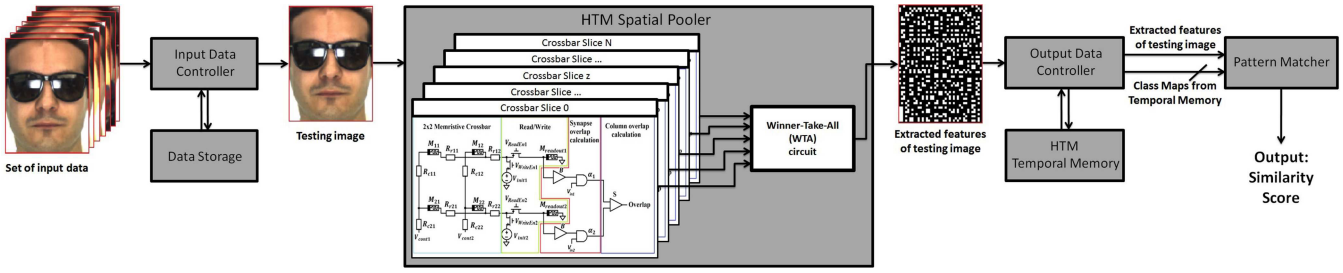

Fig. 4 *High-level block diagram of the HTM system on which the proposed discrete analogue memory cells were verified*

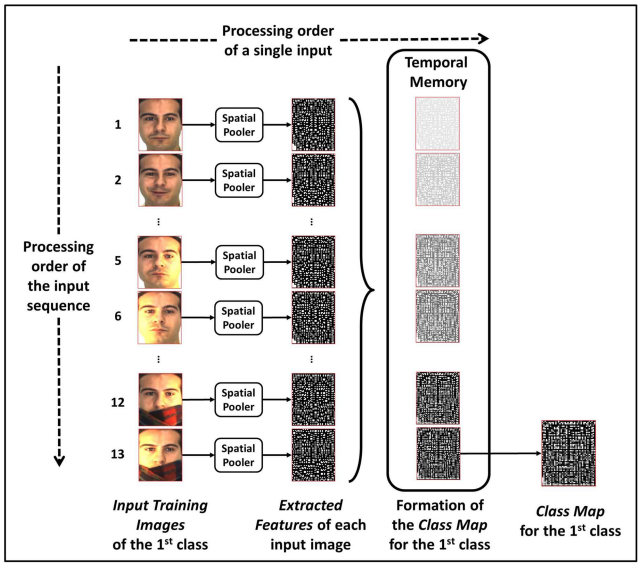

Fig. 5 *Main principle of single class map formation using TM and feature-extracted images obtained from SP*

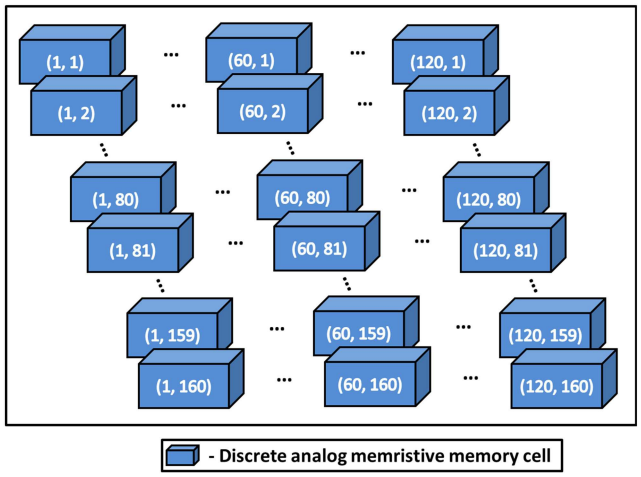

Fig. 6 *Design of TM consisting of* $120 \times 160 = 19,200$ *memory cells and that is used for storing a single class map trained by fetching input images having dimensions of* 120×160 *bits*^{2}

4.3 Application results

On the basis of the proposed memory cells, HTM architecture was verified on face and speech recognition applications. For face recognition purposes, the system was tested utilising the images from AR database [52], ORL database [53], and cropped images dataset from UFI database [54].

AR database consists of 100 classes and 26 facial images of a single person within each class, which were taken in two recording sessions (13 images in each session). The images in each session are categorised into five groups: (a) one image for neutral expression, (b) three images for different emotional expressions (smile, anger, and scream), (c) three images under different light conditions, (d) three images with sunglasses, and (e) three images with scarf both servings as occlusions under different light conditions.

ORL database consists of 40 classes and 10 facial images of a single person within each class. The images in this database are unsystematic, with not only various emotional expressions and light conditions for different classes, but also with tolerance to side movements.

Cropped images dataset from UFI database consist of 605 classes and on average 7.1 facial images of a single person within each class. These images were collected under real-world conditions and are real photographs of Czech News Agency and, therefore, tolerate side movements and unsystematic occlusions in the form of glasses, hands etc. under various light conditions.

4.3.1 Face recognition: As a first analysis, the design employing both HTM SP and TM was simulated using AR database to identify the most optimal delta among initially specified deltas, for which the system provides the maximum face recognition accuracy. The images of the first and the second sessions of AR database were used during the training and testing phases of HTM TM, respectively, and number of training images was increased from 1 to 13 to observe the trend.

Fig. 7 illustrates the results of the simulation. As can be seen for all the combinations of training set and delta amounts, higher accuracies are achieved when weight update value is on the order of ± 0.1 or less. Moreover, as can be noted with the increasing size of the training set the accuracy drops for large delta amounts. Thus, for the further simulation purposes with HTM SP and TM designs ± 0.05 was used as a weight update amount.

The second analysis was aimed to investigate the effectiveness of the proposed TM. This was done by comparing face recognition accuracy results obtained by the architecture utilising only SP presented in [11] and the architecture combining the same SP with the proposed TM. To make common settings, for the architecture of only SP the training images belonging to a single class were initially averaged and only then the averaged images were processed by SP to provide a feature-extracted training template for each class. For the proposed architecture, the training images were processed by SP and feature-extracted outputs were used to create a single class map for each class within TM. As a result, pattern matching for both architectures was performed by comparing testing images with only one training template or only one class map for each trained (memorised) class.

Fig. 8*a* illustrates the recognition accuracy results for AR face database achieved by two different architectures for various ratios of training to testing images. Similarly, Fig. 8*b* illustrates the recognition accuracy results for ORL face database achieved by two different architectures for various ratios of training to testing images. Both of these results imply that, when the number of images available to train the system is sufficiently high, the design utilising discrete analogue memory cells as TM is more effective than the architecture based on SP alone.

The third analysis was aimed at determining the effectiveness of the proposed TM design at differentiating various categories within each class. Specifically, SP alone and the combination of SP and the proposed TM were used to distinguish four different categories within each class of AR database and to make gender classification of UFI database images. The images of AR database were divided to have 13 training images and 13 testing images for each class. The images of UFI database were divided to have 111 training and

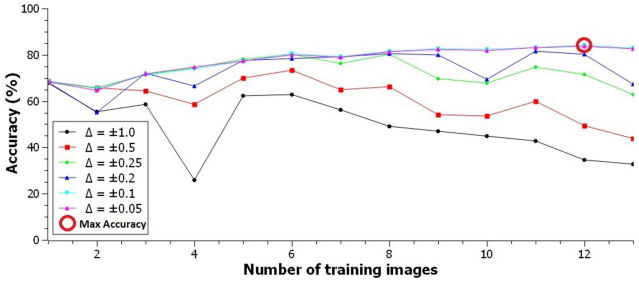

Fig. 7 *Optimal delta (Δ) estimation for AR database based on recognition accuracy results*

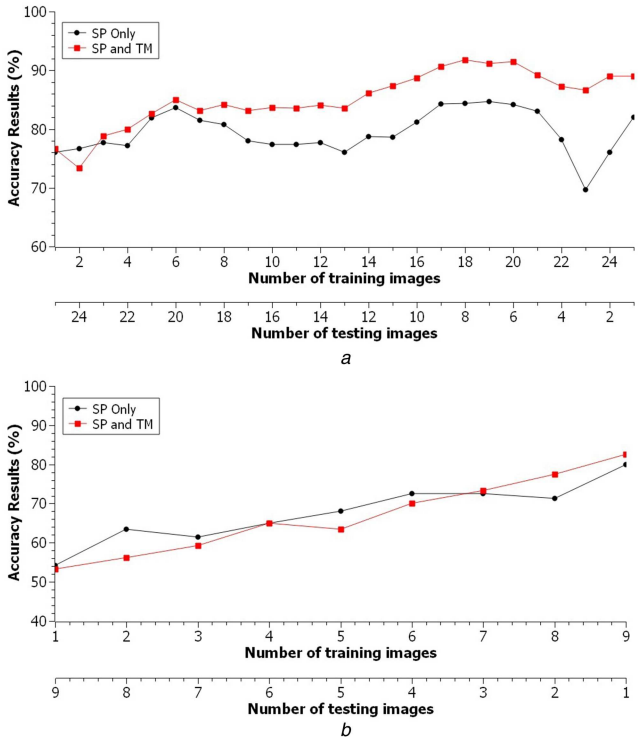

Fig. 8 *Recognition accuracy results achieved by two different architectures for various ratios of training to testing images for* ***(a)*** AR face database, ***(b)*** ORL face database

111 testing images for males and females. Moreover, similarly to the previous analysis, for the architecture utilising only SP, the input images were initially averaged and only then fetched to provide fair comparison.

Table 5 illustrates the results of classifying test images of AR database into one of the four categories (different emotions, light conditions, glasses as occlusion, and scarf as occlusion) achieved by two different architectures. Table 6 illustrates the results of gender classification of UFI database images achieved by two different architectures. As it can be seen, addition of the proposed TM to SP circuits increases recognition accuracy results by almost 7% in the case of AR face database and above 8% in the case of UFI face database. In turn, these results suggest that, when the number of training images is large, addition of the proposed TM increases the capability of the HTM architecture to distinguish different categories within each class.

4.3.2 Speech recognition: While SP deals with spatial feature extraction, TM is a vital part that deals with time-varying features.

Table 6 Gender classification results of UFI database images done by two different architectures using single template or class map per each class

Architecture	Male, %	Female, %	Total, %
SP [11]	47.75	78.38	63.06
SP and TM	75.68	66.67	71.17

Hence, while face recognition results mainly illustrated capability of HTM to extract spatial features and only then indicated how features vary from sample to sample (that is time-varying features), speech recognition was important to illustrate how HTM deals with context-based dynamic sequential processing of samples with inherited time variations.

The capability of the proposed design to perform recognition of fast-varying temporal patterns was verified by performing speech recognition analysis on the TIMIT database [55]. TIMIT database contains sentences with over 7000 unique words, very few of which are repeated enough to use them as a matching template for speech recognition purposes. Most of the words that are repeated are articles (a, the) and prepositions (in etc.). Hence, due to short duration of the repeated words, the system was tested for speech recognition using only *sa1* and *sa2* sentences that are usually reserved for speaker recognition studies:

sa1: She had your dark suit in greasy wash water all year.
sa2: Do not ask me to carry an oily rag like that.

These sentences contain 21 words in total. The training set contains 50 instances per class, whereas the testing set consists of 15 instances per class. It is important to highlight that when fetching entire sentences at once the words within these sentences are actually samples fetched together in a particular sequence. As a result, when the waveform for each sentence sample (that is sa1 or sa2) is fetched, it is actually the sequence of words being fetched, where each word has a different context based on its position within the sentence.

In this work, for a speech waveform to be compatible with the proposed HTM architecture, the initial preprocessing was performed using perceptual linear prediction (PLP) feature extraction method. The procedure and codes described in [56] were used to convert a speech waveform into an image, representing 12th-order PLP features without representations relative spectra filtering (band-pass filtering to the energy in each frequency subband [57]). Consequently, each speech sample was converted into an image without significant degradation of temporal details.

Next, the images representing PLP features were used in the similar recognition procedure described for the face recognition. For the design based on only SP, the images of a single class were averaged and only then fetched to the system to extract features and to create a training template for the corresponding class. Contrary, for the design based on SP and TM the images of a single class were sequentially fetched to SP to extract features and by using these features to create class maps within TM.

Also, the additive white Gaussian noise was added to speech samples to examine noise robustness of the proposed architecture. As a result, Fig. 9 illustrates speech recognition accuracy results for different values of signal-to-noise ratio.

5 Conclusion

We presented a design of the discrete-level memristive memory cell and demonstrated its application in spatiotemporal data classification with HTM architecture. The proposed HTM architecture incorporates SP circuit and TM with presented

Table 5 Classification results of test images into each category of AR database done by two different architectures using single template or class map per each class

Architecture	Emotions, %	Light conditions, %	Occlusions (glasses), %	Occlusions (scarf), %	Total, %
SP [11]	77.50	91.00	84.33	53.33	76.54
SP and TM	84.25	96.33	85.67	67.67	83.48

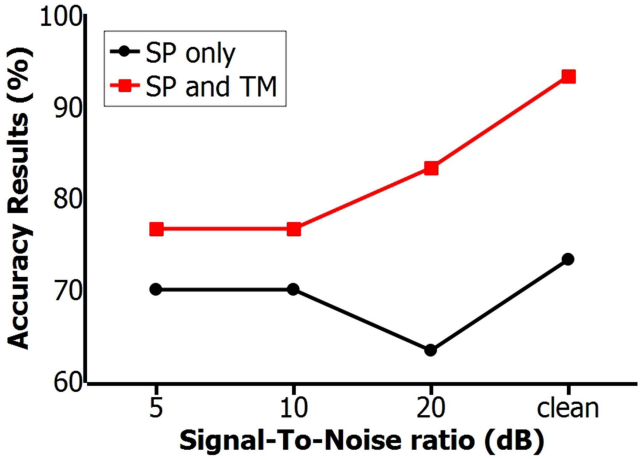

Fig. 9 *Speech recognition accuracies obtained by two different architectures for various values of signal-to-noise ratio using PLP feature extraction method as preprocessing*

memristive cells. Simulations of the circuit on face recognition and speech recognition tasks show that combining the TM circuits with the SP circuits result in the increase of accuracy results of data classification tasks and enhance the noise robustness of the overall system.

It is noted that recognition results of the proposed HTM architecture compared with the previous HTM SP design at the maximum possible number of training images improve in accuracy to 10% for AR database and to about 2% for ORL database recognition. On the classification task, the proposed design increases the accuracy by about 7% for constrained images of AR database and by about 8% for unconstrained images of UFI database. On speech recognition task, simulations show that the proposed architecture provides stable recognition accuracies for various noise figures.

As regards to memory, the minimum estimated error rate for the proposed discrete-level memory cell was at $6.01 \pm 0.33\%$ with 243 level memory, using the ternary logic of programming memristive cells. Even though the memory cell with memristive cells programmed to ternary states exhibits a higher degree of tolerance, a higher number of discrete output levels can be achieved using quaternary logic. For given configuration of the resistive network of memristive sub-cells, the highest number of discrete output levels of memory achieved 1024 through a combination of five sub-cells. Importantly, calibrating the values of the resistive network around the sub-cell will result in different figures of output levels, which also reflect modelling issue of the physical behaviour of memristors. This implies that it is possible to achieve continuous output levels of memory states while still having low error rate value. Further work can investigate this possibility and apply the proposed HTM architecture for real-time video processing. Such application will emphasise the importance of adding cognitive processing units and give a plausible solution for other associated problems as data deluge.

6 Acknowledgments

The initial discussion of HTM with Dhireesha Kudithipudi, and her group members at NanoComputing Lab with RIT, is greatly acknowledged. The initial work on HTM TM with Ulan Myrzakhan from the Nazarbayev University is also acknowledged. The funding support provided through NU ORAU research grant no. SOE2015008 is acknowledged.

7 References

[1] Dubois, J., Mattavelli, M.: 'Embedded co-processor architecture for CMOS based image acquisition'. Proc. 2003 Int. Conf. on Image Processing 2003 ICIP 2003, 2003, vol. **2**, p. II–591

[2] Rojas, R.: '*Neural networks: a systematic introduction*' (Springer Science & Business Media, 2013)

[3] Fujita, O., Amemiya, Y.: 'A floating-gate analog memory device for neural networks', *IEEE Trans. Electron Devices*, 1993, **40**, (11), pp. 2029–2035

[4] Haller, G.M., Wooley, B.A.: 'An analog memory integrated circuit for waveform acquisition up to 900 MHz'. 1993 IEEE Nuclear Science Symp. Medical Imaging Conf., San Francisco, CA, 2–5 November 1993, vol. **1**, pp. 2–5

[5] O'Halloran, M., Sarpeshkar, R.: 'A 10 nW 12 bit accurate analog storage cell with 10 aa leakage', *IEEE J. Solid-State Circuits*, 2004, **39**, (11), pp. 1985–1996

[6] Diorio, C., Mahajan, S., Hasler, P., *et al.*: 'A high-resolution non-volatile analog memory cell'. 1995 IEEE Int. Symp. Circuits and Systems 1995 ISCAS'95, 1995, vol. **3**, pp. 2233–2236

[7] Chen, Y., Jung, G.-Y., Ohlberg, D.A.A., *et al.*: 'Nanoscale molecular-switch crossbar circuits', *Nanotechnology*, 2003, **14**, (4), p. 462

[8] Hu, X.F., Duan, S.K., Wang, L.D., *et al.*: 'Memristive crossbar array with applications in image processing', *Sci. China Inf. Sci.*, 2012, **55**, (2), pp. 461–472

[9] Duan, S.K., Hu, X.F., Wang, L.D., *et al.*: 'Analog memristive memory with applications in audio signal processing', *Sci. China Inf. Sci.*, 2014, **57**, (4), pp. 1–15

[10] Rabbani, P., Dehghani, R., Shahpari, N.: 'A multilevel memristor–CMOS memory cell as an ReRAM', *Microelectron. J.*, 2015, **46**, (12), pp. 1283–1290

[11] James, A.P., Fedorova, I., Ibrayev, T., *et al.*: 'HTM spatial pooler with memristor crossbar circuits for sparse biometric recognition', *IEEE Trans. Biomed. Circuits Syst.*, 2017, **PP**, (99), pp. 1–12

[12] Anderson, J.A., Rosenfeld, E.: '*Talking nets: an oral history of neural networks*' (MIT Press, 2000)

[13] Angell, J.B., Widrow, B., Pierce, W.H.: 'Birth, life, and death in microelectronic'. Technical Report No. 1552-2/1851-1, May 1961

[14] Hasler, P., Minch, B., Mead, C., *et al.*: 'A high-resolution nonvolatile analog memory cell'. Proc. 1995 IEEE Int. Symp. Circuits and Systems, 1995, vol. **3**, pp. 2233–2236

[15] Lee, B.W., Sheu, B.J., Yang, H.: 'Analog floating-gate synapses for general-purpose VLSI neural computation', *IEEE Trans. Circuits Syst.*, 1991, **38**, (6), pp. 654–658

[16] Lee, S.-K., Park, D.-H.: 'Multi level flash memory device and program method'. US Patent 7,054,199, May 2006

[17] Lee, J.-W.S.: 'Multilevel phase change memory'. US Patent 7,488,968, 10 February 2009

[18] Khan, S.: 'The divided flash memory device for implementing neurons and neural networks'. , 2013 Int. Conf. Informatics, Electronics & Vision (ICIEV), 2013, pp. 1–5

[19] Strukov, D., Merrikh-Bayat, F., Prezioso, M., *et al.*: 'Memory technologies for neural networks'. 2015 IEEE Int. Memory Workshop (IMW), 2015, pp. 1–4

[20] Simpson, R.E., Fons, P., Kolobov, A.V., *et al.*: 'Interfacial phase-change memory', *Nat. Nanotechnol.*, 2011, **6**, (8), pp. 501–505

[21] Lee, B.C., Zhou, P., Yang, J., *et al.*: 'Phase-change technology and the future of main memory', *IEEE Micro*, 2010, **30**, (1), pp. 131–141

[22] Eryilmaz, S.B., Kuzum, D., Jeyasingh, R., *et al.*: 'Brain-like associative learning using a nanoscale non-volatile phase change synaptic device array', arXiv preprint arXiv:1406.4951, 2014

[23] Chua, L.: 'Memristor – the missing circuit element', *IEEE Trans. Circuit Theory*, 1971, **18**, (5), pp. 507–519

[24] Mostafa, H., Ismail, Y.: 'Process variation aware design of multi-valued spintronic memristor-based memory arrays', *IEEE Trans. Semicond. Manuf.*, 2016, **29**, (2), pp. 145–152

[25] Kannan, S., Karimi, N., Karri, R., *et al.*: 'Modeling, detection, and diagnosis of faults in multilevel memristor memories', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2015, **34**, (5), pp. 822–834

[26] Maan, A.K., Jayadevi, D.A., James, A.P.: 'A survey of memristive threshold logic circuits', *IEEE Trans. Neural Netw. Learn. Syst.*, 2016, **PP**, (99), pp. 1–13

[27] Versace, M., Chandler, B.: 'The brain of a new machine', *IEEE Spectr.*, 2010, **47**, (12), pp. 30–37

[28] George, D., Hawkins, J.: 'A hierarchical Bayesian model of invariant pattern recognition in the visual cortex'. Proc. 2005 IEEE Int. Joint Conf. Neural Networks, 2005 IJCNN '05, July 2005, vol. **3**, pp. 1812–1817

[29] Hawkins, J., Blakeslee, S.: '*On intelligence*' (Macmillan, 2007)

[30] Hawkins, J., Ahmad, S., Dubinsky, D.: 'Hierarchical temporal memory including HTM cortical learning algorithms'. Technical Report, Numenta, Inc., Palto Alto, 2010. Available at http://www.numenta.com/htmoverview/education/HTM_CorticalLearningAlgorithms.pdf, accessed 16/03/2017

[31] Hawkins, J., Ahmad, S., Purdy, S., *et al.*: 'Biological and machine intelligence (BAMI)', Initial online release 0.4, 2016

[32] Stanley Williams, R.: 'How we found the missing memristor', *IEEE Spectr.*, 2008, **45**, (12), pp. 28–35

[33] Biolek, D., Kolka, Z., Biolkova, V., *et al.*: 'Memristor models for spice simulation of extremely large memristive networks'. 2016 IEEE Int. Symp. Circuits and Systems (ISCAS), 2016, pp. 389–392

[34] Pickett, M.D., Strukov, D.B., Borghetti, J.L., *et al.*: 'Switching dynamics in titanium dioxide memristive devices', *J. Appl. Phys.*, 2009, **106**, (7), p. 074508

[35] Rose, G.S.: 'Overview: memristive devices, circuits and systems'. Proc. 2010 IEEE Int. Symp. Circuits and Systems (ISCAS), 2010, pp. 1955–1958

[36] Kim, H., Sah, M.P., Yang, C., *et al.*: 'Memristor based multilevel memory'. 2010 12th Int. Workshop on Cellular Nanoscale Networks and their Applications (CNNA 2010), 2010, pp. 1–6

[37] Manem, H., Rose, G.S., He, X., *et al.*: 'Design considerations for variation tolerant multilevel CMOS/nano memristor memory'. Proc. 20th Symp. Great lakes Symp. VLSI, 2010, pp. 287–292

[38] Manem, H., Rose, G.S.: 'A read-monitored write circuit for 1T1M multi-level memristor memories'. 2011 IEEE Int. Symp. Circuits and systems (ISCAS), 2011, pp. 2938–2941

[39] Duarte, J.C., Martins, E.V., Alves, L.N.: 'Frequency characterization of memristive devices'. 2013 European Conf. Circuit Theory and Design (ECCTD), 2013, pp. 1–4
[40] Hpl.hp.com HP Memristor FAQ
[41] Zyarah, A.M.: 'Design and analysis of a reconfigurable hierarchical temporal memory architecture'. Master's thesis, 2015
[42] Mnatzaganian, J., Fokoué, E., Kudithipudi, D.: 'A mathematical formalization of hierarchical temporal memory's spatial pooler', arXiv preprint arXiv:1601.06116, 2016
[43] Farahmand, N., Dezfoulian, M.H., GhiasiRad, H., *et al.*: 'Online temporal pattern learning'. 2009 Int. Joint Conf. Neural Networks, June 2009, pp. 797–802
[44] Ramli, I., Ortega-Sanchez, C.: 'Pattern recognition using hierarchical concatenation'. 2015 Int. Conf. Computer, Control, Informatics and its Applications (IC3INA), October 2015, pp. 109–113
[45] Csapo, A.B., Baranyi, P., Tikk, D.: 'Object categorization using VFA-generated nodemaps and hierarchical temporal memories'. IEEE Int. Conf. Computational Cybernetics 2007 ICCC 2007, October 2007, pp. 257–262
[46] Melis, W.J.C., Chizuwa, S., Kameyama, M.: 'Evaluation of the hierarchical temporal memory as soft computing platform and its VLSI architecture'. 39th Int. Symp. Multiple-Valued Logic, 2009, pp. 233–238
[47] Fan, D., Sharad, M., Sengupta, A., *et al.*: 'Hierarchical temporal memory based on spin-neurons and resistive memory for energy-efficient brain-inspired computing', *IEEE Trans. Neural Netw. Learn. Syst.*, 2016, **27**, (9), pp. 1907–1919
[48] Ibrayev, T., James, A.P., Merkel, C., *et al.*: 'A design of HTM spatial pooler for face recognition using memristor–CMOS hybrid circuits'. 2016 IEEE Int. Symp. Circuits and Systems (ISCAS), May 2016, pp. 1254–1257
[49] Cui, Y., Ahmad, S., Hawkins, J.: 'The HTM spatial pooler: a neo cortical algorithm for online sparse distributed coding', bioRxiv, 2017, p. 085035
[50] Numenta Inc.: 'Hierarchical temporal memory including HTM cortical learning algorithms'. Technical Report, 2006
[51] Cui, Y., Ahmad, S., Hawkins, J.: 'Continuous online sequence learning with an unsupervised neural network model', *Neural Comput.*, 2016, (28), pp. 2474–2504
[52] Martınez, A., Benavente, R.: 'The AR face database'. Rapport Technique 24, 1998
[53] Samaria, F.S., Harter, A.C.: 'Parameterisation of a stochastic model for human face identification'. Proc. Second IEEE Workshop on Applications of Computer Vision 1994, 1994, pp. 138–142
[54] Lenc, L., Král, P.: 'Unconstrained facial images: database for face recognition under real-world conditions'. 14th Mexican Int. Conf. Artificial Intelligence (MICAI 2015), Cuernavaca, Mexico, 25–31 October 2015
[55] Garofolo, J.S., Lamel, L.F., Fisher, W.M., *et al.*: 'DARPA TIMIT acoustic–phonetic continuous speech corpus CD-ROM. nist speech disc 1-1.1'. NASA STI/Recon Technical Report N 93, 1993
[56] Ellis, D.P.W.: PLP and RASTA (and MFCC, and inversion) in MATLAB, 2005
[57] Hermansky, H., Morgan, N.: 'Rasta processing of speech', *IEEE Trans. Speech Audio Process.*, 1994, **2**, (4), pp. 578–589