Pattern Recognition with Memristor Networks

Patrick Sheridan, Wen Ma, Wei Lu*
Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI, U.S.A.
wluee@eecs.umich.edu

***Abstract*—In this paper we develop the concept of implementing pattern recognition algorithms in analog memristor networks. First, a device model is presented with experimental results demonstrating the feasibility of using WOx-based memristors to represent the tunable weights in a neural network. Next, simulation results demonstrate that an array of these memristors can be used to implement an unsupervised learning algorithm for pattern recognition. Handwritten digits are classified as an example problem while the concept is developed for more general use.**

Keywords— resistive switching, pattern recognition, memristor, unsupervised learning

I. Introduction

Pattern recognition is an important task in developing autonomous, intelligent computers capable of assisting or replacing humans in dangerous or tedious tasks such as disease diagnosis, autonomous transportation, and sorting mail [1]. The emergence of nanoscale resistive switching devices (memristors) enables the implementation of pattern recognition algorithms directly in hardware with designs that are analogous to the visual cortex [2-4]. This allows dense storage of algorithm parameters and parallel execution, which can improve recognition performance as well as conserve space and energy. In this paper, we demonstrate the feasibility of using tungsten-oxide based resistive switches to implement an unsupervised learning algorithm for handwritten digit recognition tasks which can be generalized to solve many clustering problems.

II. Resistive Switching

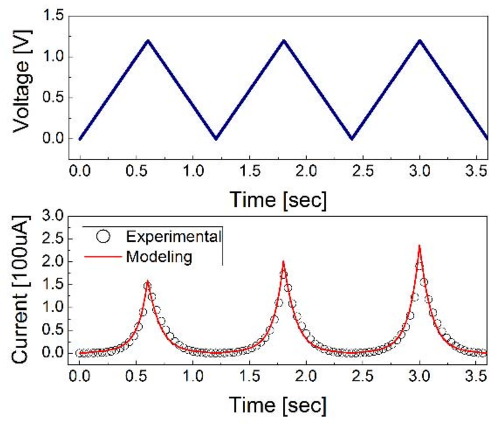
Reliable analog resistive switching has been achieved in nanoscale Pd/WO_x/W devices. Under an applied voltage bias oxygen vacancies within the WO_x layer redistribute and affect the conductance of the cell. As described in [5], the device conductance can be modeled with the follow set of equations, which include a state variable, w.

$$\frac{dw}{dt} = \lambda \sinh(\eta V) \quad (1)$$

$$I = (1-w)\alpha(1-\exp(-\beta V)) + w\gamma \sinh(\delta V) \quad (2)$$

The state variable represents the fraction of device area consumed by conductive vacancy filaments, and its dynamic growth is described by Eq. 1. Eq. 2 describes the current through the device, which can be divided into conduction through the filament region (modeled as tunneling current in this case as the 2nd term in Eq. 2) and conduction through the rest of the film (modeled as a Schottky diode).

This work was supported in part by NSF through grant ECCS-0954621 and by DARPA through grant HR0011-13-2-0015.



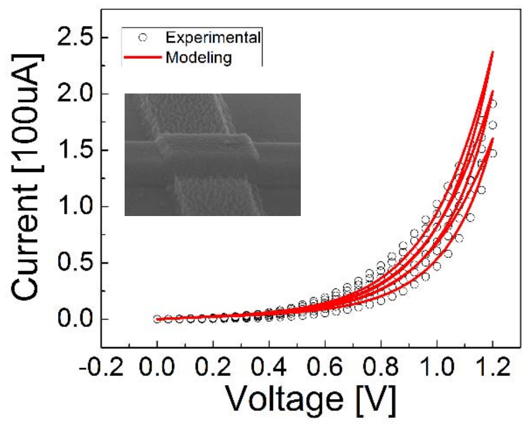


Fig. 1. Experimental data fitted by the memristor model. (a) Voltage and current plotted against time under three consecutive +1.2V sweeps. (b) Data from (a) replotted to show current versus voltage. Inset in (b) show SEM of actual device.

As shown in Fig. 1, the model can accurately predict the memristor behavior. For example, by applying consecutive positive DC sweeps, the device conductance has been gradually enhanced. The device can be similarly erased in an analog fashion.

III. Network Description

The network is conceptually arranged in a crossbar fashion with input units on the left hand side and outputs at the bottom as shown in Fig. 2(a). Each input neuron is connected to a row of the crossbar while an output neuron connects to a column. As will be discussed, the memristors connected to a given column form the receptive field of the output neuron at the bottom. The weights of these memristors, along with the neuron's threshold voltage, determine to which input patterns the output neuron is selective.

A. Input Neurons

Training images are mapped to input neurons with a 1-1 correspondence for each pixel. Each image is normalized to have zero mean and unit maximum: $p_i := \frac{p_i - \bar{p}}{\max(p - \bar{p})}$ where $\bar{p}$ is the average of all pixels. The normalized pixel values are used to determine the firing probability and voltage polarity of the input neurons. At each time-step of the simulation, a random number, r_i, is generated for the neuron, and if $r_i \leq |p_i|$, the neuron fires a voltage pulse of fixed amplitude. The sign of p_i determines the polarity of the pulse ($\mathrm{sgn}(p_i) = \mathrm{sgn}(V_i)$). The input pulse amplitudes are designed to generate current for the output neurons, but not be of such magnitude that, by themselves, cause significant change in the state of the connected memristors.

B. Crossbar Array

When an input neuron fires, it generates a voltage pulse on the row to which it is connected. This horizontal electrode serves as the shared top electrode for all memristors in that row. Each memristor passes a current that is approximately proportional to its current weight through to the bottom electrode it is connected to. Since all memristors in a given column share the same bottom electrode, currents from all of the input units, modulated by the weights of the co-columnar memristors, are summed and passed to the output neuron at the bottom of the column. Since the state variable responses very non-linearly to the input voltage (1), the device allows input pulses to pass current through the memristor without significantly altering the stored weights when the amplitude of the input pulse is kept low. During training, however, an output neuron will generate a back-propagating pulse to the bottom electrode of the memristor, increasing the total voltage drop across the memristor and allowing the network to learn as described in section III-D.

C. Output Neurons

The output neurons are modeled using the leaky integrate and fire model (LIF). Each neuron integrates the currents it receives from its connected electrode. The current charges the neuron's 'membrane potential', modeled as a capacitor in series with a resistor which provides the leak term. Each output neuron also has a variable threshold potential that determines when the neuron will fire. If the membrane potential of a neuron reaches its threshold, the neuron fires and emits a dual polarity voltage pulse on its electrode; no current is integrated for any neuron during this time. The neuron subsequently resets the membrane potentials of all neurons (itself included). In doing so, the network implements a winner-take-all strategy which is a computationally power paradigm that can be used for unsupervised learning [6]. The output neurons also implement a homeostatic condition wherein each neuron attempts to achieve a target average firing frequency. By adjusting its threshold potential, each neuron can control how sensitive it is to input current and, and thus, how often it will fire. The homeostasis requirement is argued to improve network performance by encouraging all neurons participate and ensuring that no neuron dominates by becoming sensitive to all input patterns [3,7]. Instead neurons must specialize and become sensitive to a limited set of features within the input.

D. Learning

Learning in the network occurs when an output neuron fires a dual polarity pulse to its electrode. This electrode is shared as the bottom electrode by all neurons in a given column. The first half of the pulse has a negative polarity which, when applied to the bottom electrode of the memristor, increases w, lowering its resistance, while the second half has a positive polarity which will decrease w, increasing its resistance (Fig. 2(b)). If an input neuron is simultaneously sending a positive polarity pulse to the memristor's top electrode, the temporal overlap of the two pulses has a strengthening effect on the memristor during the first half of the output neuron's pulse, while reducing the weakening effect of the second half. The net effect is an overall strengthening of the device. If, on the other hand, an input neuron is sending a negative pulse, the strengthening phase is reduced, while the weakening phase is enhanced for an overall net decrease in memristor weight. This effect is verified through the memristor model, where the potentiation and depression cases are plotted in Fig. 2(c). The input-dependent change in the state variable is made possible by the non-linear relationship between voltage and state variable derivative described in (1).

The crossbar array was initialized by setting the memristor weights using a Gaussian distribution with parameters μ=0.5 and σ^2=0.1. After training, the weights take on a bimodal distribution with the majority of weights near either 0 or 1 as shown in Fig. 3, due to repeated potentiation and depression of the weights as the neurons develop specific receptive fields during the unsupervised training process.

IV. Handwritten Digit Recognition

Recognizing handwritten characters is a classic machine learning problem with practical applications that has been studied extensively [8]. In order to test our network, the MNIST database of handwritten digits was used for training and validation. The database consists of 60,000 labeled training and 10,000 test images of handwritten digits (0–9). Each sample is a 28x28 pixel gray scale image of a single digit. For input into our network, each training image was reshaped into a 784-valued column, and normalized to have zero mean and unit maximum. During validation, the same setup was used, and the indices of the firing neurons were recorded. To make the classification, the label outputs of the receptive fields of the winning neurons were averaged, weighted by their firing counts, and the maximum output strength was chosen as the label. Figure 4(a) shows a random sample of the 10 receptive field from a network with 50 output neurons. As expected, as the number of output neurons increases, the recognition rate of the network increases as shown in Figure 4(b). With more neurons, there is a greater chance that there exists a receptive field, that is similar to a given test image; this allows the network to dedicate neurons to identifying specific forms of written digits (e.g. connected versus open top glyphs for the number four). While the network's correct classification rate is lower than state of the art techniques [9], this simulation serves as a proof of concept using a realistic device model; further improvements are expected to obtain competitive results.

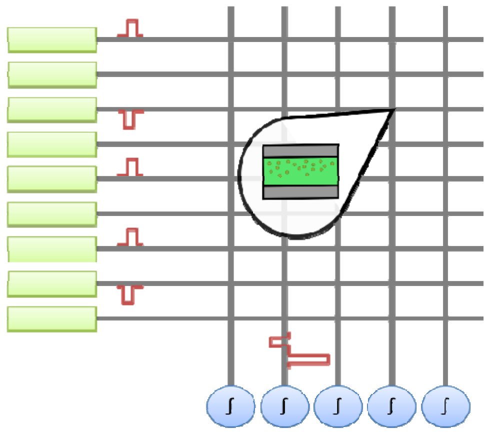
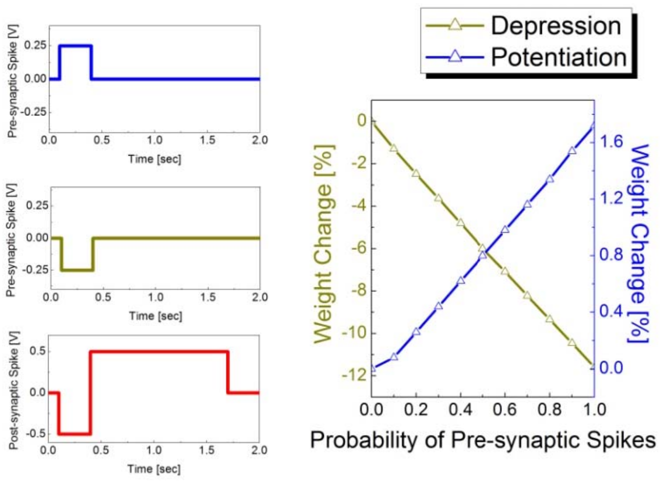



Fig. 2. Network topology (a showing input neurons (green rectangles), output neurons (blue circles) and memristor crossbar array (in grey). (b) Pre-synaptic and post-synaptic pulses, and (c) their temporal overlap alters the crosspoint device's resistance.

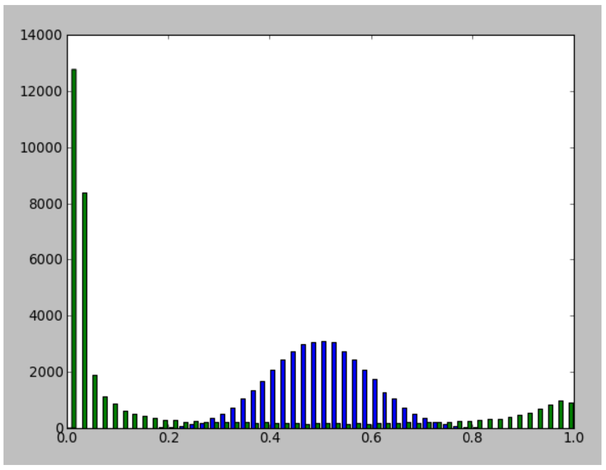
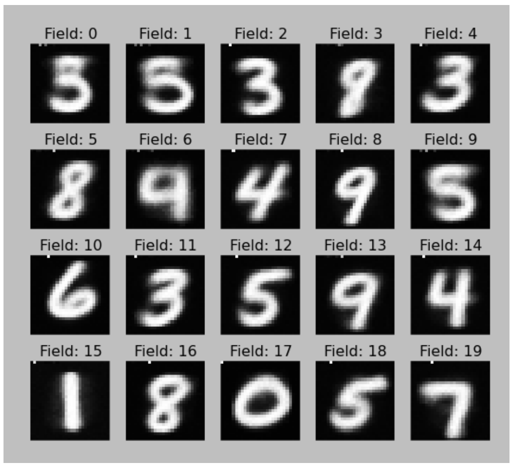


Fig. 3. Distribution of weights before (blue) and after (green) training.



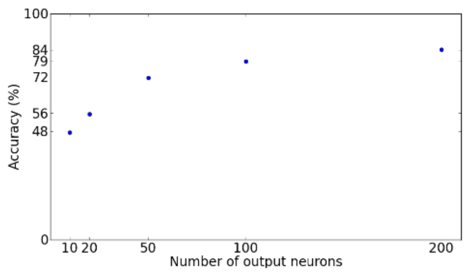


Fig. 4. (a) Sample of 20 receptive fields (from a network with 50 output neurons). (b) Recognition rate as a function of the number of output neurons.

V. Methods

The devices used for tuning the model were fabricated by sputter depositing a layer of tungsten on Si/SiO2 carrier wafers. The bottom electrode was defined with electron-beam lithography and tungsten etch. The bottom electrode was then partially oxidized in a rapid thermal annealing furnace to create the WOx switching layer. The top electrode was deposited by evaporation and liftoff. The device model was developed and tested using LTSpice, while the network simulations were written in Python.

VI. Discussion

The algorithm works by forming and continually refining prototypes for the input classes. Each time an input is shown to the network, the neuron with the most similar prototype will fire, effectively declaring its belief that the input belongs to a certain class. The dual-polarity pulse emitted by the firing neuron has the effect of moving the neuron's prototype (or receptive field) closer to the input by a small amount. This can be most easily visualized by considering a two-dimensional input as shown in Fig. 5. The prototypes of the network are shown as solid red dots located on the unit circle (it should be noted that the memristive weights in the simulation were not renormalized after each training sample, but it was found that the L_2norm of the weight vectors varied by less than 2% after training), and the input is depicted as a green arrow. Because the input is normalized to unit length it will lie on the circle as well. The network has selected the closest prototype, highlighted in yellow as the winner. The winning neuron's prototype is then moved in the direction of the input, indicated by the dashed red circle in the figure. In the case of the MNIST database, the inputs lie on a 784-dimensional sphere, but the algorithm proceeds in a similar manner.

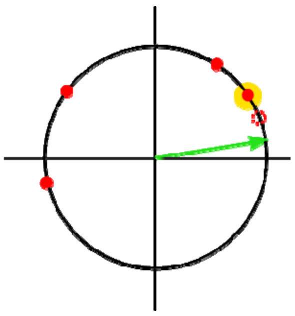


Fig. 5. Visualization of learning vector quantization algorithm. Prototypes are indicated as red dots, while the input is shown as a green arrow.

Using this method, the neurons acts as fuzzy classifiers [10], indicating the extent to which an input belongs in each of the label classes. Thus, a given neuron should not be considered as identifying a single label (e.g., a 'six neuron'); rather, the output neurons act as feature detectors, where features may be found it multiple digits. Indeed, it can be seen from the receptive fields that some neurons are ambiguous in their label association. For example, Field 0 in Fig. 4(a) indicates that the neuron fires strongly for inputs that have a wide, left-opened loop at the bottom. The ambiguity in the upper half of the receptive field is reflected in the distribution presented during classification – when this neuron fires, it indicates that the input has features strongly associated with handwritten 3's and 5's. Ultimately, a single, definitive class label is determined by weighting the neurons' label distributions with their firing counts and then choosing the maximum.

The results of the simulation show that learning vector quantization can be effectively implemented in a memristive array for unsupervised machine learning tasks. The use of resistive switching crossbar arrays allows dense vector storage and parallel execution of comparison operations. Further, the ability to couple resistive switching arrays with CMOS transistors in a back-end-of-line process [11] offers a powerful computing substrate that could enhance machine learning implementations in the future. Additional research is needed to characterize fabricated device variability and its impact on algorithm performance. A detailed analysis of network operation and supporting circuitry is also needed to characterize the energy savings potential of this approach.

VII. Conclusion

In this paper, we demonstrated a pattern recognition algorithm implemented in a WO_x-based memristor array. Image samples of handwritten digits are converted to voltage pulses and the network is able to successfully determine the digit represented based on the firing counts of output neurons. The network learns features of the input in an unsupervised manner. A winner-take-all approach is used to select the neuron most similar to a given input, and overlapping voltage pulses are used to adjust the receptive field of the winning neuron.

While handwritten digit recognition was used as an example, the data clustering algorithm shown here is not limited to this task and there are many more machine learning algorithms, supervised and unsupervised, which can be implemented using arrays of memristors. The analog tunability of WO_x, combined with their low-power characteristics and high packing density, make these devices an attractive candidate for use implementing these algorithms in hardware.

Acknowledgment

The authors would like to thank Dr. G. Kenyon, Dr. Z. Zhang and Dr. C. Teuscher for helpful discussions. The views, opinions, and/or findings contained in this paper are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of the DARPA or the Department of Defense. Approved for Public Release. Distribution Unlimited.

References

[1] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 22, no. 1, pp. 4–37, 2000.

[2] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, "Immunity to Device Variations in a Spiking Neural Network With Memristive Nanodevices," IEEE Transactions on Nanotechnology, vol. 12, no. 3, pp. 288–295, May 2013.

[3] J. Zylberberg, J. T. Murphy, and M. R. DeWeese, "A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields," PLoS computational biology, vol. 7, no. 10, p. e1002250, 2011.

[4] C. Zamarreño Ramos, L. A. Camuñas Mesa, J. A. Pérez-Carrasco, T. Masquelier, T. Serrano-Gotarredona, and B. Linares-Barranco, "On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex," Frontiers in neuroscience, vol. 5, 2011.

[5] T. Chang, S.-H. Jo, K.-H. Kim, P. Sheridan, S. Gaba, and W. Lu, "Synaptic behaviors and modeling of a metal oxide memristive device," Applied Physics A, vol. 102, no. 4, pp. 857–863, 2011.

[6] W. Maass, "On the computational power of winner-take-all," Neural Computation, vol. 12, no. 11, pp. 2519–2535, 2000.

[7] D. Querlioz, O. Bichler, and C. Gamrat, "Simulation of a memristorbased spiking neural network immune to device variations," in Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011, pp. 1775–1781.

[8] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: a case study in handwritten digit recognition," in Proceedings of the 12th IAPR International Conference on Pattern Recognition (Cat. No.94CH3440- 5), vol. 2. IEEE Comput. Soc. Press, pp. 77–82.

[9] D. Keysers, "Comparison and combination of state-of-the-art techniques for handwritten character recognition: topping the mnist benchmark," arXiv preprint arXiv:0710.2231, 2007.

[10] Z. Chi, J. Wu, and H. Yan, "Handwritten numeral recognition using self-organizing maps and fuzzy rules," Pattern Recognition, vol. 28, no. 1, pp. 59–66, 1995.

[11] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, "A Functional Hybrid Memristor Crossbar-Array/CMOS System for Data Storage and Neuromorphic Applications", Nano Lett., vol. 12, pp. 389–395, 2012.