United States Patent 12387104
Kind Code B2
Date of Patent August 12, 2025
Inventor(s) Kendall; Jack D. et al.

Deep learning in bipartite memristive networks

Abstract

A bipartite memristive network and method of teaching such a network is described herein. In one example case, the memristive network can include a number of nanofibers, wherein each nanofiber comprises a metallic core and a memristive shell. The memristive network can also include a number of electrodes deposited upon the nanofibers. A first set of the number of electrodes can include input electrodes in the memristive network, and a second set of the number of electrodes can include output electrodes in the memristive network. The memristive network can be embodied as a bipartite memristive network and trained according to the method of teaching described herein.

Inventors: **Kendall; Jack D. (Gainesville, FL), Nino; Juan C. (Gainesville, FL), Suarez; Laura E. (Bogota, CO)**

Applicant: **University of Florida Research Foundation, Inc.** (Gainesville, FL)

Family ID: **1000008748438**

Assignee: **UNIVERSITY OF FLORIDA RESEARCH FOUNDATION, INC. (Gainesville, FL)**

Appl. No.: **18/138984**

Filed: **April 25, 2023**

Prior Publication Data

Document Identifier	Publication Date
US 20230297839 A1	Sep. 21, 2023

Related U.S. Application Data

continuation parent-doc US 15985212 20180521 ABANDONED child-doc US 18138984
us-provisional-application US 62509423 20170522

Publication Classification

Int. Cl.: **G06N3/08** (20230101); **G06N3/049** (20230101); **G06N3/065** (20230101); **G06N3/084** (20230101); **G06N5/046** (20230101)

U.S. Cl.:

CPC **G06N3/084** (20130101); **G06N3/049** (20130101); **G06N3/065** (20230101); **G06N5/046** (20130101);

Field of Classification Search

CPC: G06N (3/084); G06N (3/049); G06N (3/065); G06N (5/046); G11C (2013/0073); G11C (13/0069)

References Cited

U.S. PATENT DOCUMENTS

Patent No.	Issued Date	Patentee Name	U.S. Cl.	CPC
8930291	12/2014	Srinivasa et al.	N/A	N/A
9418331	12/2015	Gupta et al.	N/A	N/A
9646243	12/2016	Gokmen	N/A	N/A
9691479	12/2016	Merced Grafals	N/A	G11C 13/004
9847132	12/2016	Zheng	N/A	G11C 13/0069
2004/0091264	12/2003	Beshai	398/47	H04Q 11/0005
2004/0153426	12/2003	Nugent	N/A	N/A
2007/0291524	12/2006	Davis	365/49.17	B82Y 10/00
2010/0081958	12/2009	She	N/A	N/A
2011/0096589	12/2010	Bratkovski et al.	N/A	N/A
2012/0011088	12/2011	Aparin et al.	N/A	N/A
2012/0011090	12/2011	Tang et al.	N/A	N/A
2012/0011093	12/2011	Aparin et al.	N/A	N/A
2012/0194967	12/2011	Keane et al.	N/A	N/A
2014/0151623	12/2013	Jeon et al.	N/A	N/A
2014/0156574	12/2013	Piekniewski et al.	N/A	N/A
2014/0156576	12/2013	Nugent	N/A	N/A
2015/0106316	12/2014	Birdwell et al.	N/A	N/A
2015/0269483	12/2014	Nishitani	706/25	G06N 3/065
2016/0004960	12/2015	Saïghi et al.	N/A	N/A
2016/0275395	12/2015	Amir et al.	N/A	N/A
2017/0011290	12/2016	Taha	N/A	G06N 3/065
2017/0017877	12/2016	Kataeva	N/A	G06N 3/08
2017/0017879	12/2016	Kataeva et al.	N/A	N/A
2017/0098156	12/2016	Nino et al.	N/A	N/A

FOREIGN PATENT DOCUMENTS

Patent No.	Application Date	Country	CPC
104779950	12/2014	CN	N/A
105160401	12/2014	CN	N/A
H07282168	12/1994	JP	N/A
H09185596	12/1996	JP	N/A
2013534676	12/2012	JP	N/A
1020170019414	12/2016	KR	N/A

OTHER PUBLICATIONS

Nguyen et al., Electrospun nanofibers with a core-shell structure of silicon nanoparticles and carbon nanotubes in carbon for use as lithium-ion battery anodes, Jul. 2014, 8 pages (Year: 2014). cited by examiner

CN Office Action for CN 2018800311967 mailed Jan. 19, 2023. cited by applicant

PCT Patent Application PCT/US2018/033669 filed on May 21, 2018, International Search Report and Written Opinion mailed on Aug. 6, 2018. cited by applicant

PCT Patent Application PCT/US2017/058757 filed on Oct. 27, 2017, International Search Report and Written Opinion mailed on Jan. 18, 2018. cited by applicant

PCT Patent Application PCT/US2017/058816 filed on Oct. 27, 2017, International Search Report and Written Opinion mailed on Jan. 26, 2018. cited by applicant

European Patent Application EP15810294.7 filed on Jun. 5, 2015, Supplementary European Search Report mailed on Jan. 26, 2018. cited by applicant

Deng et al., Deep convex net: A scalable architecture for speech pattern classification. Proceedings of the Annual Conference of the International Speech Communication Association, Interspeech, 2285-2288 (Aug. 2011). cited by applicant

Grishin et al., Nb2O5 nanofiber memristor, Applied Physics Letters, 103(5), 053111 (Jul. 2013). cited by applicant

Hasler et al., Finding a roadmap to achieve large neuromorphic hardware systems, Frontiers in Neuroscience, 7, 118 (Sep. 2013). cited by applicant

Hutchinson et al., Tensor deep stacking networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), 1944-1957 (Aug. 2013). cited by applicant

Izhikevich et al., Large-scale model of mammalian thalamocortical systems. Proceedings of the National Academy of Sciences of the United States of America, 105(9), 3593-8 (Mar. 2008). cited by applicant

Jo et al., Nanoscale memristor device as synapse in neuromorphic systems, Nano Letters, 10(4), 1297-1301 (Mar. 2010). cited by applicant

Li et al., Fabrication of Titania Nanofibers by Electrospinning. Nano Letters, 3(4), 555-560 (Apr. 2003). cited by applicant

Manan Suri et al.; “Bio-Inspired Stochastic Computing Using Binary CBRAM Synapses”; IEEE Transactions on Electron Devices; vol. 60. No. 7; Jun. 4, 2013; pp. 2402-2409. cited by applicant

Merolla et al., A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm. Custom Integrated Circuits Conference (CICC), 2011 IEEE. IEEE, (Sep. 2011). cited by applicant

Moghe et al., Co-axial Electrospinning for Nanofiber Structures: Preparation and Applications, Polymer Reviews, Polymer Reviews 48.2, 353-377 (May 2008). cited by applicant

Naderi, M. A., Analysis and classification of EEG signals using spectral analysis and recurrent neural networks, Biomedical Engineering (ICBME), 2010 17th Iranian Conference of. IEEE, (Nov. 2010). cited by applicant

Strukov et al., The missing memristor found, Nature, 453(7191), 80-3, (May 2008). cited by applicant

Waser et al., Redox-Based Resistive Switching Memories—Nanoionic Mechanisms, Prospects, and Challenges, Advanced Materials, 21(25-26), 2632-2663 (Jul. 2009). cited by applicant
Williams, R. S. (n.d.), How We Found the Missing Memristor, Spectrum, IEEE, 45(12), 28-35 (Dec. 2008). cited by applicant
Zhi et al., Highly conductive electrospun carbon nanofiber/MnO2 coaxial nano-cables for high energy and power density supercapacitors, Journal of Power Sources, 208, 345-353 (Jun. 2012). cited by applicant
Zidan et al., Single-Readout High-Density Memristor Crossbar, Scientific reports 6, 2-10 (Jan. 2016). cited by applicant
Astaras, A. "Pulse-Stream Binary Stochastic Hardware for Neural Computation: The Helmoltz Machine." A Thesis Submitted for the Degree of Doctor of Philosophy, The University of Edinburgh. (pp. 1-40, 150-176), Sep. 13, 2004. [online] [retrieved on Dec. 27, 2017 (Dec. 27, 2017)] Retrieved from the Internet: URL: https://www.era.lib.ed.ac.uk/handle/1842/11620, entire document, especially Abstract; Figure 4, 5; pp. 23-33, 154-155, 172. cited by applicant
Ben et al., Neurogrid : A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations. Proceedings of the IEEE 102.5, 699-716 May 2014). cited by applicant
Kee et al. "Feed-Forward versus Feedback Inhibition in a Basic Olfactory Circuit" In: PLoS Comput Biol 11(10):, Oct. 15, 2015 [online] [retrieved on Dec. 27, 2017 (Dec. 27, 2017)] Retrieved from the Internet: URL: http://journals.plos.org/p1oscompbiol/article?id=10.1371/journal.pcbi.1004531, entire document. cited by applicant
Pantazi et al. "All-Memristive Neuromorphic Computing With Level-Tuned Neurons" In: Nanotechnology 27 (2016), Jul. 26, 2016.[online] [retrieved on Dec. 27, 2017 (Dec. 27, 2017)] Retrieved from the Internet: URL: http://iopscience.iop.org/article/10.1088/0957-4484/27/35/355205/meta, entire document. cited by applicant
Burger et al., Hierarchical composition of memristive networks for real-time computing. Nanoscale Architectures (NANOARCH), 2015 IEEE/ACM International Symposium on. IEEE, (Jul. 2015). cited by applicant
Chan et al., Spike timing dependent plasticity with memristive synapse in neuromorphic systems, The 2012 International Joint Conference on Neural Networks (IJCNN), 1-6 (Jun. 2012). cited by applicant
A. Carnell, "An analysis of the use of Hebbian and Anti-Hebbian spike time dependent plasticity learning functions within the context of recurrent spiking neural networks," Neurocomputing, vol. 72, No. 4-6, pp. 685-692, Jan. 2009. cited by applicant
A. E. J. Senador, M. T. Shaw, and P. T. Mather, "Electrospinning of Polymeric Nanofibers: Analysis of Jet Formation," Mat. Res. Soc. Symp. Proc., 2001, vol. 661. cited by applicant
A. Joubert, B. Belhadj, O. Temam, and M. Campus, "Hardware spiking neurons design?: analog or digital??" (2012). cited by applicant
K. Szot, M. Rogala, W. Speier, Z. Klusek, a Besmehn, and R. Waser, "TiO2—a prototypical memristive material.," Nanotechnology, vol. 22, No. 25, p. 254001, Jun. 2011. cited by applicant
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.," Front. Neurosci., vol. 5, No. March, p. 26, Jan. 2011. cited by applicant
Chronakis, I. S. (2005). Novel nanocomposites and nanoceramics based on polymer nanofibers using electrospinning process—A review. Journal of Materials Processing Technology, 167(2-3), 283-293. doi:10.1016/j.matprotec.2005.06.053. cited by applicant
J. Matser, "Structured Liquids in Liquid State Machines," Utrecht University, 2010. cited by applicant
J. Sundaramurthy, V. Aravindan, P. S. Kumar, S. Madhavi, and S. Ramakrishna, "Electrospun TiO 2-d Nanofibers as Insertion Anode for Li-Ion Battery Applications," J. Phys. Chem. C, 2014. cited

by applicant
Tran, T. “Simulations of Artificial Neural Network With Memristive Devices”; Boise State University; Publication [online]. Dec. 2012 [retrieved Jul. 14, 2018]. cited by applicant
W. Maass, T. Natschläger, and H. Markram, “Real-time computing without stable states: a new framework for neural computation based on perturbations.,” Neural Comput., vol. 14, No. 11, pp. 2531-2560, Nov. 2002. cited by applicant
Yang, J. J., Pickett, M. D., Li, X., Ohlberg, D. A., Stewart, D. R., & Williams, R. S. (2008). Memristive switching mechanism for metal/oxide/metal nanodevices. Nature Nanotechnology, 3(7), 429-433. doi:10.1038/nnano.2008.160. cited by applicant
Z. Biolek, D. Biolek and V. Biolková, “SPICE Model of Memristor with Nonlinear Dopant Drift,” Radioengineering, vol. 18 (2) pp. 210-214. cited by applicant
Buzsaki, G. (2006). Rhythms of the Brain. Oxford University Press. cited by applicant
Japanese Patent Application JP2016-573557 filed on Jun. 5, 2015, Office Action mailed on Sep. 11, 2018. cited by applicant
Nicole Casal Moore, College of Engineering, image processing 1,000 times faster is goal of new $5M contact, U-M Press Release, USA, University of Michigan, Aug. 14, 2013. URL, http://www.eecs.umich.edu/eecs/about/articles/2013/Lu-image-processing-1000-times-faster.html. cited by applicant
European Search Report and Written Opinion for European U.S. Appl. No. 18/805,503 mailed Mar. 10, 2021. cited by applicant
European Examination Report for Application No. 2018273847 mailed Mar. 4, 2022. cited by applicant
Alibart, et al., “Pattern classification by memristive crossbar circuits using ex situ and in situ training”, Nature Communications, Jun. 25, 2013. cited by applicant
Korean Office Action for Korean Patent Application No. 10-2019-7037028 mailed Dec. 29, 2022. cited by applicant

---

Primary Examiner: Fernandez Rivas; Omar F

Assistant Examiner: Nguyen; Tri T

Attorney, Agent or Firm: Thomas | Horstemeyer, LLP

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS (1) This application is a continuation claiming priority to, and the benefit of, U.S. Ser. No. 15/985,212, filed May 21, 2018, which claims the benefit of U.S. Provisional Application No. 62/509,423, filed May 22, 2017, the entire contents of both are hereby incorporated herein by reference. This application is also related to U.S. Non-Provisional patent application Ser. No. 15/383,527, titled “MEMRISTIVE NANOFIBER NEURAL NETWORKS,” the entire contents of which is hereby incorporated herein by reference.

BACKGROUND
(1) A memristor is a passive non-linear two-terminal electrical component. In a memristor, the electrical resistance of the device depends on a history of current that previously flowed through it (or voltage applied across it). Thus, the resistance of a memristor can be changed incrementally by placing a voltage differential or drop across it. When this voltage drop is removed, the memristor retains its current value of resistance. If the polarity of the voltage drop is reversed, the resistance will change in the opposite direction.

(2) In certain circuit configurations, a memristor can act as an artificial synapse with a weight equivalent to its conductance. The strength of the weight can be modified over time by controlling the voltage drop across the memristor. Memristors typically have a threshold voltage drop, below which no change in conductance will occur.
(3) The use of memristors for synaptic networks has been proposed as a promising approach in the field of neuromorphic circuit development. Memristors have been incorporated into crossbar synapse networks with complementary metal oxide semiconductor (CMOS)-based circuitry for presynaptic and postsynaptic voltage control. Those circuits demonstrate that CMOS-based voltage and current control can be used to generate changes in the resistance or impedance (e.g., memory) of memristors.
(4) In a memristive neural network, inputs can be presented as voltages to a set of electrodes in a network of memristors. At a separate set of electrodes in the network, voltages or currents are read out. These are taken to be the outputs of the neural network. The goal of learning in such a memristive neural network is to change the conductances of the memristors in order to minimize some cost or error function.
SUMMARY
(5) One example embodiment includes a method to train a memristive network having a number of input nodes and a number of output nodes. The memristive network can include a bipartite memristive network, and the method can reproduce a backpropagation algorithm for training the memristive network.
(6) In one example, the method includes applying an input voltage or current to an input node among the number of input nodes, grounding an output node among the number of output nodes, measuring an output current or voltage at the output node, comparing the output current or voltage to a target current or voltage to determine an error delta, and applying a threshold voltage or current to the output node for a time period proportional to a magnitude of the error delta.
(7) When the error delta is negative, the method can also include applying a positive threshold voltage or current to the output node for the time period proportional to the error delta, and applying a negative threshold voltage or current to the output node for the time period proportional to the error delta. When the error delta is positive, the method can include reversing a polarity of the input voltage or current applied to the input node, applying a positive threshold voltage or current to the output node for the time period proportional to the error delta, and applying a negative threshold voltage or current to the output node for the time period proportional to the error delta.
(8) In other aspects, the method can further include transforming the error delta into an error delta voltage or current, applying the error delta voltage or current to the output node, and applying the threshold voltage or current to the input node for a second time period proportional to an absolute value of the error delta voltage or current.
(9) When the input voltage or current applied to the input node was positive, the method can include applying a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current, and applying a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current. When the input voltage or current applied to the input node was negative, the method can further include reversing a polarity of the error delta voltage or current applied to the output node, applying a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current, and applying a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage.
(10) In another embodiment, a memristive network includes a number of nanofibers, wherein each nanofiber comprises a metallic core and a memristive shell, a number of electrodes deposited upon the nanofibers, wherein the number of electrodes comprise a number of input nodes and a number

of output nodes, and a training processor.
(11) The training processor can be configured to apply an input voltage or current to an input node among the number of input nodes, ground an output node among the number of output nodes, measure an output current or voltage at the output node, compare the output current or voltage to a target current or voltage to determine an error delta, and apply a threshold voltage or current to the output node for a time period proportional to a magnitude of the error delta.

Description

BRIEF DESCRIPTION OF THE DRAWINGS
(1) Many aspects of the present disclosure can be better understood with reference to the following drawings. The components in the drawings are not necessarily drawn to scale, with emphasis instead being placed upon clearly illustrating the principles of the disclosure. In the drawings, like reference numerals designate corresponding parts throughout the several views.
(2) FIG. **1**A illustrates an example network architecture of a memristive nanofiber neural network according to various embodiments described herein.
(3) FIG. **1**B illustrates example connections between input and output electrodes and memristive nanofibers in the neural network shown in FIG. **1**A according to various embodiments described herein.
(4) FIG. **1**C illustrates an example equivalent circuit of a portion of the neural network shown in FIG. **1**A according to various embodiments described herein.
(5) FIG. **2** illustrates an example bipartite memristive network and an alternate feedforward form of the bipartite memristive network with inputs and outputs segregated according to various embodiments described herein.
(6) FIG. **3** illustrates an example linear model of a memristor with a threshold V.sub.T according to various embodiments described herein.
(7) FIG. **4** illustrates an example training processor for training the feedforward network shown in FIG. **2** according to various embodiments described herein.
(8) FIG. **5**A illustrates an example memristive network training process according to various embodiments described herein.
(9) FIG. **5**B further illustrates the example memristive network training process shown in FIG. **5**A according to various embodiments described herein.
(10) FIG. **5**C further illustrates the example memristive network training process shown in FIG. **5**A according to various embodiments described herein.
DETAILED DESCRIPTION
(11) As noted above, a memristor can act as an artificial synapse with a weight equivalent to its conductance. The strength of the weight can be modified over time by controlling the voltage drop across the memristor. Memristors typically have a threshold voltage drop, below which no change in conductance will occur.
(12) In a memristive neural network, inputs can be presented as voltages to a set of electrodes in a network of memristors. At a separate set of electrodes in the network, voltages or currents are read out. These are taken to be the outputs of the neural network. The goal of learning in such a memristive neural network is to change the conductances of the memristors in order to minimize some cost or error function.
(13) The concepts described herein are directed to a bipartite memristive network and method of teaching such a network. In one example case, the memristive network can include a number of nanofibers, wherein each nanofiber comprises a conductive or metallic core and a memristive shell. The memristive network can also include a number of electrodes deposited upon the nanofibers. A first set of the number of electrodes can include input electrodes in the memristive network, and a

second set of the number of electrodes can include output electrodes in the memristive network. The memristive network can be embodied as a bipartite memristive network and trained according to the method of teaching described herein. The concepts described herein are applicable to other forms and structures of bipartite memristive networks (e.g., other than using nanofibers). Further, while the figures illustrate a number of example networks, electrodes, etc., the concepts described herein can be applied to other forms, structures, and sizes of bipartite memristive networks.
(14) FIG. **1**A illustrates an example network architecture of a memristive nanofiber neural network **10** according to various embodiments described herein. As described in further detail below, the neural network **10** takes the form of a bipartite memristive network. In the neural network **10**, a number of memristive nanofibers **12** ("nanofibers **12**" or "nanofiber **12**") are randomly aligned on a substrate, and a set of input/output electrodes **14** are deposited on top of the nanofibers **12**.
(15) FIG. **1**B illustrates example connections between the input/output electrodes **14** and the memristive nanofibers **12** in the neural network **10** shown in FIG. **1**A. As shown in FIG. **1**B, each of the nanofibers **12** includes a conductive or metallic core **12**A and a memristive shell **12**B. Further, at each junction between an electrode **14** and a nanofiber **12**, a memristive synapse **16** is formed.
(16) The neural network **10** can be formed in any suitable manner, such as the method described in U.S. Non-Provisional patent application Ser. No. 15/383,527, titled "MEMRISTIVE NANOFIBER NEURAL NETWORKS," the entire contents of which is hereby incorporated herein by reference. For example, the conductive core **12**A of the nanofibers **12** can be formed from doped titanium dioxide (TiO.sub.2), using a percursor of titanium isopropoxide, titanium butoxide, or another suitable precursor. The memristive shell **12**B can be formed around the conductive metallic core **12**A by electrospinning the outer shell of the conductive core **12**A with a memristive material.
(17) FIG. **1**C illustrates an example equivalent circuit of a portion of the neural network **10** shown in FIG. **1**A. The interface resistance between two of the nanofibers **12**, when overlapping and contacting each other in the neural network **10**, can be much higher than the resistance between an electrode **14** and a nanofiber **12**. This is because the contact area between two of the nanofibers **12** is typically very small, and the contact point is between two insulating memristive shells **12**B. In contrast, each electrode **14** contacts a longer length or section of the memristive shell **12**B of each nanofiber **12**, and the voltage drop is across only a single memristive shell **12**B, rather than two memristive shells **12**B between two nanofibers **12**.
(18) FIG. **2** illustrates an example bipartite memristive network **20** and an alternate feedforward form **30** of the bipartite memristive network **20**. The electrodes **14** in FIG. **1**A, which are referred to as external electrodes in FIG. **2**, are connected to the conductive metallic cores **12**A of the nanofibers **12** through the memristive shells **12**B. The electrodes **14** are not directly connected to one another. The conductive metallic cores **12**A of the nanofibers **12**, which are referred to as internal electrodes, are connected to the external electrodes **14** through the memristive shells **12**B.
(19) In the alternate feedforward form **30** of the bipartite memristive network **20**, the electrodes **14** are separated into a number of input electrodes **14**A and output electrodes **14**B. If it is assumed that the nanofibers **12** are effectively insulated or isolated from one another electrically, the feedforward form **30** of the bipartite memristive network **20** will take the form of a bipartite graph as shown.
(20) Thus, if the external electrodes **14** are divided or segregated into the input electrodes **14**A and the output electrodes **14**B, the alternate feedforward form **30** of the bipartite memristive network **20** can be operated as a three-layer feedforward neural network. If more hidden layers are needed, several such networks (e.g., similar to the feedforward form **30**) can be linked together. An approach to teach the alternate feedforward form **30** (also the "the feedforward network **30**") of the bipartite memristive network **20** described in further detail below.
(21) In order to determine how the feedforward network **30** behaves, Kirchoff's laws can be used to determine the voltage drops across the memristive shells **12**B (or simply "memristors" as described below). Once the voltage drops and currents through the feedforward network **30** are identified, an

appropriate memristor model can be applied to determine how to change the conductances of these memristors over time.
(22) First, a procedure for changing the conductances of the output to internal node weights can be developed. If the input voltages to the feedforward network **30** are presented as voltages between
(23) $-\frac{1}{2}V_T$
and
(24) $\frac{1}{2}V_T$,
then it avoids changing any memristor conductances, since the maximum voltage drop across any memristor cannot exceed V.sub.T. Next, it can be assumed that the outputs of the feedforward network **30** are read as currents at the output nodes, where the voltages are kept at ground.
(25) Since the total current into any internal node must sum to zero (i.e., according to Kirchoff's current law), we have the following equation:
(26) $I_j = \text{.Math.}_{i=1}^{N} G_{ij}(V_i - V_j) = 0, \quad (1)$
where G.sub.ij is the instantaneous conductance of a memristor connecting external node i to internal node j and the sum runs over all external nodes (i.e., the input and output nodes).
(27) Equation (1) can be arranged to solve for the voltage at internal node j as shown below. Since summation is linear and V.sub.j is not indexed by i:
(28) $\text{.Math.}_{i=1}^{N} (G_{ij}V_i - G_{ij}V_j) = 0, \quad (2) \quad \text{.Math.}_{i=1}^{N} G_{ij}V_i = \text{.Math.}_{i=1}^{N} G_{ij}V_j, \quad (3)$ $\text{.Math.}_{i=1}^{N} G_{ij}V_i = V_j\ \text{.Math.}_{i=1}^{N} G_{ij}$, and $(4) \quad V_j = \frac{\text{.Math.}_{i=1}^{N} G_{ij}V_i}{\text{.Math.}_{i=1}^{N} G_{ij}} = \frac{\text{.Math.}_{i=1}^{N} G_{ij}V_i}{G_j}. \quad (5)$
(29) In Equation (5), the normalization factor, G.sub.j=Σ.sub.i=1.sup.NG.sub.ij, does not depend on the instantaneous values of the voltages or currents. The normalization factor G.sub.j does not significantly influence the operation of the feedforward network **30**. If all of the output nodes are left at ground (i.e., V.sub.i=0), the only contributions to the sum are those of the input nodes. Therefore, the voltages at the internal nodes can be considered a linear transformation of the input by a matrix of conductances divided by a normalization factor.
(30) Similarly, the currents at the output node k can be solved for using the voltages at the internal nodes. Since the output nodes are all grounded, the current is given by:
(31) $I_k = \text{.Math.}_{j=1}^{M} G_{kj}V_j. \quad (6)$
(32) This is a linear transformation of the hidden nodes by a matrix of conductances. Since this current is accessible by an external circuit, a further non-linear transformation can take place on this sum, allowing the network to possess a non-linear output layer. For simplicity in deriving the learning algorithm, however, all layers can be assumed to be linear.
(33) Now that an example three-layer neural network with memristive connections has been described and shown in the feedforward network **30**, an appropriate algorithm is needed to train such a network. For this purpose, an approximation to the backpropagation algorithm which is suitable for training the network is described.
(34) FIG. **3** illustrates an example linear model of a memristor with a threshold V.sub.T according to various embodiments described herein. To start, the weights between the internal and output nodes are considered since these weights require only the delta rule for learning. For linear neurons in this network, the delta rule is given as follows:
Δ*G*.sub.kj=α(*T*.sub.k−*I*.sub.k)*V*.sub.j, (7)
where T.sub.k is the desired output current, I.sub.k is the measured output current, V.sub.j is the applied voltage at internal node j, and a is the learning rate.
(35) Intuitively, if the error delta (T.sub.k-I.sub.k) is positive, it would be desirable to make the current I.sub.k more positive, thus decreasing the error. This can be accomplished by strengthening the connections to internal nodes which are positive, and weakening the connections to internal nodes which are negative. If the error delta is negative, the opposite should occur. The form of the

delta rule is such that it is linear in both the error delta and the activation of the internal nodes. If a linear neuron with a threshold V.sub.T is assumed, the following equation is of the form of the conductance change:

(36) $$\Delta G = \begin{cases} \alpha(\Delta V - V_T)t & \text{if } \Delta V > V_T \\ -\alpha(\Delta V + V_T)t & \text{if } \Delta V < -V_T \\ 0 & \text{if } \Delta V_T > \Delta V > -V_T \end{cases} . \quad (8)$$

(37) In Equation (8), α is analogous to the learning rate, ΔV is the voltage drop across the memristor, and t is the duration of the applied voltage. The graph of the above equation, not including duration, is given in FIG. **3**. Equation (8) is a simple model for a memristor with a threshold, so it can be used in deriving the learning algorithm. However, other memristor models, including those with noise, also work with the derived algorithm.
(38) Since the error delta (T.sub.k-I.sub.k) is available for measurement by an external circuit, the duration t can be set to be proportional to the magnitude of this error delta. The resulting change in conductance of the memristor G.sub.kj can also be made to be proportional to the voltage V.sub.j at internal node j as described below.
(39) The voltages at the internal nodes (e.g., the conductive cores **12**A) cannot be changed directly and may be unknown. However, the voltages and/or currents at the inputs and/or outputs can be directly measured. Since the outputs are all grounded initially, changing the voltage on a single output will perturb the voltages at the internal nodes. However, provided there are many inputs and outputs, this will result in only a small perturbation.
(40) If an output node is selected and voltage set equal to V.sub.T, the connections to all internal nodes with positive voltages (+V.sub.j) will not change their conductances because their voltage drop (V.sub.T-V.sub.j) will be, by definition, less than V.sub.T since V.sub.j is positive. This isolates the internal nodes with negative V.sub.j, whose voltage drops will be larger than V.sub.T. Also, the voltage drop across each negative internal node is now equal to:
Δ*V*=*V*.sub.T−*V*.sub.j,such that (9):
V.sub.j=−(Δ*V*−*V*.sub.T). (10)
(41) The quantity driving the conductance change, ΔG∝(ΔV−V.sub.T), is therefore proportional to V.sub.j. If the error delta is negative, the current should be more negative. Therefore, it would be desirable to strengthen the connection to the internal nodes with negative voltages. Since, in the above case, ΔV>V.sub.T, the application of V.sub.T to an output node would increase the conductances to all negative internal nodes by an amount proportional to the V.sub.j of each node. If it is assumed that this voltage is applied for a time t proportional to the error delta on this output node, the delta rule, ΔG.sub.kj=α(T.sub.k−I.sub.k) V.sub.j, can be reconstructed for this set of connections.
(42) This can be extended to the positive internal nodes connected to this output node by applying a voltage −V.sub.T after the application of +V.sub.T for time t. Now, the voltage drop (−V.sub.T−V.sub.j) across all internal nodes with negative voltages will be under the threshold for conductance change, effectively isolating the internal nodes with positive V.sub.j. For a similar reason, the magnitude of the conductance changes across these positive internal nodes are also proportional to the voltages at each internal node in the direction to decrease, rather than increase, the conductance. Again, this is what is required for an output node with negative error delta.
(43) To account for outputs with positive error delta, the directions of these conductance changes can be reversed. Specifically, connections to negative internal nodes can be weakened and connections to positive internal nodes can be strengthened. This can be accomplished by reversing the voltage polarities of all inputs (thus flipping the signs of all internal nodes), followed by the same procedure described above. This will flip the direction of the voltage drop across each memristor in each case.

(44) Next, a procedure for changing the conductances of the input to internal node weights can be developed. Since the input voltages are restricted to between
(45) $-\frac{1}{2}V_T$
and
(46) $\frac{1}{2}V_T$,
these weights could not have changed during the training of the output weights, as the maximum voltage drop from input node to internal node cannot exceed V_T.
(47) The backpropagation step of the backpropagation algorithm is given by the following equation:
$$\Delta G_{ij}=\alpha\left(\Sigma_k G_{kj}(T_k-I_k)\right)V_i. \qquad (11)$$
(48) Instead of the center term being proportional to the error delta, it is now equal to a weighted sum of error deltas. This is the internal node j's “contribution to the error”. The term on the right is the voltage at the input, which is accessible for measurement. The duration t can be made proportional to the absolute value of this input V_i.
(49) Now, the above weighted sum can be reproduced as a voltage on internal node j, and the same methods described above can be applied to change the conductances proportionally to this voltage.
(50) If the error deltas (T_k-I_k) are taken and converted into voltages, this weighted sum can be achieved by placing these voltages on the output nodes while grounding the input nodes. The resulting contribution to the error on each internal node is accurate up to a normalization factor G_j, which again does not significantly influence the learning process.
(51) By applying a voltage V_T to an input node with a positive input, it is possible to selectively strengthen the conductances to all nodes connected to it with a negative voltage, decreasing these internal nodes' contributions to the error. Similarly, applying a voltage $-V_T$ will selectively weaken the conductances to all nodes with positive voltage, again reducing these nodes' contributions to the error. Flipping the signs of all the error deltas, input nodes with negative input can be handled in the same way.
(52) FIG. **4** illustrates an example training processor **110** for training the feedforward network **30** shown in FIG. **2** according to various embodiments described herein. The training processor **110** includes a data store **120**, an interface module **130**, a measurement module **140**, and a training module **150**. The data store **120** stores various types of data, including the training data **122** and the network data **124**. The training data **122** can be relied upon by the training module **150** to train the feedforward network **30**. For example, the training data **122** can include data that defines the target weights between the internal nodes (e.g., the conductive metallic cores **12**A) in the feedforward network **30** and the output nodes (e.g., the input electrodes **14**A and/or the output electrodes **14**B). Thus, the training data **122** can define the target levels or values of conductance to be programmed into each of the memristive shells **12**B in the feedforward network **30**. The training data **122** can be based on any suitable model for training, for any purpose of training a network.
(53) The network data **124** can include data that defines the network structure of the feedforward network **30**, such as the connections between the conductive metallic cores **12**A, the input electrodes **14**A, and the output electrodes **14**B. Thus, the network data **124** can define the memristor connections formed by the memristive shells **12**B between the conductive metallic cores **12**A, the input electrodes **14**A, and the output electrodes **14**B shown in FIG. **4**. The training module **150** can reference the network data **124** to determine the appropriate sequence of steps to use for training the feedforward network **30**.
(54) The interface module **130** can include one or more circuits to provide an electrical interface with the input electrodes **14**A and the output electrodes **14**B. For example, the interface module **130** can include circuitry capable of providing input voltages and/or currents to one or more of the input electrodes **14**A and/or the output electrodes **14**B. The interface module **130** can include circuitry capable of grounding one or more of the input electrodes **14**A and/or the output electrodes **14**B. The interface module **130** can also include circuitry capable of reading voltages and/or

currents at one or more of the input electrodes **14**A and/or the output electrodes **14**B.
(55) The measurement module **140** can include one or more circuits to take measurements at one or more of the input electrodes **14**A and/or the output electrodes **14**B. For example, the measurement module **140** can measure voltages and/or currents at one or more of the input electrodes **14**A and/or the output electrodes **14**B. The measurement module **140** can also calculate the difference between various voltages, currents, etc., to determine the differences between measured and target values. Both the interface module **130** and the measurement module **140** can be embodied as any suitable network of analog and/or digital circuitry, including amplifiers, filters, and processing circuitry, with memory.
(56) The training module **150** can be embodied as processing circuitry capable of conducting the memristive network training process shown in FIGS. **5**A, **5**B, and **5**C. In that context, the training module **150** can direct the interface module **130** to train the conductances or weights between the input electrodes **14**A and the conductive cores **12**A of the internal nodes. The training module **150** can also direct the interface module **130** to train the conductances or weights between the conductive cores **12**A of the internal nodes and the output electrodes **14**B.
(57) In the practical sense, the internal structure of the feedforward network **30** (e.g., the individual connections and/or conductances of the memristive shell **12**B connections between the input electrodes **14**A, the output electrodes **14**B, and the conductive cores **12**A) may be unknown. Thus, the algorithm performed by the training module **150** is designed to perform the training process implicitly. Through training the feedforward network **30**, the training module **150** is effectively self-organizing a set of pathways between the input electrodes **14**A and the conductive cores **12**A, and between the output electrodes **14**B and the conductive cores **12**A. The pathways are defined or organized through adjusting the conductances of the memristive shells **12**B between the input electrodes **14**A, the output electrodes **14**B, and the conductive cores **12**A. In some cases, rather than using a set of target weights for training, the training module **150** can start with a set of target currents at the output electrodes **14**B.
(58) FIGS. **5**A-**5**C illustrate an example memristive network training process for training a feedforward network according to various embodiments described herein. The training process is described in connection with the feedforward network **30** and the training processor **110** shown in FIG. **4**. Thus, the training processor **110** is configured to perform the process shown in FIGS. **5**A-**5**C. However, other feedforward networks can be trained by other training processors using the process shown in FIGS. **5**A-**5**C. Further, although the process diagrams show an order of operation or execution, the order can differ from that which is shown. For example, the order of execution of two or more process steps can be switched relative to the order shown or as described below. Also, two or more process steps shown in succession can be executed concurrently or with partial concurrence. Further, in some examples, one or more of the process steps shown in the process diagrams can be skipped or omitted.
(59) Turning to FIG. **5**A, the training module **150** can first train the conductances or weights (e.g., the conductances of the memristors) the between the conductive cores **12**A of the internal nodes and the output electrodes **14**B in the feedforward network **30**. To that end, at step **202**, the process includes applying input power to one or more input nodes of a feedforward network. For example, the training module **150** can direct the interface module **130** to apply a voltage between
(60) $-\frac{1}{2}V_T$
and
(61) $0\frac{1}{2}V_T$
to one or more of the input electrodes **14**A of the feedforward network **30**. In some cases, the training module **150** can direct the interface module **130** to apply a voltage between
(62) $-\frac{1}{2}V_T$
and
(63) $\frac{1}{2}V_T$

to each of the input electrodes **14**A. As an alternative to the application of voltages, the training module **150** can direct the interface module **130** to apply or drive a certain level of current to one or more of the input electrodes **14**A.
(64) At step **204**, the process can include grounding one or more output nodes of the feedforward network. For example, the training module **150** can direct the interface module **130** to ground one or more of the output electrodes **14**B of the feedforward network **30**. In some cases, the training module **150** can direct the interface module **130** to ground each of the output electrodes **14**B.
(65) As noted above, in Equation (5), the normalization factor,
(66) $G_j = .\text{Math.}_{i=1}^{N} G_{ij}$,
does not depend on the instantaneous values of the voltages or currents. The normalization factor G.sub.j does not significantly influence the operation of the feedforward network **30**. If all of the output nodes are left at ground (i.e., V.sub.i=0), the only contributions to the sum are those of the input nodes. Therefore, the voltages at the internal nodes can be considered a linear transformation of the input by a matrix of conductances divided by a normalization factor.
(67) Similarly, the currents at the output electrodes **14**B can be solved for by the training module **150**. Since the output electrodes **14**B are all grounded at step **204**, the current can be given by Equation (6). As noted above, this is a linear transformation of the hidden nodes by a matrix of conductances. At the same time, the actual currents at the output electrodes **14**B are accessible by the interface module **130** for measurement by the measurement module **140**.
(68) At step **206**, the process can include the measurement module **140** measuring an output current at one or more of the output electrodes **14**B. The measurement module **140** can measure an output current each of the output electrodes **14**B, in turn, and store those values in the data store **120** for further processing. In addition to current, the measurement module **140** can also measure output voltages at one or more of the output electrodes **14**B at step **106** in some cases.
(69) At step **208**, the process can include the training module **150** comparing the output current (or voltage) at the one or more of the output electrodes **14**B, as measured at step **206**, with a target or desired output current (or voltage) to determine a first error delta. Step **208** can be performed, in sequence, for each of the output electrodes **14**B, and a first error delta can be determined for each of the output electrodes **14**B. The error deltas can be stored in the data store **120** for further processing.
(70) Step **208** follows from the application of the delta rule according to Equation (7) above. If the error delta (e.g., (T.sub.k−I.sub.k) in Equation (7)) is positive, it would be desirable to make the current or currents measured at step **206** greater, thus decreasing the error. This can be accomplished by strengthening the connections to internal nodes which are positive, and weakening the connections to internal nodes which are negative. If the error delta is negative, it would be desirable to make the current or currents measured at step **206** smaller to decrease the error.
(71) At step **210**, the process can include the training module **150** applying a threshold voltage or current to one or more of the output electrodes **14**B for a time period proportional to a magnitude of the first error delta determined at step **208**. For example, a first error delta can be determined at step **208** for a first one of the output electrodes **14**B. Then, at step **210**, the training module **150** can direct the interface module **130** to apply a positive threshold voltage V.sub.T or a negative threshold voltage −V.sub.T to the first one of the output electrodes **14**B for a time period proportional to a magnitude of the first error delta determined for the first one of the output electrodes **14**B at step **208**. The remaining output electrodes **14**B (i.e., other than the first) can remain grounded. Step **210** can iterate over each of the of the output electrodes **14**B, in turn.
(72) The procedure of applying a threshold voltage at step **210** is shown in greater detail in FIG. 5B. At step **302** in FIG. 5B, the process includes the training module **150** determining whether the error delta (e.g., (T.sub.k−I.sub.k) in Equation (7)) from step **208** for the first one of the output electrodes **14**B is negative. If the error delta is negative, then the process proceeds to step **304**. If

the error delta is positive, then the process proceeds to step **308**.
(73) At step **304**, the process includes the training module **150** applying a positive threshold voltage or current to one or more of the output nodes for the time period proportional to the error delta. For example, an error delta can be determined at step **208** for a first one of the output electrodes **14**B. Then, at step **304**, the training module **150** can direct the interface module **130** to apply a positive threshold voltage V.sub.T to the first one of the output electrodes **14**B for a time period proportional to the magnitude of the error delta determined at step **208**. The remaining output electrodes **14**B (i.e., other than the first) can remain grounded at step **304**.
(74) As noted above, if a first one of the output electrodes **14**B is selected and voltage set equal to V.sub.T at step **304**, then the connections to all internal nodes with positive voltages (+V.sub.j) will not change their conductances because their voltage drop (V.sub.T−V.sub.j) will be, by definition, less than V.sub.T since V.sub.j is positive. This isolates the internal nodes with negative V.sub.j, whose voltage drops will be larger than V.sub.T. Also, the voltage drop across each negative internal node is now equal to the relationships in Equations (9) and (10).
(75) Further, the quantity driving the conductance change, ΔG ∝(ΔV−V.sub.T), is therefore proportional to V.sub.j. If the error delta is negative (i.e., the “Yes” branch from step **302** in FIG. **5**B), the current should be more negative. Therefore, it would be desirable to strengthen the connection to the internal nodes with negative voltages. Since, in the above case, ΔV>V.sub.T, the application of V.sub.T at step **304** to the first one of the output electrodes **14**B can increase the conductances to all negative internal nodes by an amount proportional to the V.sub.j of each node. If it is assumed that this voltage is applied for a time t proportional to the error delta on this output node, then the backpropagation algorithm, ΔG.sub.kj=α(T.sub.k−I.sub.k) V.sub.j, can be reconstructed for this set of connections.
(76) This can be extended to the positive internal nodes connected to the first one of the output electrodes **14**B by applying a voltage −V.sub.T after the application of +V.sub.T for the time t. Thus, at step **306**, the process can also include applying a negative threshold voltage or current to one or more of the output nodes for the time period proportional to the error delta. For example, at step **306**, the training module **150** can direct the interface module **130** to apply a negative threshold voltage −V.sub.T to the first one of the output electrodes **14**B for a time period t proportional to the magnitude of the error delta determined at step **208**. The remaining output electrodes **14**B (i.e., other than the first) can remain grounded at step **306**.
(77) Due to step **306**, the voltage drop (−V.sub.T−V.sub.j) across all internal nodes with negative voltages will be under the threshold for conductance change, effectively isolating the internal nodes with positive V.sub.j. For a similar reason, the magnitude of the conductance changes across these positive internal nodes are also proportional to the voltages at each internal node in the direction to decrease, rather than increase, the conductance. Again, this is what is required for an output node with negative error delta (i.e., the “Yes” branch from step **302** in FIG. **5**B). Steps **304** and **306** can occur for all output electrodes **14**B with a negative error deltas.
(78) To account for outputs with positive error deltas (i.e., the “No” branch from step **302** in FIG. **5**B), the directions of these conductance changes can be reversed. Specifically, connections to negative internal nodes can be weakened and connections to positive internal nodes can be strengthened. This can be accomplished by reversing the voltage polarities of all inputs (thus flipping the signs of all internal nodes) at step **308**. At step **308**, the process includes the training module **150** reversing a polarity of the input voltage or current applied to the input electrodes **14**A. This can be a reversal of the polarity of the voltage or current applied to the input electrodes **14**A at step **202**. Further, at steps **310** and **312**, the process includes steps similar to steps **304** and **306**, respectively. Steps **308**, **310**, and **312** can occur for all output electrodes **14**B with a positive error delta. The process can return from FIG. **5**B to step **212** in FIG. **5**A.
(79) Turning back to FIG. **5**A, the training module **150** can now train the conductances or weights between the input electrodes **14**A and the conductive cores **12**A of the internal nodes. Since the

voltages applied to input electrodes **14**A were restricted to between
(80) $-\frac{1}{2}V_T$
and
(81) $\frac{1}{2}V_T$
among steps **202**, **204**, **206**, **208**, and **210**, the and weights between the input electrodes **14**A and the conductive cores **12**A of the internal nodes could not have changed during the training of the output weights, as the maximum voltage drop from input node to internal node cannot exceed V.sub.T.
(82) At step **212**, the process can include the training module **150** transforming, for one or more of the output electrodes **14**B, the first error deltas of the output electrodes **14**B determined at step **208**, into second error deltas for the output electrodes **14**B. As one example, the training module **150** can transform the first error delta current (e.g., (T.sub.k−I.sub.k) in Equation (7)) for the first one the output electrodes **14**B into a second error delta, which may be a voltage. The second error delta can be proportionate to the first error delta, but converted from a current to a voltage, for example, or from a voltage to a current. Step **212** can be performed for each of the output electrodes **14**B with reference to the first error deltas from step **208** which were stored in the data store **120**.
(83) At step **214**, the process can include grounding one or more input nodes of the feedforward network. For example, the training module **150** can direct the interface module **130** to ground one or more of the input electrodes **14**A of the feedforward network **30**. In some cases, the training module **150** can direct the interface module **130** to ground each of the input electrodes **14**A.
(84) At step **216**, the process can include, for each of the output electrodes **14**B, applying the second error delta determined at step **212** to the output electrode **14**B. For example, a second error delta, which can be a voltage, can be determined at step **212** for each of the output electrodes **14**B as described above. The training module **150** can direct the interface module **130** to apply the second error delta voltages for each of the output electrodes **14**B, respectively, to the output electrodes **14**B. Step **216** can also include applying threshold voltages or currents to the input electrodes **14**A, in turn, for a second time period proportional to an absolute value of the second error delta.
(85) The procedure of step **216** is shown in greater detail in FIG. **5**C. At step **320** in FIG. **5**C, the training module **150** can direct the interface module **130** to apply the second error delta voltages determined at step **212** for each of the output electrodes **14**B, respectively, to the output electrodes **14**B.
(86) At step **322** in FIG. **5**C, the process includes the training module **150** determining whether original input applied to one of the input electrodes **14**A at step **202** in FIG. **5**A, was negative. If the original input was negative, then the process proceeds to step **324**. If the original input was positive, then the process proceeds to step **328**.
(87) At step **324**, the process includes the training module **150** applying a positive threshold voltage or current to one or more of the input nodes for the time period proportional to the second error delta. For example, the training module **150** can direct the interface module **130** to apply a positive threshold voltage V.sub.T to one of the input electrodes **14**A for a second time period proportional to an absolute value of the second error delta determined at step **212** in FIG. **5**A. Further, at step **326**, the training module **150** can direct the interface module **130** to apply a negative threshold voltage-V.sub.T to the first one of the input electrodes **14**A for the second time period. The remaining input electrodes **14**A (i.e., other than the first) can remain grounded at steps **324** and **326**.
(88) At step **328** in FIG. **5**C, the process includes the training module **150** reversing a polarity of the second error delta voltages applied to the output electrodes **14**B at step **320**. At step **3330**, the process includes the training module **150** applying a positive threshold voltage or current to one or more of the input nodes for the time period proportional to the second error delta. For example, the training module **150** can direct the interface module **130** to apply a positive threshold voltage

V.sub.T to one of the input electrodes **14**A for a second time period proportional to an absolute value of the second error delta determined at step **212** in FIG. **5**A. Further, at step **332**, the training module **150** can direct the interface module **130** to apply a negative threshold voltage-V.sub.T to the first one of the input electrodes **14**A for the second time period. The remaining input electrodes **14**A (i.e., other than the first) can remain grounded at steps **324** and **326**.
(89) The algorithm presented in FIGS. **5**A-**5**C reproduces the backpropagation algorithm in bipartite memristive networks. Although presented in one example order or sequence in FIGS. **5**A-**5**C, the algorithm can, in some cases, be applied or used in alternative sequences. Additionally, while certain voltages and currents are described as being applied and/or measured in the algorithm, it should be appreciated that the measurement or application of a voltage can be substituted by that of a current. Similarly, the measurement or application of a current can be substituted by that of a voltage. Since the algorithm only iterates over each external node once per training example, it is first order in time with respect to the number of nodes. This is in contrast with the standard implementation of the backpropagation algorithm, which is second order in time.
(90) The components described herein, including the interface module **130**, the measurement module **140**, and the training module **150**, can be embodied in hardware, software, or a combination of hardware and software. If embodied in software, each element can represent a module or group of code that includes program instructions to implement the specified logical function(s). The program instructions can be embodied in the form of, for example, source code that includes human-readable statements written in a programming language or machine code that includes machine instructions recognizable by a suitable execution system, such as a processor in a computer system or other system. If embodied in hardware, each element can represent a circuit or a number of interconnected circuits that implement the specified logical function(s).
(91) The training processor **110** can include at least one processing circuit. Such a processing circuit can include, for example, one or more processors and one or more storage or memory devices coupled to a local interface. The local interface can include, for example, a data bus with an accompanying address/control bus or any other suitable bus structure. The storage or memory devices can store data or components that are executable by the processors of the processing circuit. For example, the training module **150** and/or other components can be stored in one or more storage devices and be executable by one or more processors in the training processor **110**.
(92) The training module **150** and other components described herein can be embodied in the form of hardware, as software components that are executable by hardware, or as a combination of software and hardware. If embodied as hardware, the components described herein can be implemented as a circuit or state machine that employs any suitable hardware technology. The hardware technology can include, for example, one or more microprocessors, discrete logic circuits having logic gates for implementing various logic functions upon an application of one or more data signals, application specific integrated circuits (ASICs) having appropriate logic gates, programmable logic devices (e.g., field-programmable gate array (FPGAs), and complex programmable logic devices (CPLDs)).
(93) Also, one or more or more of the components described herein that include software or program instructions can be embodied in any non-transitory computer-readable medium for use by or in connection with an instruction execution system such as, a processor in a computer system or other system. The computer-readable medium can contain, store, and/or maintain the software or program instructions for use by or in connection with the instruction execution system.
(94) A computer-readable medium can include a physical media, such as, magnetic, optical, semiconductor, and/or other suitable media. Examples of a suitable computer-readable media include, but are not limited to, solid-state drives, magnetic drives, or flash memory. Further, any logic or component described herein can be implemented and structured in a variety of ways. For example, one or more components described can be implemented as modules or components of a single application. Further, one or more components described herein can be executed in one

computing device or by using multiple computing devices.
(95) The above-described examples of the present disclosure are merely possible examples of implementations set forth for a clear understanding of the principles of the disclosure. Many variations and modifications can be made without departing substantially from the spirit and principles of the disclosure. All such modifications and variations are intended to be included herein within the scope of this disclosure and protected by the following claims.

Claims

1. A method to train a memristive network comprising a number of input nodes and a number of output nodes, comprising: applying an input voltage or current to an input node among the number of input nodes; grounding an output node among the number of output nodes; measuring an output current or voltage at the grounded output node; determining an error delta from the measured output current or voltage and a target current or voltage, where the error delta equals the target current or voltage minus the measured output current or voltage; applying a threshold voltage or current to the output node after ungrounding the output node, the threshold voltage or current applied for a time period proportional to a magnitude of the error delta; transforming the error delta into a second error delta; grounding an input node among the number of input nodes; and applying the threshold voltage or current to the input node after ungrounding the grounded input node, the threshold voltage or current applied for a second time period proportional to the second error delta.
2. The method of claim 1, wherein, when the error delta is negative, applying the threshold voltage or current to the output node comprises: applying a positive threshold voltage or current to the output node for the time period proportional to the error delta; and applying a negative threshold voltage or current to the output node for the time period proportional to the error delta.
3. The method of claim 1, wherein, when the error delta is positive, applying the threshold voltage or current to the output node comprises: reversing a polarity of the input voltage or current applied to the input node; applying a positive threshold voltage or current to the output node for the time period proportional to the error delta; and applying a negative threshold voltage or current to the output node for the time period proportional to the error delta.
4. The method of claim 1, wherein: the second error delta comprises an error delta voltage or current; and the method further comprises: applying the error delta voltage or current to the output node with the grounded input node grounded; and applying the threshold voltage or current to the input node for the second time period after ungrounding the grounded input node, the second time period being proportional to an absolute value of the error delta voltage or current.
5. The method of claim 4, wherein, when the input voltage or current applied to the input node was positive, applying the threshold voltage or current to the input node for the second time period after ungrounding the grounded input node comprises: applying a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current; and applying a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current.
6. The method of claim 4, wherein, when the input voltage or current applied to the input node was negative, applying the threshold voltage or current to the input node for the second time period after ungrounding the grounded input node comprises: reversing a polarity of the error delta voltage or current applied to the output node; applying a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current; and applying a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage.
7. The method of claim 1, wherein the memristive network further comprises: a memristive network of memristive nanofibers; and internal electrodes electrically coupled between the number of input nodes and the number of output nodes by memristive shells of the memristive nanofibers.

8. The method of claim 1, wherein the method reproduces a backpropagation algorithm for training the memristive network of permissive nanofibers.
9. A memristive network, comprising: a number of nanofibers, wherein each nanofiber comprises a metallic core and a memristive shell; a number of electrodes deposited upon the nanofibers, wherein the number of electrodes comprise a number of input nodes and a number of output nodes; and a training processor configured to: apply an input voltage or current to an input node among the number of input nodes; ground an output node among the number of output nodes; measure an output current or voltage at the grounded output node; determine an error delta from the measured output current or voltage and a target current or voltage, where the error delta equals the target current or voltage minus the measured output current or voltage; apply a threshold voltage or current to the output node after ungrounding the grounded output node, the threshold voltage or current applied for a time period proportional to a magnitude of the error delta; transform the error delta into a second error delta; ground an input node among the number of input nodes; and apply the threshold voltage or current to the input node after ungrounding the grounded input node, the threshold voltage or current applied for a second time period proportional to the second error delta.
10. The memristive network according to claim 9, wherein, when the error delta is negative, the training processor is further configured to: apply a positive threshold voltage or current to the output node for the time period proportional to the error delta; and apply a negative threshold voltage or current to the output node for the time period proportional to the error delta.
11. The memristive network according to claim 9, wherein, when the error delta is positive, the training processor is further configured to: reverse a polarity of the input voltage or current applied to the input node; apply a positive threshold voltage or current to the output node for the time period proportional to the error delta; and apply a negative threshold voltage or current to the output node for the time period proportional to the error delta.
12. The memristive network according to claim 9, wherein: the second error delta comprises an error delta voltage or current; and the training processor is further configured to: apply the error delta voltage or current to the output node with the grounded input node grounded; and apply the threshold voltage or current to the input node for the second time period after ungrounding the grounded input node, the second time period being proportional to an absolute value of the error delta voltage or current.
13. The memristive network according to claim 12, wherein, when the input voltage or current applied to the input node was positive, the training processor is further configured to: apply a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current; and apply a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current.
14. The memristive network according to claim 12, wherein, when the input voltage or current applied to the input node was negative, the training processor is further configured to: reverse a polarity of the error delta voltage or current applied to the output node; apply a positive threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage or current; and apply a negative threshold voltage or current to the input node for the second time period proportional to the absolute value of the error delta voltage.
15. The memristive network according to claim 9, wherein the memristive network further comprises: a memristive network of memristive nanofibers; and internal electrodes electrically coupled between the number of input nodes and the number of output nodes by memristive shells of the memristive nanofibers.
16. A method to train a memristive network comprising a number of input nodes and a number of output nodes, comprising: applying an input voltage to an input node among the number of input nodes; grounding an output node among the number of output nodes; measuring an output current at the grounded output node; determining an error delta from the measured output current and a

target current to determine an error delta, where the error delta equals the target current or voltage minus the measured output current or voltage; applying a threshold voltage to the output node after ungrounding the grounded output node, the threshold voltage or current applied for a time period proportional to a magnitude of the error delta; transforming the error delta into a second error delta; grounding an input node among the number of input nodes; and applying the threshold voltage to the input node after ungrounding the grounded input node, the threshold voltage or current applied for a second time period proportional to the second error delta.
17. The method of claim 16, wherein, when the error delta is negative, applying the threshold voltage to the output node after ungrounding the grounded output node comprises: applying a positive threshold voltage to the output node for the time period proportional to the error delta; and applying a negative threshold voltage to the output node for the time period proportional to the error delta.
18. The method of claim 16, wherein, when the error delta is positive, applying the threshold voltage to the output node after ungrounding the grounded output node comprises: reversing a polarity of the input voltage applied to the input node; applying a positive threshold voltage to the output node for the time period proportional to the error delta; and applying a negative threshold voltage to the output node for the time period proportional to the error delta.
19. The method of claim 16, wherein the memristive network further comprises: a memristive network of memristive nanofibers; and internal electrodes electrically coupled between the number of input nodes and the number of output nodes by memristive shells of the memristive nanofibers.
20. The method of claim 19, wherein the method reproduces a backpropagation algorithm for training the memristive network of memristive nanofibers.