SANDIA REPORT
SAND201X-XXXX
Unlimited Release
Printed Month and Year

Hardware Acceleration of Adaptive Neural Algorithms

Conrad D. James, Kevin R. Dixon, James B. Aimone, Ojas D. Parekh, Nadine E. Miner, Aleksandra Faust, Stephen J. Verzi, Frances S. Chance, Tu-Thach Quach, Christopher Lamb, Meghan Galiardi, Samuel A. Mulder, William M. Severa, Kristofor Carlson, Michael R. Smith, Cynthia A. Phillips, Jacob A. Hobbs, Robert G. Abbott; Jeffrey A. Piersol, Emily Donahue, John H. Naegle, Alexander W. Hsia, Sapan Agarwal, Craig M. Vineyard, Fredrick Rothganger, Jonathon W. Donaldson, Gabriel Popoola, Aaron J. Hill, Matthew J. Marinella, Thomas E. Beechem, Ronald S. Goeke, Albert A. Talin, Paul G. Kotula, Farid El Gabaly, Elliot J. Fuller, James Stevens, David R. Hughart, Andrew Armstrong, Michael D. Henry, Gad S. Haase, Steven L. Wolfley, Seth Decker, Christopher B. Saltonstall, Jamison Wagner, John Niroula, Rudeger H. Wilke, Michael Van Heukelom, Patrick S. Finnegan, Carl Smith, Robin B. Jacobs-Gedrim, Steven J. Plimpton, Justin E. Doak, Jean-Luc Watson, Richard Schiek, Brian D. Tierney, Robert Bondi, Harold P. Hjalmarson, Timothy J. Draelos, Jonathan Cox, Joe B. Ingram, Jason W. Wheeler, David R. Follett, Duncan Townsend, Pamela L. Follett, Isaac Richter, Engin Ipek, Felix Wang

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods/

SAND2017-12810
Printed November 2017
Unlimited Release

Hardware Acceleration of Adaptive Neural Algorithms

Author Name(s)
Department Name(s)
Sandia National Laboratories
P. O. Box 5800
Albuquerque, New Mexico 87185-MS1080

Abstract

As traditional numerical computing has faced challenges, researchers have turned towards alternative computing approaches to reduce power-per-computation metrics and improve algorithm performance. Here, we describe an approach towards non-conventional computing that strengthens the connection between machine learning and neuroscience concepts. The Hardware Acceleration of Adaptive Neural Algorithms (HAANA) project has developed neural machine learning algorithms and hardware for applications in image processing and cybersecurity. While machine learning methods are effective at extracting relevant features from many types of data, the effectiveness of these algorithms degrades when subjected to real-world conditions. Our team has generated novel neural-inspired approaches to improve the resiliency and adaptability of machine learning algorithms. In addition, we have also designed and fabricated hardware architectures and microelectronic devices specifically tuned towards the training and inference operations of neural-inspired algorithms. Finally, our multi-scale simulation framework allows us to assess the impact of microelectronic device properties on algorithm performance.

ACKNOWLEDGMENTS

We thank the Laboratory Directed Research and Development Office for supporting this project, and we thank all of the Sandia National Laboratories management, staff, technologists, and students, as well as our collaborators for supporting and contributing to this project.

TABLE OF CONTENTS

1. Project purpose – Hardware Acceleration of Adaptive Neural Algorithms (HAANA) 11
2. Relevant Background 13
   - 2.1. The history of neural-inspiration in data-driven computing 13
   - 2.2. Neuromorphic hardware accelerators 13
   - 2.3. Novel microelectronic devices for neuromorphic hardware weights 14
   - 2.4. Current work in machine learning for cyber and imaging applications 15
3. Summary of Accomplishments 17
   - 3.1. Theoretical assessment of neural-inspired algorithms 17
   - 3.2. Neurogenic Deep Learning 18
   - 3.3. Adaptive algorithms 19
   - 3.4. Deep learning for cyber data processing 20
   - 3.5. Neuromorphic Spiking Algorithms and Architectures 21
     - 3.5.1. Spiking algorithms 21
     - 3.5.2. Spiking Neural Architecture - Neuromorphic Data Microscope . 22
     - 3.5.3. Spiking Neural Architecture - Spiking Temporal Processing Unit (STPU) 23
   - 3.6. Microelectronic devices for implementing artificial neural network weights 25
   - 3.7. Resistive memory device technology 26
     - 3.7.1. Modeling and experimentation with ReRAM films 27
     - 3.7.2. Theoretical analysis of accelerating computation with neuromorphic crossbar arrays 27
     - 3.7.3. Resistive memory device fabrication and characterization 28
     - 3.7.4. Optimized resistive memory crossbar arrays for neuromorphic computing 28
4. Conclusion 31
5. Presentations 33
6. Intellectual Property 37
7. Recognitions 39
8. References (project publications in bold with joint collaborations noted) 41

FIGURES

Figure 1: (left) Plot of the vertical feature and horizontal feature scores of “1” (green) and “7” (yellow) handwritten digit data used to train a DNN. (right) Neurogenic deep learning uses new processing nodes to function as new feature detectors. 18

Figure 2: Conceptual schematic of the non-von Neumann Neuromorphic Data Microscope (NDM) architecture, with complex memory layers (mapping/efficacy and temporal/spatial) and simple integrators. 22

Figure 3: Speech detection using a liquid state machine (LSM) network. Spoken digits are processed into Mel-Frequency Cepstral Coefficients (MFCCs) and then input into an LSM. The weights between the liquid and the readout neurons (black) are then trained with a classifier designed to accurately categorize the speech data. 23
Figure 4: Raster plot of LSM neuron firings in the STPU hardware as a function of training examples for spoken digits “0” and “1”. 24
Figure 5: (left) Three-layer artificial neural network with two sets of weights w_{ij} and w_{kl}. (right) Implementation of the first two layers of the network with weights represented by variable resistors with conductances G_{ij}. 25
Figure 6: (left) Schematic of ReRAM tantalum oxide devices in the “off” and “on” states. (right) Image of a SNL-fabricated tantalum oxide ReRAM device array (10x10) with ~ 2 μm x 2 μm active regions. 26
Figure 7: The accuracy of classifying MNIST digits using a ReRAM crossbar array with A/D and D/A conversions. 28
Figure 8: Diagram of the conductance G of a hardware synapse device as a function of input write voltage pulses. Examples of an ideal device (blue) and a non-ideal device (grey) are shown, in addition to the write noise (red). 29
Figure 9: Plot of the change in conductance (ΔG) as a function of initial conductance measured from a tantalum oxide device. 30

TABLES

Table 1: Difference in confusion matrices for file fragment classification using sparse dictionary learning & LSTMs compared to the SVM method described in (45). 20

EXECUTIVE SUMMARY

As traditional numerical computing has faced challenges due to the end of Dennard scaling and increasing power budgets for new generation supercomputers, researchers have turned towards alternative computing approaches to reduce power-per-computation metrics and to develop more efficient algorithms. Neural-inspired computing has matured into a field that has produced promising methodologies for solving difficult problems including pattern recognition and anomaly detection. Yet the challenges with neural-inspired computing are well-known: the need for large amounts of training data, the relative brittleness of algorithms to data variability, and the ad-hoc nature of optimization techniques. We have taken an approach to address these concerns by strengthening the connection between machine learning (ML) and neuroscience concepts – termed neural machine learning (NML) - in order to improve the resiliency and adaptability of such algorithms.

The Hardware Acceleration of Adaptive Neural Algorithms (HAANA) project at Sandia National Laboratories has developed NML algorithms and hardware for applications in image processing and cybersecurity. While ML methods can extract relevant features from many types of data, the effectiveness of these algorithms can diminish over time in real-world environments. Our team has generated several approaches that support continual adaptation of algorithms to changing data and that incorporate unsupervised learning to reduce the reliance on subject-matter experts to manually craft features of interest.

In addition to developing algorithms, we have also designed several hardware architectures to enhance learning and performance in a suite of spiking (i.e. time-encoded) and non-spiking neural-inspired algorithms. We have designed a non-von Neumann architecture to implement complex temporal dynamics in spiking algorithms such as liquid state machines (LSMs), and this led to an improvement in classification accuracy in speech data. To accelerate training in algorithms such as Deep Learning (DL), we used a resistive memory crossbar architecture to parallelize prevalent computational kernels such as vector-matrix multiplication. In order to quantitatively evaluate the impact of architecture design on algorithm performance, we constructed a multiscale simulation framework that couples circuit-level architecture characteristics with microelectronic device properties and algorithm tuning parameters. Also, included within our research is the fabrication of novel microelectronic devices that are specifically tuned for neural-inspired computing applications.

NOMENCLATURE

Abbreviation	Definition
Abbreviation	Definition
HAANA	Hardware Acceleration of Adaptive Neural Algorithms
DL	Deep Learning
LSM	Liquid State Machine
CNN	Convolutional Neural Network
LISTA	lithium-ion synaptic transistor for analog computation
PCRE	Perl-Compatible Regular Expressions
ReRAM	Resistive random access memory, resistive memory
ML	Machine learning
DNN	Deep neural network
ANN	Artificial neural network
GPU	Graphical processing unit
CPU	Central processing unit
DG	Dentate gyrus
EC	Entorhinal cortex
SVM	Support vector machine
LSTM	Long short-term memory
FPGA	Field programmable gate array
LIF	Leaky integrate and fire
STPU	Spiking temporal processing unit
NDM	Neuromorphic data microscope
MLP	Multi-layer perceptron
ASIC	Application specific integrated circuit
MFCC	Mel-frequency cepstral coefficients
LDA	Linear discriminant analysis
CMOS	Complementary metal oxide semiconductor
MNIST	Modified National Institute of Standards and Technology

1. PROJECT PURPOSE – HARDWARE ACCELERATION OF ADAPTIVE NEURAL ALGORITHMS (HAANA)

Modern threats to national security are increasingly difficult to detect. They emerge rapidly, and once active they engage in evasive behavior to avoid detection. These types of time dynamics are very difficult to detect given that humans cannot handle large volumes of data and automated techniques require time-consuming training. These factors make the current state-of-the-art in detection system technology slow to adapt to new threats and threats that evolve over time. Data-driven computing methods that train models – in contrast to numerical computing methods that solve explicit equations with defined boundaries and initial conditions – have demonstrated enormous success at challenging pattern recognition tasks. However, ML and other data-driven computing methods suffer from several disadvantages such as brittleness, expensive training costs, and erratic false positives and false negatives. Part of the challenge with ML algorithms is that although the approach of using training examples to build predictive models is inspired by neurobiological systems, ML has not incorporated modern developments in neuroscience that have led to improved understanding of how organisms process information and learn. We contend that ML algorithms need to become more neural-inspired in order to mature these technologies to solve challenging problems.

Here, we describe the Hardware Acceleration of Adaptive Neural Algorithms (HAANA) project. The purpose of this project was to identify deficits in current algorithmic approaches for addressing cyber and image processing problems. Neural-inspired DL and recurrent network algorithms are capable of unsupervised feature extraction (no prior labeling of threats by an expert) from training data. We used these algorithms to detect signatures of interest in the cyber and image processing problem domains. We also developed neural-inspired hardware architectures designed to accelerate data processing, algorithm training, and algorithm inference. Modeling and simulation was used to evaluate the performance of algorithms and projected hardware architectures, and we benchmarked the performance using objective metrics including classification accuracy, power consumption, and time-to-detect.

2. RELEVANT BACKGROUND

With the growth in mobile device usage, the volume of cyber data and image data has grown exponentially. This naturally produces a challenge to analytics aimed at searching through such data to identify items of interest, and the reliance on humans to conduct such analysis is problematic as the number of human analysts will not scale to the volume of data. Autonomous techniques for analyzing data help mitigate this problem, however subject matter experts are still required at some level. New computational algorithms and hardware are clearly needed to address this concern.

2.1. The history of neural-inspiration in data-driven computing

Advanced computing machines have long been a topic of research and development. The earliest tools were developed for simple counting procedures, and over time, computing machines became more complex and capable of producing more sophisticated calculations. The mechanisms of neurobiological computation have also been of interest, given the substantive ability for biological organisms to learn and adapt to changing conditions. **As detailed in James et al. (1), some of the earliest research in neuroscience and psychology in the first quarter of the 20th century led to the development of what we refer to as "data-driven computing."** Instead of relying on solving explicit equations and processing lines of computational instructions, these approaches use large amounts of data to "train" algorithms to develop generalized models of data. These methods include artificial neural networks (ANNs, such as McCulloch-Pitt neurons), statistical machine learning (such as decision trees), and dynamical machine learning (such as Markov models). These methods, to one extent or another, take a non-conventional approach towards computing that is much more neural-inspired than traditional computing. As neuroscience and psychology matured as scientific fields over the years, the influence of these disciplines on data-driven computing fluctuated. Neuroscience in the 50s and 60s influenced the development of data-driven computing; for example, the work of Hubel and Wiesel (2) inspired the development of the Cognitron(3), a neural network algorithm that leveraged the concept of receptive fields and varying neuron characteristics for processing training data. However, it is our contention that the influence of neuroscience on data-driven computing over the last century has led to many of the improvements in performance of these algorithms. For instance, the development of deep neural network (DNN) algorithms (4) since the mid-2000s demonstrates the significant benefits of transitioning algorithms towards more neural-inspired concepts – in this case, implementing the multi-layered hierarchical structure of neurobiological circuits into perceptron-based ANNs. Machine learning has now been applied to a large number of applications including particle physics (5), medical imaging (6), computational biology (7), remote sensing (8), cyber-attack analysis (9), and data-file categorization (10).

2.2. Neuromorphic hardware accelerators

In addition to advances in neural-inspired algorithms, the improvement in microelectronic hardware over the last three decades, from faster processors to new architectures such as Graphical Processing Units (GPUs), has also contributed to the growth of data-driven computing. Also, worth noting, advances in microelectronics have

led to advances in neuroscience via improved computational neuroscience models and data analytics tools (11). As an example, GPUs, though not designed specifically for accelerating machine learning algorithms, turned out to be important tools for training large neural networks. This naturally led to researchers developing new hardware accelerators that mimic the structure and/or function of neurobiological systems. Many researchers have leveraged existing complementary metal oxide semiconductor (CMOS) transistor technology to build application specific integrated circuit (ASIC) chips and to reconfigure field programmable gate array (FPGA) boards into neural-inspired hardware systems for a variety of computing applications. Platforms such as IBM's TrueNorth ASIC chip have been used to implement low-power neural-inspired pattern recognition (12). Neurogrid (13), from Stanford University, and SpiNNaker (14), from the University of Manchester, are hybrid analog-digital systems for simulating neurobiological and artificial neural networks. BrainScales is an FPGA-based neuromorphic system from Heidelberg University and partners, and this platform has been used for training and running spiking neural networks (15).

While many of these platforms have been designed and fabricated as generalized neuromorphic computing systems, researchers are also developing hardware for specific algorithms and applications. Chen et. al designed DianNao, an ASIC for accelerating the operations in convolutional neural networks (CNN) and DNNs (16). Google Inc. developed a Tensor Processing Unit™ chip for reducing the energy consumption of running CNNs, Multi-Layer Perceptrons (MLPs), and Long Short-Term Memory (LSTM) algorithms in their data-centers (17). Microsoft Inc. recently demonstrated Project Brainwave, an FPGA-based neural network accelerator (18). The large number of hardware accelerators being built has led to the development of new software tools for interfacing to such systems. In addition to Google's Tensorflow (19), other researchers have developed software simulation environments such as the Dynamic Adaptive Neural Network Arrays (DANNA) framework from Oak Ridge National Laboratories and partners, and the Neurons-to-Algorithms (N2A) software system (20) from Sandia National Laboratories. These software platforms streamline the difficult process of mapping ML and neural-inspired algorithms onto different hardware architectures.

2.3. Novel microelectronic devices for neuromorphic hardware weights

Researchers have also looked beyond existing transistor technology to fabricate microelectronic devices that are specifically designed for accelerating the training and/or inference of machine learning and neural network algorithms. As an example, this can be accomplished by using variable resistors to serve as the algorithm weights. The benefit of such an approach is that the resistors can be modified iteratively throughout the training process, and then set to a particular resistance for the inference procedure. There are many requirements for such a system: the resistors must be tunable in either direction with sufficient precision, they must remain stable without a supply of energy at a set state for long periods (non-volatile), and the energy required to set the resistor to a given state must be low. The first neuromorphic hardware systems for training and running ANN's relied on controlled potentiometers to adjust and store neural network weights (21-23). Over the decades as the microelectronics industry matured, hardware synapses were fabricated using CMOS transistors as demonstrated in Mead et. al (24). Concerns about power consumption and scalability have recently led to researchers investigating new

device technologies to instantiate ANN synapses in hardware (25). A variety of technologies have been investigated including phase change memory (26, 27), spin torque transfer memory (28), ferroelectric memory (29), and resistive memory (ReRAM) & memristors (30-33). Given the current state of neuromorphic computing technology, a hybrid platform of neural cores and conventional microelectronic support circuits is the likely format of such accelerators for the near term.

2.4. Current work in machine learning for cyber and imaging applications

There are many research efforts into using machine learning and neural-inspired algorithms for cybersecurity and imaging applications. Overall, the community recognizes that existing algorithms require large amounts of training data and are limited in terms of their ability to adapt to changing training data. These are characteristics that neurobiological networks are proficient at handling. In imaging, recent work from Lake et. al indicates that researchers are examining learning methods that are more in-tune with how biological organisms learn (34). In this work, the objective was to use probabilistic methods to achieve “one-shot learning”, where an algorithm is trained on a small number of example data-points to form a generalized model of the data. This type of algorithm is more amenable to handling issues such as concept drift, where data, features, and other components of a problem change over the course of time.

Cybersecurity has similar challenges in dealing with drift and rapid changes in data. A recent survey article describes machine learning techniques being applied for intrusion detection (35). A variety of techniques were discussed such as SVMs, Markov models, and Bayesian inference, and a number of applications were also mentioned including misuse detection, malware tracking, and internet traffic classification. A point stressed in this article is that new methods for “fast incremental learning” are needed to keep models up-to-date for real-time detection of threat activities, and they also discuss the challenge of handling the large volume of cyber-data that exists. Taken together, a major theme in both imaging and cyber applications is that there is a strong demand for algorithms that are capable of handling streaming data, classifying data in real-time, and updating models dynamically. Elements of these challenges served as mission-drivers for the HAANA project and will be discussed throughout the rest of the report.

3. SUMMARY OF ACCOMPLISHMENTS

The HAANA project has produced a significant body of published work that will be cited throughout this report. The algorithms thrust was tasked with developing new techniques for addressing data mission drivers in cybersecurity and image processing. This led to two separate approaches in the development of spiking (input data is encoded in digital signals during processing within the algorithm) and non-spiking algorithms. These separate tracks resulted from the different types of data being processed: non-spiking DL was used for static image data and a spiking liquid state machine was used for temporally-varying speech data. This also led the architecture team to develop a spiking architecture – the Spiking Temporal Processing Unit (STPU) – and a non-spiking architecture – the resistive memory crossbar. The learning hardware team was tasked with developing new memory technologies for storing and updating the weights of ML and ANN algorithms. As part of this effort and due to the difficulty in meeting all of the optimization requirements, we developed and modeled several devices with different switching characteristics. Filament-based devices that switch from one resistance state to another via the growth of a conductive filament between two electrodes were developed, and non-filamentary devices that change conductivity due to the redistribution of conductive carriers under an electric field were also developed. Filamentary devices are small in footprint and have faster but more catastrophic switching, which makes their use in training algorithms difficult, given that many small updates are applied to the algorithm weights during training. Nonfilamentary devices are well-suited for weights given that they can be easily switched into a large number of resistance states, however, their switching is slower and their areal footprint is larger.

3.1. Theoretical assessment of neural-inspired algorithms

One of the concerns of ANNs and ML is the black-box nature of these algorithms and the difficulty in understanding how decision boundaries form in such systems. Deciphering how these algorithms work is an important step towards mainstream use of such approaches as this will address concerns about performance robustness. A related concern is the lack of theory surrounding the construction, operation, and optimization of neural algorithms. In fact, our understanding of how biological neural networks operate, and specifically how they encode information is also somewhat unknown. **To partly address this concern, we developed an approach to measure complexity in action potential signaling in order to estimate information content in biological neural networks (36).** The reasoning behind this methodology was to use theories on information encoding in neurobiological networks to assess information content in ANNs and other machine learning algorithms. Thus, we constructed mathematical models of the transmission and encoding of information in biological neural networks as opposed to implementing biochemical models of these networks. In this study, we used our methodology to assess the impact of new neurons being generated in the adult hippocampus region of the brain. This neurogenesis process has been hypothesized to play a central role in increasing the information content in biological organisms (37, 38), and **our computational model of the dentate gyrus (DG) portion of the hippocampus has been used by colleagues to guide experimental neuroscience research (39).** In Vineyard et. al (36), we determined that a combination of mature neurons and newly generated neurons would provide the optimum information encoding system for an

organism: mature neurons provide stability to networks by virtue of being tightly tuned to encode limited and fixed amounts of information, and immature neurons provide adaptability to networks by being broadly tuned and capable of encoding new information in an online learning-type process.

As another component of our interest in the hippocampus as a source of inspiration for algorithm development, we augmented our model of this region with a quantitative metric to measure information processing. **In Severa et. al (40), we used a normalized dot product to measure the pattern separation function of the DG.** This region has been hypothesized to decorrelate similar data, and the normalized dot product is a formal reference metric for the function of the mapping between the entorhinal cortex (EC) and the DG. We also measured the impact of DG neurogenesis on the model, and we determined that biologically-plausible mixed-weight sampling improved pattern separability in the EC to DG pathway.

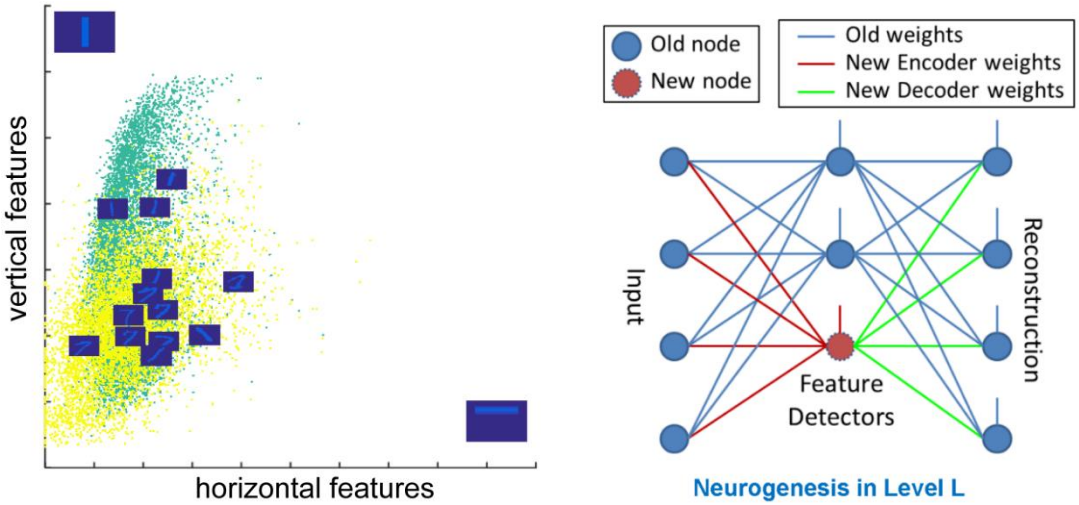

Figure 1: (left) Plot of the vertical feature and horizontal feature scores of “1” (green) and “7” (yellow) handwritten digit data used to train a DNN. (right) Neurogenic deep learning uses new processing nodes to function as new feature detectors.

3.2. Neurogenic Deep Learning

A benefit to developing mathematical models of neural systems is that we can directly take our models and apply them to ANN and ML algorithms. A major limitation of data-driven computing algorithms is that they are limited by the amount and nature of training data. This is especially concerning for unsupervised learning algorithms such as DNNs which require tremendous amounts of time and energy to train. Figure 1(left) shows an example of a DNN trained only on handwritten “1s” and “7s” from the Modified National Institute of Standards and Technology (MNIST) handwritten digit database. We tracked the features detected in each hand-written digit so that we could measure the distribution of features being detected in different training data. In this experiment, we tracked the scores for vertical and horizontal features as a test, given that handwritten “1s” (green) will be dominated by vertical scores, while 7s (yellow) should have scores for both features. Tracking the feature scores is one method to evaluate the decision boundaries

in DNNs when trained on these types of datasets, and the goal is that such analyses will improve our ability to tune algorithms for optimum performance.

We developed Neurogenic Deep Learning specifically as a modified version of a DNN in which new processing nodes are added throughout the network to improve the algorithms ability to be trained on new data while not damaging representations of previously trained data (41). In this article, we designed a deep autoencoder, a structure that contains a set of encoder layers to transform training data and then a set of decoder layers to reconstruct the data. In control experiments with a static autoencoder structure, the network was trained sequentially on digits (i.e. “1s”, “7s”, “0s”, etc.) from the MNIST database and then queried with all ten digits. For the control experiments, the autoencoder is only capable of reconstructing the most recent digit that has been trained. Figure 1(right) shows an example of how neurogenic deep learning is implemented in an autoencoder. New processing nodes are inserted into a particular layer, and new encoder and decoder weights are inserted and trained. Layers are chosen for new processing nodes based on the reconstruction error of training data: when a particular error threshold is met, new nodes are inserted in a layer until the error is reduced to a chosen amount. When trained sequentially, the neurogenic autoencoder was able to reconstruct the most recently trained digit while maintaining the ability to reconstruct previously trained digits.

3.3. Adaptive algorithms

A strong driver of the HAANA research mission is to develop new approaches towards ML, specifically in regard to implementing functionality found in neurobiological systems. One particular area of interest is in developing techniques for adaptive learning in order to address situations where the data is changing over the course of time (virtual concept drift), or where the rules of the situation are changing over the course of time (real concept drift). **One of our project’s first publications was to examine the use of a support vector machine (SVM) to categorize data (42).** In this publication, we coupled a SVM (an algorithm traditionally used to optimize the separation of data into classes with maximum differentiation) with MapReduce, a programming model designed to parallelize algorithms. Here, we use this methodology to break up data into batches and process batches separately to identify multiple support vectors. The support vectors from each batch are then “played” against each other to yield a final support vector for the entire data set. This methodology is helpful for extremely large datasets where parallel processing can be used to reduce data processing time. **This work was expanded to the case where a similar SVM Game was employed to handle dynamic data (43).** In this scenario, the data to be categorized is not available all at once but is accumulated over time. Multiple iterations of the SVM Game can be played to continuously adapt the categorization of the data as a function of time. This approach is useful for maintaining accurate and up-to-date processing of data in real-world scenarios such as mobile deployed systems. **Recently, the HAANA team used this game theory methodology to create a generalized theory around adaptive learning (44).** The primary motivation for this work was to highlight the challenges of having training and inference as separated processes, and to promote the use of “neural adaptive learning” as a method to overcome those challenges by having training and performance seamlessly integrated.

3.4. Deep learning for cyber data processing

The SVM work utilized example datasets such as the automobile mileage data set described in Vineyard et al. 2015 (43). Using these types of data sets is necessary in order to demonstrate the utility of candidate algorithms and associated machine learning technologies. In the first year of the project, we also decided to work with more relevant cyber and imaging data sets. **In Cox et al., we used a file-type dataset consisting of 4500 files of 9 different types (e.g. pdf, html, etc.) (46).** The purpose of this work was to develop an algorithm that would predict the identification of a file-type based on inherent characteristics. The header information was removed from the files, and then three features were used to categorize the files into the 9 groups: entropy, power spectral density, and byte distribution. We used a DNN for the categorization consisting of one input layer, two hidden layers, and an output softmax layer. We found that the best predictive categorization results occurred when we concatenated all three features together and sent that data through the network. We were able to achieve 100% prediction accuracy on several file-types such as html and gif, while other file-types were more difficult to categorize (e.g. png files).

The next step in this effort was to move towards more real-world cyber scenarios where conditions are not ideal. For this work, we used fragments of files and then used an unsupervised machine learning technique called sparse coding, or sparse dictionary learning (47). Sparse dictionary learning uses regular transformation techniques to create a basis set of vectors to reconstruct data, while imposing a sparsity constraint to have fewer non-zero coefficients for the basis vectors. We first formatted the file fragment data into 1D vectors, and generated a byte dictionary for all the different file-type fragments. We hypothesize that the sparse dictionary was useful for generating local features in the

Table 1: Difference in confusion matrices for file fragment classification using sparse dictionary learning & LSTMs compared to the SVM method described in (45).

Actual class \ Predicted class	csv	doc	gif	gz	html	jpg	pdf	png	ppt	ps	rtf	swf	txt	xls	xml
csv	-9.8	-0.9			-3.8		-0.3			-0.3	-1.1		-0.9		-2.5
doc	-0.5	11.9	-0.5	6.6	-3.0	-1.0	-0.4	4.1	-1.5	-0.5	-3.6	-1.0	14.0	0.6	-1.4
gif			36.5	29.3		-10.0		17.3				-0.1			
gz			-12.0	-28.1		-31.8	-0.4	16.6				-0.5			
html	-1.4	-9.0			-17.5		-0.4		-0.6	-1.6	-10.4		-3.4	-0.1	9.4
jpg		0.4	-9.6	18.7		15.2	-1.7	6.4	-0.4			0.2	1.2	-0.4	0.4
pdf		-0.1	-7.2	15.8	-0.6	-17.9	9.4	12.6	-1.2	5.7	-0.5	-0.3	3.0	-0.1	0.2
png			-13.4	9.1		-21.0	-0.4	-26.7	-0.2			-0.8			
ppt		3.2	-5.7	15.0	-0.1	-14.3	-0.1	11.2	9.1	0.2	-0.1	1.0	0.7	-2.1	0.2
ps	-1.4		-0.8	0.2	-2.3	-1.1	-0.3	-0.3		-3.2	-1.1		5.7	-1.8	
rtf	-0.5	-6.2	0.6		-6.1	0.4	-0.7			3.0	-4.2		6.3		-1.0
swf		0.7	-5.3	19.8		-23.6	-1.0	10.2	-0.9	-0.1		2.0		1.8	0.4
txt	-4.3	-4.2			-8.2		0.2			9.3	-4.8		-15.4	-0.6	-2.8
xls		11.7	-0.3	0.2	-2.2	-0.2	-0.4	0.8	-0.8	-0.3	-0.2	-0.8	-0.9	6.3	-0.3
xml	-0.6	-2.0			16.1		1.2			-0.6	-4.2		6.0		15.9

fragments, so we turned to another algorithm – Long Short-Term Memory (LSTM) networks – to generate global features within the data. LSTMs are multi-terminal neural networks that can be stacked in order to generate “long-distance” features by mitigating the vanishing gradient problem (48). The combination of sparse dictionary learning and LSTM networks produced file recognition accuracies that rivaled the best methods described in the literature. However, those techniques used supervised learning with SVM classifiers (45), whereas our methodology has the added benefit of being unsupervised and thus not requiring subject matter experts to manually craft features. Table 1 shows our methodology’s preliminary results for 13 file-types compared to the Fitzgerald et. al results.

3.5. Neuromorphic Spiking Algorithms and Architectures

At this point we have described several algorithms that have been focused on processing static data. We have also examined algorithms that require training with example data to build generalizable models to make predictions about newly acquired data. Our team has also examined the use of spiking – the digital encoding of data into the time domain – as a method to process data. At the core of spiking algorithms is the leaky-integrate-and-fire (LIF) neuron, a simplified computational model of biological neurons that sum a series of inputs and sends an output if a threshold is reached within a certain integration time (49). One of the arguments behind this methodology is that the energy consumption in the spiking domain can be reduced: in essence, a LIF neuron consumes energy only when a digital “1” is transmitted/processed while no energy is consumed otherwise. In addition, the temporal dynamics of transforming data into the time domain can lead to improved data separation and thus improved algorithm performance. Using temporal encoding as a method for data transformation is a strategy that can not only be beneficial for data-driven computing, but for traditional numerical computing as well.

3.5.1. Spiking algorithms

The objective of numerical computing is to explicitly compute a known function using input data, and **in Rothganger et. al (50), our team detailed an approach to use dynamical spiking neural networks to compute linear algebra operations such as matrix decomposition.** Studies such as this demonstrate that many of the neuromorphic computing hardware platforms that are traditionally designed for low-power spike-based data-driven computing algorithms, can be formatted for use in scientific computing applications. **Further extending this notion, in Severa et al. (51) we describe an analysis of spiking algorithms for scientific computing.** The methodology described here can be used for applications such as particle image velocimetry where the velocity vector field of a set of moving objects are computed as a function of time. In this example, there are $O(n^2)$ neurons (where n is a dimension determined by the size of the input data) so that the computation can be made in constant time. This article also details a method by which we use time-coded multiplexing to reduce the number of neurons to $O(n)$ while increasing the time required for the computation to $O(n)$. This flexibility allows for spiking algorithms to be implemented in different formats depending on which resource (neurons or time) is constrained in a particular application. **Recently, we described a set of impacts that neural computing could have in the realm of scientific computing (52).**

For instance, we provide evidence that a simplified version of a LIF neuron – termed a threshold gate – can be used for direct calculations such as matrix multiplication with improved scaling over Boolean logic circuits.

Another demonstration of the advantage of using spiking algorithms is in the realm of numerical optimization as detailed in Verzi et. al 2017 (53). In this manuscript, we used a spiking neuron network to compute several optimization functions including the maximum, minimum, and median of a set of integers. We then demonstrated the use of a spiking median filter for image processing. The advantage of using a spiking neural network to compute these functions is that they can potentially be implemented in low-power fast-computation spiking neuromorphic architectures such as TrueNorth, SpiNNaker, and Neurogrid. In this manuscript, we also provide a complexity analysis of the spiking algorithms to generalize their computational advantages under different conditions (time constrained vs. neuron-constrained).

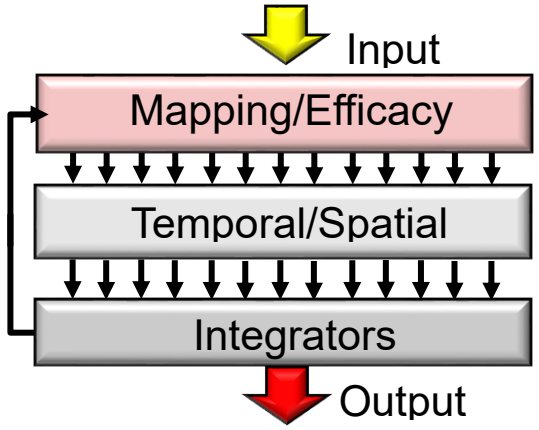

Figure 2: Conceptual schematic of the non-von Neumann Neuromorphic Data Microscope (NDM) architecture, with complex memory layers (mapping/efficacy and temporal/spatial) and simple integrators.

3.5.2. Spiking Neural Architecture - Neuromorphic Data Microscope

Spiking algorithms present a set of advantages and challenges for their use in different applications. As mentioned, they can be configured to consume less power if the data to be processed can be adequately encoded in time for processing with a spiking architecture. We have examined two applications for spiking architectures. The first is in detecting patterns in textual data using Perl Compatible Regular Expressions (PCRE). PCREs are sequences of textual characters (alphanumeric and special characters) that can be used as search patterns in textual data (http://www.pcre.org/). **In collaboration with our partners at Lewis Rhodes Laboratories, we developed a neural-inspired spiking architecture – the Neuromorphic Data Microscope (NDM) - for rapidly detecting PCREs in streaming network data (54).** Specifically, this hardware architecture design is non-von Neumann in that the traditional computer structure of having memory, a control unit, and arithmetic logic in separately connected structure is not followed. Instead, our architecture is inspired by neurobiological systems and uses multiple layers of complex memory structures to determine the strengths and locations of temporal and spatial connections between data "integrators" (Figure 2). This architecture is essentially a feed-forward artificial neural network with integrators as neurons and the connections between neurons are stored in the layers of memory structures. This architecture was instantiated in an FPGA format (Intel Arria 10), tested on a benchmarked data stream (https://pen-testing.sans.org/holiday-challenge/2013), and demonstrated an ~1000x improvement in operations/W for PCRE detection.

3.5.3. Spiking Neural Architecture - Spiking Temporal Processing Unit (STPU)

The strength of the Neuromorphic Data Microscope architecture is the reversed complexity (compared to CPUs) of having simple neuron integrators and complex synaptic connections. The parameters of the synaptic connections – weights, network connectivity, and decay characteristics of post-synaptic connections - are maintained in a look-up table. This provides a high degree of flexibility to hardware-coded neural networks, and specifically regarding how information is encoded by the dynamics of spiking within the network of neurons. Another strength of the Neuromorphic Data Microscope is that it can be generalized to other problems outside of detecting patterns

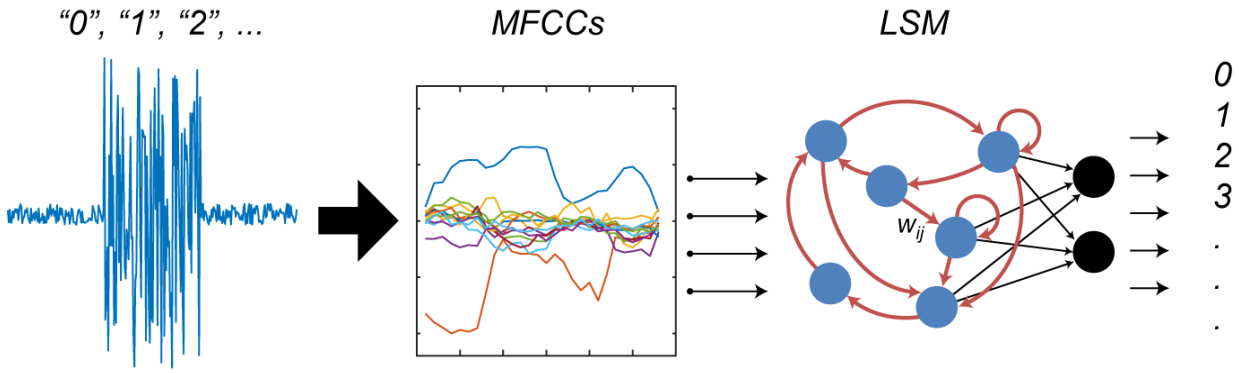

Figure 3: Speech detection using a liquid state machine (LSM) network. Spoken digits are processed into Mel-Frequency Cepstral Coefficients (MFCCs) and then input into an LSM. The weights between the liquid and the readout neurons (black) are then trained with a classifier designed to accurately categorize the speech data.

in streams of network data. The compatibility with highly dynamic data allows this architecture to be formatted to handle video data, speech data, and other types of temporal data. An algorithm that has been frequently used to process dynamic temporal data is a liquid state machine (LSM). Developed by Wolfgang Maass et. al, an LSM is a data transformation tool that can be used to convert temporal data into a spatio-temporal representation of higher dimensionality (55). LSMs rely on LIF neurons which perform temporal integration, and the liquid consists of a set of random and recurrent connections between LIF neurons. The theory behind the liquid is that the temporal dynamics of data input into the liquid will be transformed by the spiking dynamics of the neurons, and that the transformed data will be simpler to separate and classify. Generally, there is no training or learning within the liquid. A separate set of readout neurons receive the processed data from the liquid and the weights between the liquid neurons and the readout neurons are trained using a standard algorithm such as a SVM or linear regression.

Like DL algorithms, LSMs have the computational bottleneck of completing large numbers of vector-matrix multiply operations in addition to handling the complex synaptic response functions of spiking neurons. Efficiently performing vector-matrix multiples and instantiating spiking are difficult to accomplish with conventional CMOS-based CPUs. This is largely why specialized hardware platforms such as TrueNorth and SpiNNaker have been developed. However, working with specialized hardware has challenges of its own, specifically regarding data processing and software interfaces. Thus, we designed

a generalized architecture based on the NDM for accelerating spiking algorithms and vector-matrix multiplies – termed the Spiking Temporal Processing Unit. **Smith et. al describes this architecture design and simulates the classification of speech data (56).** In this demonstration, spoken digits were processed into Mel-frequency cepstral coefficients (MFCCs) which are essentially short-term power spectra of the sounds. The MFCCs were then input into an LSM and the average firing rate was used to train linear classifiers to categorize each sound as a particular spoken digit (Figure 3). In this paper, we used several linear classifiers and found that linear discriminant analysis (LDA) provided the most accurate digit recognition as compared to an SVM and several regression methods. The liquid in these simulations had varying numbers of neurons (<1000) and different structures of synaptic connections. A major tuning parameter for the liquid was the synaptic response function, which is the structure of the spike signal that exits a spiking neuron. The synaptic response function can be made into an arbitrary function such as a rapidly decaying signal or a sustained signal, with different functions leading to different dynamics within the liquid and thus impacting algorithm performance. An advantage of the STPU architecture is that it is capable of efficiently instantiating such functions discretely in its complex memory structure. **After we evaluated the LSM-LDA performance on classifying spoken digits with the STPU simulator, we implemented the STPU in an FPGA board (57).** As with the NDM, we programmed the STPU onto an Intel Arria 10 GX 1150 FPGA for the hardware, and we used the N2A software stack (20) to run algorithms on the platform. We conducted a series of power measurements on the STPU to assess metrics such as Joules-per-event, dynamic power, and seconds-per-leaky-integrate-and-fire-operation, and these numbers were reported in comparison to other neuromorphic hardware platforms. A final experiment was completed implementing the LSM in the STPU's FPGA hardware. Figure 4 shows the raster plot of the final neuron firing state from the LSM. In this case, we programmed 148 neurons (135 liquid neurons and 13 input neurons) into the FPGA and there were 660 training examples with 330 each for spoken digits "0" and "1". As suggested by theory, there is indeed a clear difference in the final spike output between the two classes of spoken digits. Figure 4 provides qualitative visual evidence that the LSM has transformed the data into a higher dimensional space where separation and classification are more readily apparent, with spoken "1s" in general producing more neuron firings in the final output for a given training example. For this hardware demonstration, we used linear regression (off the FPGA) to perform the spoken digit classification.

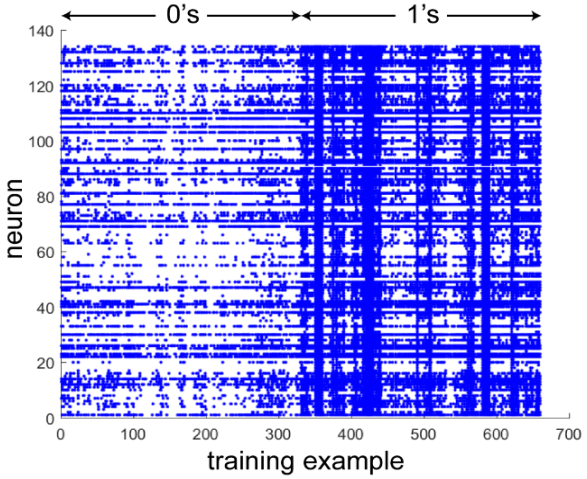

Figure 4: Raster plot of LSM neuron firings in the STPU hardware as a function of training examples for spoken digits "0" and "1".

3.6. Microelectronic devices for implementing artificial neural network weights

A major challenge for machine learning and ANN algorithms is mapping them onto microelectronic hardware to accelerate elements of the computations required for training and inference. To reap the computational benefits of such systems, hybrid platforms composed of neural "core" circuits that are trained to instantiate neural algorithms and conventional digital circuits for other functionality (e.g. communication and analog-to-digital conversion) will likely be required. However, modern microelectronics are limited in terms of power and scaling, two characteristics that are especially important for the expensive training required by large machine learning algorithms. For instance, our neuromorphic architectures that have been built with the latest FPGA technology are

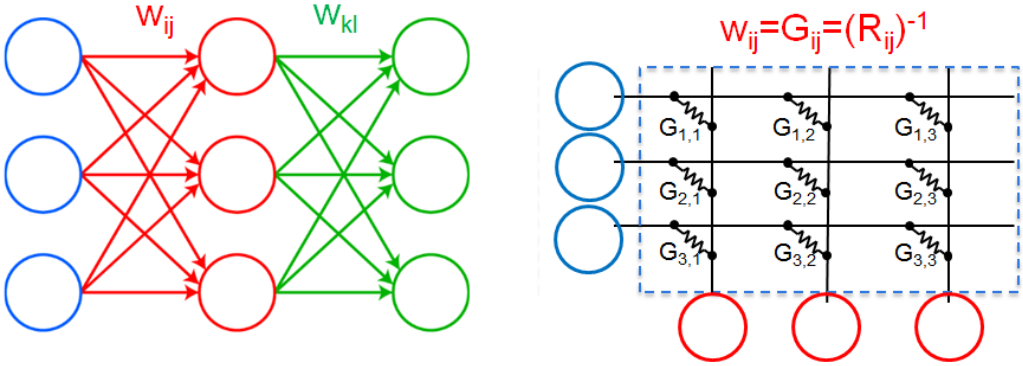

Figure 5: (left) Three-layer artificial neural network with two sets of weights w_{ij} and w_{kl}. (right) Implementation of the first two layers of the network with weights represented by variable resistors with conductances G_{ij}.

limited in terms of the amount of SRAM memory that can be kept on chip – typically on the order of tens of MB. With lower power microelectronic devices that can be scaled below 5nm, we could replace megabytes of SRAM memory with terabytes of memory, and thus improve the speed and power consumption of these and other neuromorphic platforms.

For many years, researchers have sought to develop microelectronic devices for storing and updating the weights in neural network algorithms, including conventional CMOS-based technology such as EEPROMs (58) and emerging technology such as metal oxide resistive memory devices (31). Figure 5 shows an example three-layer ANN. Earlier, we discussed how these weights could be implemented in hardware to speed up the training of such weights to their optimal inference performance values. Figure 5 also indicates how the weights between the first two layers would be implemented in hardware with variable resistors, where the inverse of the resistance value would serve as the weight. The primary motivation behind the development of such an approach is that the training procedure of machine learning and ANN algorithms is time-consuming and expensive. For instance, in Le et al. (59), the Google team constructed a DNN with over 10 layers and on the order of 10^9 weights that required 3 days to train. It is hypothesized that the increased depth of networks facilitates generalization in data processing by allowing features to be combined hierarchically to represent and/or reconstruct the training data. However, deeper networks consume more time and energy to train. Thus, many researchers have focused on developing microelectronic devices that are low-power to reduce the energy consumption during training. Technologies that can

potentially be fabricated at dimensions below 5nm have also been of interest in order to reduce the areal footprint of large networks instantiated in hardware.

We recently published a review of resistive memory technologies including bipolar and unipolar metal oxide cells, filamentary and nonfilamentary devices, phase change devices, and ion-conducting devices (60). The technologies have different advantages and disadvantages regarding issues such as power consumption, switching speed, linearity, asymmetry, and fabrication scaling. For the HAANA project, we focused on filamentary and non-filamentary ReRAM devices due to their advantages in low-power switching and potential scalability below 5nm. **We also published an analysis of the use of resistive memory devices in crossbar architectures for training neural algorithms (61).** In this study, we used experimental data collected from fabricated devices to simulate a crossbar architecture running the backpropagation algorithm used to train weights in DNNs. As part of this work, we developed a new bench procedure for sampling the physical characteristics of devices that allowed rapid coverage of the device operating space without closed-loop control of initial resistances. The simulations based on this data demonstrated that the lack of precision in open-loop setting of network weights severely limits backpropagation's ability to converge. We offered an alternate algorithm to backpropagation which takes advantage of noisy open-loop writes to perform a random walk to find more appropriate weight values within a network. **In Merkel et. al, we partnered with colleagues to simulate a hybrid neuromorphic architecture system complete with neural cores based on ReRAM devices and conventional microelectronic support circuitry (62).** We used an image recognition dataset for testing, and demonstrated that the ReRAM device technology provided a substantial reduction in energy consumption as compared to conventional transistor-based circuits.

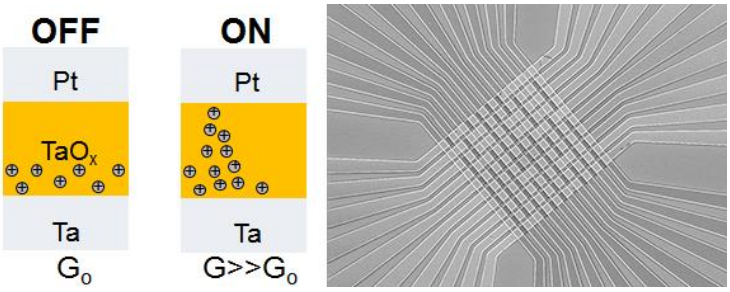

Figure 6: (left) Schematic of ReRAM tantalum oxide devices in the “off” and “on” states. (right) Image of a SNL-fabricated tantalum oxide ReRAM device array (10x10) with ~ 2 μm x 2 μm active regions.

3.7. Resistive memory device technology

To use a new device technology in platforms for accelerating neural-inspired algorithms, a basic understanding of the physics of the switching mechanism must be obtained. The HAANA project team has extensive experience in developing switching technologies including multiferroic materials such as bismuth manganite (63) and tantalum oxides (64). Figure 6 shows a diagram that describes the switching mechanism in tantalum oxide devices. Initially after fabrication, the device is in a high resistance (low conductivity) state G_0. In this state, positively charged oxygen vacancies are distributed randomly within the switching layer. After the application of a voltage pulse, the oxygen vacancies redistribute under the electric field to form a conductive filament that increases the conductivity through the film, and the

device is now in a high conductance state. In practice, we have used different electrode materials and switching layers, but the basic structure remains the same.

3.7.1. Modeling and experimentation with ReRAM films

The most important characteristic of the learning hardware under consideration for the HAANA project was the energy required for the device to switch, or change from one resistance state to another state. **One of our first publications was a study of the switching mechanism in tantalum oxide devices (65).** In this work, we used time-domain thermoreflectance to examine how the stoichiometry of the switching material impacts the thermal and electrical conductivities, and the different contributions to the total device conductivity of those two components. The conductivity of the ReRAM device ultimately impacts the operating conditions of a neural-inspired algorithm training/inference hardware accelerator in terms of the total current consumed as well as the speed of state-switching. **This work was followed-up with a Density Functional Theory (DFT) model of the different phases of the tantalum oxide switching layer in our resistive memory devices (66).** This work analyzed the impact of the tantalum oxide stoichiometry and phase on the electronic structure and electrical conductivity of the film, **and more recently, we measured the impact of different material dopants on the conductivity of tantalum oxide (67).** The use of dopants such as carbon and nitrogen to modulate the switching film conductivity would help devices meet the requirements of having large resistances to reduce the current consumption in crossbar architectures.

3.7.2. Theoretical analysis of accelerating computation with neuromorphic crossbar arrays

In Section 3.6, we describe several of our publications on neuromorphic system modeling to ascertain their impact on algorithm performance. But these models are largely phenomenological, and are thus difficult to extend beyond the specific neuromorphic architecture designs under consideration. So as part of the HAANA project, we made a concerted effort to develop mathematical theory around the advantages and disadvantages of neural-inspired algorithms. **For instance, we performed a complexity analysis regarding analog vector-matrix multiplication (68, 69).** Granted, analog computing is not necessarily neural-inspired computing, but analog multiplication is a core operation of ANNs and thus we consider analog computation to be within the more expansive realm of neural-inspired computing. In these articles, we compared the energy costs of computing a matrix multiplication operation using analog resistive memory crossbars – a potential next generation memory storage technology – to the energy costs when using conventional digital SRAM technology. We found that an O(n) savings occurs due to the fact that in SRAM, each row/column needs to be addressed separately, whereas with analog resistive memory the multiplication and addition operations are conducted in parallel with no requirement for separate addressing of columns/rows. In (68), we examined the impact of the energy savings on the sparse coding algorithm, using one resistive memory array to store the sparse coding dictionary and a second array to store updates to the dictionary. These analyses served as a strong motivation for research into leveraging neural-inspired functions to improve algorithm performance, not

just in energy consumption, but in other aspects of performance such as robustness and accuracy.

3.7.3. Resistive memory device fabrication and characterization

Navigating the complex operational space of the hardware synapses to accelerate neural algorithms presents a significant challenge. Developing a device that is linear and symmetric when pulsed with a current/voltage, while also remaining non-volatile during storage and requiring a low-power to switch resistance states is difficult to achieve. We demonstrated several resistive memory devices as candidate technologies. **In Agarwal et. al 2016 (70), we describe a series of simulations using switching data from tantalum oxide ReRAM devices. We examined the impact of read noise, write noise, symmetry, and linearity on the classification accuracy of handwritten digits and file-types.** In addition to our expertise in tantalum oxide ReRAM devices, **we also designed and fabricated a lithium-ion synaptic transistor for analog computation (LISTA) device (71).** This three-terminal device is similar to a lithium ion solid-state battery, except it is operated as an analog switch. Applying a voltage on the gate causes lithium to intercalate within the cobalt oxide cathode, resulting in a change in conductivity across the device. The device is capable of being set into a large number of stable resistance states, however scaling the size of the device down to reduce its areal footprint is difficult. For this manuscript, we also examined the impact of LISTA device characteristics on the classification accuracy of handwritten digits and file-types. **A related device technology was built in collaboration with Stanford University using polymer materials (72).** Due to the materials that were used, the fabrication cost of this device is drastically reduced, and the technology can be built upon flexible substrates for wearable electronics applications.

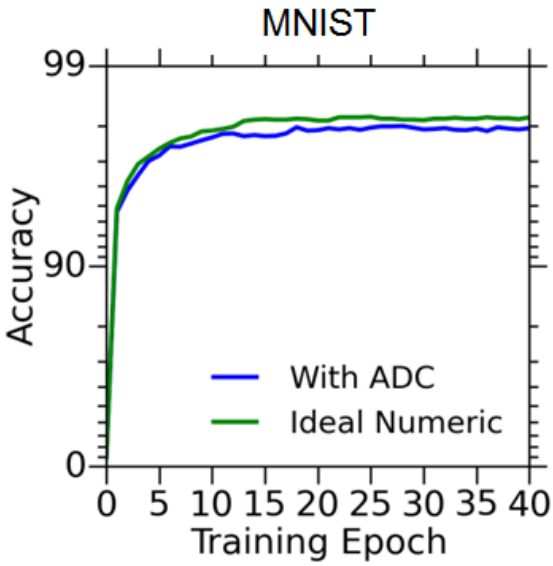

Figure 7: The accuracy of classifying MNIST digits using a ReRAM crossbar array with A/D and D/A conversions.

3.7.4. Optimized resistive memory crossbar arrays for neuromorphic computing

After designing resistive memory device technologies, we needed to assess their use in arrays for training and running ML and NML algorithms. Our previous work detailed in Section 3.7.2 was largely theoretical and lacked the detail required for an accurate energy loss and performance metric analysis. **In Agarwal et. al 2017 (73), we used our crossbar architecture simulation package CrossSim (https://cross-sim.sandia.gov) to examine the impact of different memory technologies on the training and inference of neural algorithms.** In this paper, we assessed the performance of our tantalum oxide and LISTA ReRAM devices. Figure 7 shows a simulation of training a DNN

to classify MNIST hand-written digits. For the simulation, we included the operation of D/A conversion for inputs going into the crossbar, as well as the A/D conversion for outputs leaving the crossbar. We found there was only a small decrement in algorithm performance compared to ideal numeric accuracy due to the operation of the D/A and A/D converters. We also demonstrated a novel method for improving the algorithm performance by using multiple ReRAM devices per weight to increase the resolution of the changes in the weights during training. In this methodology, the algorithm accuracy improved (as compared to using a single device for a weight) from 89% to 97% for the MNIST dataset, and from 87% to 93% for the file-type identification dataset. **Another optimization we performed for ReRAM crossbar arrays was to mitigate parasitic resistances within the array by placing additional resistors within the array to ensure that each device is subjected to the same voltage (74).** This is a serious concern with crossbar arrays of moderate size (1000x1000) as devices in the first row or column are subjected to a different voltage than devices in the last row or column.

Previously, we described the important characteristics for ReRAM devices to be successfully used in algorithm accelerators. The challenge in using resistive memory devices as neural network weights is the difficulty in setting the devices to specific resistance values without a time- and energy-consuming closed-loop control system. To set devices to specific resistances, the devices need to be linear, meaning that a given current/voltage pulse will change the resistance of the device by the same amount regardless of the number of pulses that have been applied to the device (Figure 8). The ideal device also needs to be symmetric, meaning that the resistance change in a device

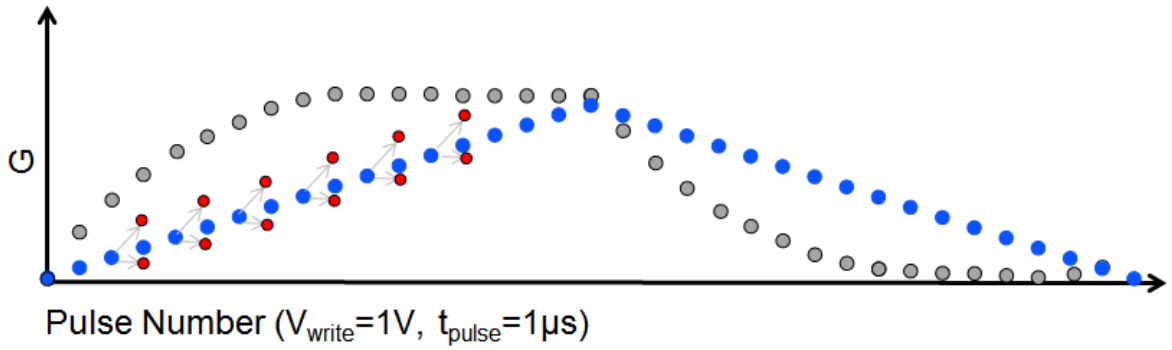

Figure 8: Diagram of the conductance G of a hardware synapse device as a function of input write voltage pulses. Examples of an ideal device (blue) and a non-ideal device (grey) are shown, in addition to the write noise (red).

produced by a positive current/voltage pulse can be reversed with an opposite polarity pulse. The example non-ideal device shown in Figure 8 exhibits strong asymmetry due to the different G curves before and after the midway point of the pulse number axis. In addition, when an input pulse is applied to a device, there is a noise associated with the final resistance state of the device (red in Figure 8). This uncertainty can produce a smaller or larger G in the final state of the device after the write pulse. Another concern is the effect of a voltage pulse on a device in a given resistance state. Figure 9 shows a

plot of the change in conductance (ΔG) of a tantalum oxide device as a function of the initial conductance state (G_0). As described by Burr et. al, this type of plot is helpful in assessing the use of hardware switching devices for training algorithms (75). Ideally, a device would have a flat distribution to ensure that the ΔG of a device would be the same regardless of G_0. Our experimental data from this particular device shows a peak at low values of G_0 which indicates a large distribution of final conductance states when voltage pulses are applied to devices with low initial conductance (~ 16-18mS). At higher G_0, the ΔG is more uniform which is beneficial for training accuracy. Developing devices with optimal switching characteristics has been a primary focus of the learning hardware device team, and we have used this switching analysis on our device technologies. **In Jacobs-Gedrim et. al, we describe a series of experiments with ReRAM devices comprised of three different material systems to examine the impact of linearity and write noise on algorithm performance (76).** We explored different write voltage pulse widths (10ns, 100ns, and 1000ns) and found that the 1000ns pulses produced the most linear conductance changes but also the widest variation in the final device conductance state. CrossSim was used to evaluate the impact of the different device materials on MNIST classification accuracy. Specifically, the nonlinearity of the devices had the strongest negative impact on algorithm accuracy, more so than the write noise. These results have led us to focus on addressing the nonlinearity of our algorithm weight hardware devices in future work.

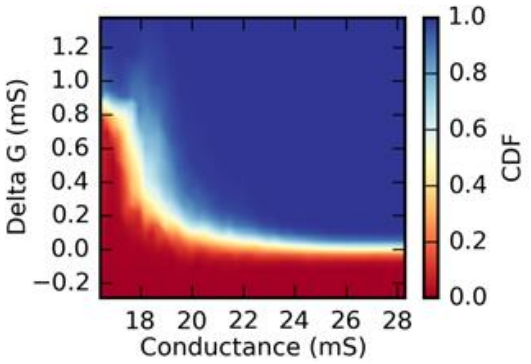

Figure 9: Plot of the change in conductance (ΔG) as a function of initial conductance measured from a tantalum oxide device.

4. CONCLUSION

In conclusion, the HAANA Grand Challenge project has developed and matured a suite of neuromorphic and neural-inspired algorithms and architectures for pattern-matching problems in imaging and cybersecurity applications. HAANA focused on application domains where the world is dynamic, and we developed machine learning algorithms that can adapt continuously over time to changes in application demands. We also developed an infrastructure by which we can assess the impact of next-generation neuromorphic architectures and microelectronic devices for use in hardware accelerators. Essential to the success of this approach was generating a theoretical analysis of neural-inspired computing, and to identify scenarios under which such approaches provide performance advantages. For instance, we identified the energy advantage of performing linear algebra operations in analog with crossbar architectures. We have also developed algorithms for determining the type of a data-file using both supervised and unsupervised techniques. Our methodology was used to identify situations in which translating a problem into the time domain with spiking algorithms provides a time and/or energy advantage. An analysis of spiking algorithms and neurobiological systems also led us to design and build a non-von Neumann architecture in an FPGA platform that improved the speed of pattern matching in network data. We have also designed and fabricated new resistive memory microelectronic devices to reduce the energy consumption and foot-print of future neuromorphic architectures. This included the assembly of a simulation and modeling framework for evaluating the impact of device technologies on circuit operation and algorithm performance. Future work includes the continued maturation of these technologies for developing robust solutions to computational challenges facing the nation.

5. PRESENTATIONS

1 Can memristors learn? 2015 February NICE Albuquerque, NM USA; F. Rothganger talk

2 Adaptive neural algorithms: The What, Why, and How; 2015 February NICE conference; Albuquerque, NM USA; B. Aimone talk

3 Resistive memory for neuromorphic algorithm acceleration 2015 February NICE conference; Albuquerque, NM USA M. Marinella talk

4 Characterizing switching variability in TaOx memristors; 2015 April MRS Conference San Francisco, CA USA D. Hughart talk

5 Transport in TaOx films 2015 June EMC Columbus, OH USA; P. Mickel; talk

6 An assessment of switching variability in TaOx and LiNb based multi-state resistive memories 2015 June EMC Columbus, OH USA; D. Hughart talk

7 Stoichiometric control of thermal transport in tantalum oxide films for memristive applications 2015 June EMC Columbus, OH USA;T. Beechem talk

8 Neural circuit models on emulated hardware 2015 October SFN Conference; Chicago, IL USA; F. Rothganger poster

9 Quantifying neural information content: a case study of the impact of hippocampal adult neurogenesis through computational modeling 2015 October SFN Conference; Chicago, IL USA; C. Vineyard poster

10 Dimensionality reduction of cortical spiking networks - quantifying the structure behind the chaos 2015 October SFN Conference; Chicago, IL USA; B. Aimone poster

11 Temporal processing Unit 2015 October MILCOM Tampa, FL USA; D. Follett demonstration

12 Temporal processing Unit 2015 November Supercomputing Austin, TX USA; D. Follett demonstration

13 A Sparse Coding Model of the Hippocampal Dentate Gyrus 2016 Jan Joint Mathematics Meeting Seattle, WA; W. Severa

14 Hardware Acceleration of Adaptive Neural Algorithms (HAANA) 2016 March NICE Berkeley, CA USA; C. James talk

15 Modulating Neural Computation 2016 March NICE Conference; Berkeley, CA USA; B. Aimone talk

16 A combinatorial model of dentate gyrus sparse coding and pattern separation 2016 March NICE Conference; Berkeley, CA USA; W. Severa talk

17 Temporal processing Unit 2016 March NICE Berkeley, CA USA; D. Follett demo

18 Thermal transport of tantalum oxide films for memristors 2016 March MRS Conference; Phoenix, AZ USA; T. Beechem talk

19 Neurogenic deep learning 2016 May ICLR Conference; San Juan, Puerto Rico; T. Draelos poster

20 Li-ion synaptic transistor for analogue computation (LISTA) 2016 June ECS Conference; San Diego, CA USA;E. Fuller talk

21 Emerging Technologies for the Acceleration of Neuromorphic Algorithms 2016 June; "The Ninth Workshop on Fault-Tolerant Spaceborne Computing Employing New Technologies" Albuquerque, NM USA; M. Marinella talk

22 Neural Computing: What Scale and Complexity is Needed? 2016 July ORNL Neuromorphic Computing Workshop Knoxville, TN USA; J. Aimone talk

23 Device to System Modeling Framework to Enable a 10 fJ per Instruction Neuromorphic Computer 2016 July ORNL Neuromorphic Computing Workshop; Knoxville, TN USA; M. Marinella talk

24 Neuromorphic data microscope 2016 August; Information Assurance Symposium Washington DC; D. Follett and J. Naegle talk

25 Revisiting the canonical model of the hippocampus 2016 September Neuroscience Seminar Series - University of New Mexico Albuquerque, NM USA; J. Aimone talk

26 Tantalum Oxide Resistive Memory Devices By Ion Assisted Deposition 2016 October; ECS Conference; Honolulu, HI USA; R. Goeke talk

27 Spiking Network Algorithms for Scientific Computing 2016 October IEEE International Conference for Rebooting Computing San Diego, CA USA; W. Severa talk

28 Computing with dynamical systems 2016 October IEEE International Conference for Rebooting Computing San Diego, CA USA; F. Rothganger talk

29 Neural machine learning algorithms and hardware for image analysis and data science applications2016 October Brain Informatics and Health Omaha, NE; C. James; talk

30 Implementation of a liquid state machine with temporal dynamics on a novel spiking neuromorphic architecture 2016 October Brain Informatics and Health Omaha, NE; M. Smith talk

31 Formalizing Function within the Hippocampal Trisynaptic Circuit 2016 November; Society for Neuroscience San Diego, CA USA; W. Severa poster

32 Can we be formal in assessing the strengths and weaknesses of neural architectures? A case study using a spiking cross-correlation algorithm 2016 December NIPS Computing with Spikes Workshop Barcelona, Spain; W. Severa

33 Optimization-based computation with spiking neurons 2016 December NIPS Computing with Spikes Workshop Barcelona, Spain; S. Verzi talk, poster

34 Ultra-Efficient Neural Algorithm Accelerator Using a ReRAM Crossbar Accelerator; 2017 February; Energy Consequences of Information; Santa Fe, NM; M. Marinella talk

35 Non-volatile redox transistors for low-power analog computing 2017 February; Energy Consequences of Information; Santa Fe, NM; A. Talin talk

36 An Efficient Implementation of a Liquid State Machine on the Spiking Temporal Processing Unit 2017 March NICE Conference; San Jose, CA; M. Smith talk

37 Staying on the path 2017 March NICE Conference; San Jose, CA; F. Rothganger talk

38 Studying Adaptive Learning through Game-Theoretic Modeling 2017 March NICE Conference; San Jose, CA; Vineyard talk

39 Ultra-Efficient Neural Algorithm Accelerator Using Processing With Memory 2017 March NICE Conference; San Jose, CA; Marinella Talk

40 Constant-Depth Neuromorphic Circuits for Matrix Multiplication 2017 March NICE Conference; San Jose, CA; O. Parekh poster

41 Hippocampus-inspired Adaptive Neural Algorithms 2017 March NICE Conference; San Jose, CA; J. Aimone Talk

42 DOE National Laboratories & BRAIN: Neural Computing at Sandia 2016 December BRAIN PI Annual Meeting Bethesda, MD; J. Aimone Poster

43 Neural-inspired computing algorithms and hardware for image analysis and cybersecurity applications 2017 April Salishan Conference on High-Speed Computing Gleneden Beach, OR; C. James invited talk

44 Efficient Memory Acquisition via Sparse Sampling 2017 March NonVolatile Memory Workshop UCSD, CA; T. Quach poster presentation

45 "Energy Efficient Neuromorphic Algorithm Acceleration Enabled by Resistive Memory (ReRAM) Crossbars" 2017 February MRQW - Microelectronics Reliability and Qualification Working Meeting Los Angeles, CA; M. Marinella keynote talk

46 Effects of RRAM Electroforming and Switching Methods on Device Performance Elucidated with Ultrafast Current Measurements 2017 June Electronic Materials Conference South Bend, IN; R. Jacobs-Gedrim talk

47 "Li-Ion Synaptic Transistor for Low Power Analogue Computing (LISTA)" 2017; April MRS 2017 Phoenix, AZ USA; E. Fuller talk

48 Non-volatile redox transistors for low-power analog computing and brain-machine interfaces 2017 May ECS 2017 New Orleans, USA; E. Fuller talk

49 In Operando Characterization of Pt-TaOx-Ta Bipolar Vacancy Change Memories 2017; ECS Conference; National Harbor, MD; R. Jacobs-Gedrim accepted

50 The Challenges of Neural-inspired computing: lessons learned from Sandia's Grand Challenge 2017 July NITRD High End Computing Interagency Working Group; Arlington VA; J. Aimone invited talk

51 Neural-inspired computing algorithms and hardware for image analysis and cybersecurity applications 2017 August New Research Ideas Forum – SNL Albuquerque, NM; C. James; talk

52 Studying Adaptive Learning through Game-Theoretic Modeling 2017 August; Machine Learning & Deep Learning Conference Albuquerque, NM; C. Vineyard talk

53 Deep Learning at Sandia: Challenges and Opportunities 2017 August Machine Learning & Deep Learning Conference Albuquerque, NM; T. Draelos talk

54 Deep Hyperspectral Classification for Data-driven Sparse Sampling 2017 August Machine Learning & Deep Learning Conference Albuquerque, NM; W. Severa talk

55 An Evaluation of Sequence Learning Methods for Detecting Malware Using System Call Traces 2017 August Machine Learning & Deep Learning Conference Albuquerque, NM; J. Ingram talk

56 Severe Sigmoid Deep Spiking Neural Networks 2017 August Machine Learning & Deep Learning Conference Albuquerque, NM; S. Verzi talk

57 Neurogenesis Deep Learning 2017 August Machine Learning & Deep Learning Conference Albuquerque, NM; J. Aimone talk

6. INTELLECTUAL PROPERTY

1. CrossSim v. 1.0; https://cross-sim.sandia.gov
2. GloVe C++ v. 1.0; https://github.com/joncox123/GloveCpp
3. Cortexsys v. 3.0; https://github.com/joncox123/Cortexsys

7. RECOGNITIONS

1. National Academy of Engineering's (NAE) 23rd annual U.S. Frontiers of Engineering (USFOE) symposium - James B. Aimone as a participant; https://www.nae.edu/173089.aspx

2. IEEE spectrum issue June 2017 - Fredrick K. Rothganger as a contributor; http://spectrum.ieee.org/computing/software/in-the-future-machines-will-borrow-our-brains-best-tricks

3. Sandia National Laboratories Labnews article on neuromorphic devices - 5/12/2017; http://www/news/publications/labnews/articles/2017/12-05/brain.html

4. Sandia National Laboratories Labnews article on neural computing - 10/3/2016; https://share-ng.sandia.gov/news/resources/news_releases/neural_computing/

5. Popular press article based on a Sandia National Laboratories Labnews story on the Neuromorphic Cyber Microscope - 3/24/2017; http://www.darkreading.com/attacks-breaches/sandia-testing-new-intrusion-detection-tool-that-mimics-human-brain/d/d-id/1328478; https://share-ng.sandia.gov/news/resources/news_releases/bad_apples/#.WNQy3xIrKYW

6. Neuromorphic Cyber Microscope - 2016 R&D100 Finalist: IT/Electrical, Software/Services, and Special Recognition: Market Disruptor; https://www.rd100conference.com/awards/winners-finalists/year/2016/

7. R&D Article - Brain-Inspired Computing Pushes the Boundaries of Technology; 6/2/2017; James B. Aimone; https://www.rdmag.com/article/2017/06/brain-inspired-computing-pushes-boundaries-technology

8. REFERENCES (PROJECT PUBLICATIONS IN BOLD WITH JOINT COLLABORATIONS NOTED)

1. **James CD*, et al.* (2017) A historical survey of algorithms and hardware architectures for neural-inspired and neuromorphic computing applications. *Biologically Inspired Cognitive Architectures*.**
2. Hubel DH & Wiesel TN (1959) Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology* 148(3):574.
3. Fukushima K (1975) Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics* 20(3-4):121-136.
4. Hinton GE, Osindero S, & Teh Y-W (2006) A fast learning algorithm for deep belief nets. *Neural Computation* 18(7):1527-1554.
5. Baldi P, Sadowski P, & Whiteson D (2014) Searching for exotic particles in high-energy physics with deep learning. *Nat Commun* 5:4308.
6. Plis SM*, et al.* (2014) Deep learning for neuroimaging: a validation study. *Frontiers in Neuroscience* 8.
7. Angermueller C, Pärnamaa T, Parts L, & Stegle O (2016) Deep learning for computational biology. *Molecular Systems Biology* 12(7):878.
8. Chen S, Wang H, Xu F, & Jin Y-Q (2016) Target classification using the deep convolutional networks for SAR images. *IEEE Transactions on Geoscience and Remote Sensing* 54(8):4806-4817.
9. Aggarwal P*, et al.* (2015) Cyber Security: A game-theoretic analysis of defender and attacker strategies in defacing-website games. *Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), 2015 International Conference on*, (IEEE), pp 1-8.
10. Beebe NL, Maddox LA, Liu L, & Sun M (2013) Sceadan: Using concatenated n-gram vectors for improved file and data type classification. *IEEE Transactions on Information Forensics and Security* 8(9):1519-1530.
11. **Bouchard KE*, et al.* (2016) High-Performance Computing in Neuroscience for Data-Driven Discovery, Integration, and Dissemination. *Neuron* 92(3):628-631 (collaboration).**
12. Merolla PA*, et al.* (2014) A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* 345(6197):668-673.
13. Benjamin BV*, et al.* (2014) Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE* 102(5):699-716.
14. Furber SB, Galluppi F, Temple S, & Plana LA (2014) The SpiNNaker Project.
15. Schmitt S*, et al.* (2017) Neuromorphic hardware in the loop: Training a deep spiking network on the brainscales wafer-scale system. *Neural Networks (IJCNN), 2017 International Joint Conference on*, (IEEE), pp 2227-2234.
16. Chen T*, et al.* (2014) Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning. *ACM Sigplan Notices*, (ACM), pp 269-284.
17. Jouppi NP*, et al.* (2017) In-datacenter performance analysis of a tensor processing unit. *arXiv preprint arXiv:1704.04760*.
18. Burger D (2017) Microsoft unveils Project Brainwave for real-time AI. (Microsoft Inc.).
19. Abadi M*, et al.* (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

20. Rothganger F, Warrender CE, Trumbo D, & Aimone JB (2014) N2A: a computational tool for modeling from neurons to algorithms. *Front Neural Circuits* 8:1.
21. Widrow B (1960) *Adaptive" adaline" Neuron Using Chemical" memistors."* (Office of Naval Research).
22. Hay JC, Lynch BE, & Smith DR (1960) Mark I perceptron operators' manual. (DTIC Document).
23. Minsky ML (1952) A neural-analogue calculator based upon a probability model of reinforcement. in *Harvard University Pychological Laboratories internal report* (Cambridge, Massachusetts).
24. Mead CA & Mahowald MA (1988) A silicon model of early visual processing. *Neural Networks* 1(1):91-97.
25. Rajendran B & Alibart F (2016) Neuromorphic computing based on emerging memory technologies. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 6(2):198-211.
26. Burr G*, et al.* (2014) Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element. *Electron Devices Meeting (IEDM), 2014 IEEE International*, (IEEE), pp 29.25. 21-29.25. 24.
27. Wong HP*, et al.* (2010) Phase change memory. *Proceedings of the IEEE* 98(12):2201-2227.
28. Vincent AF*, et al.* (2015) Spin-transfer torque magnetic memory as a stochastic memristive synapse for neuromorphic systems. *IEEE Transactions on Bomedical Circuits and Systems* 9(2):166-174.
29. Kaneko Y, Nishitani Y, & Ueda M (2014) Ferroelectric artificial synapses for recognition of a multishaded image. *Electron Devices, IEEE Transactions on* 61(8):2827-2833.
30. Yu S, Wu Y, Jeyasingh R, Kuzum D, & Wong H-SP (2011) An electronic synapse device based on metal oxide resistive switching memory for neuromorphic computation. *Electron Devices, IEEE Transactions on* 58(8):2729-2737.
31. Jo SH*, et al.* (2010) Nanoscale memristor device as synapse in neuromorphic systems. *Nano Letters* 10(4):1297-1301.
32. Wang Z*, et al.* (2017) Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nature Materials* 16(1):101-108.
33. Prezioso M*, et al.* (2015) Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* 521(7550):61-64.
34. Lake BM, Salakhutdinov R, & Tenenbaum JB (2015) Human-level concept learning through probabilistic program induction. *Science* 350(6266):1332-1338.
35. Buczak AL & Guven E (2016) A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* 18(2):1153-1176.
36. Vineyard CM, Verzi SJ, James CD, & Aimone JB (2016) Quantifying neural information content: A case study of the impact of hippocampal adult neurogenesis. *Neural Networks (IJCNN), 2016 International Joint Conference on*, (IEEE), pp 5181-5188.

37. Aimone JB, Deng W, & Gage FH (2011) Resolving new memories: a critical look at the dentate gyrus, adult neurogenesis, and pattern separation. *Neuron* 70(4):589-596.
38. Kristofor D. Carlson FR, James B, Aimone (2017) Computational perspectives on adult neurogenesis. *The Rewiring Brain: A Computational Approach to Structural Plasticity in the Adult Brain*, ed Arjen va Ooyen MB-O (Academic Press), pp 425-441.
39. Dieni CV*, et al.* (2016) Low excitatory innervation balances high intrinsic excitability of immature dentate neurons. *Nat Commun* 7:11313 (collaboration).
40. Severa W, Parekh O, James CD, & Aimone JB (2017) A Combinatorial Model for Dentate Gyrus Sparse Coding. *Neural Comput* 29(1):94-117.
41. Draelos TJ*, et al.* (2017) Neurogenesis deep learning: Extending deep networks to accommodate new classes. *Neural Networks (IJCNN), 2017 International Joint Conference on*, (IEEE), pp 526-533.
42. Vineyard CM, Verzi SJ, James CD, Aimone JB, & Heileman GL (2015) MapReduce SVM Game. *INNS Conference on Big Data,* 53:298-307.
43. Vineyard CM, Verzi SJ, James CD, Aimone JB, & Heileman GL (2015) Repeated play of the SVM game as a means of adaptive classification. *Neural Networks (IJCNN), 2015 International Joint Conference on*, (IEEE), pp 1-8.
44. Vineyard CM & Verzi SJ (2016) Overcoming the Static Learning Bottleneck-the need for adaptive neural learning. *Rebooting Computing (ICRC), IEEE International Conference on*, (IEEE), pp 1-3.
45. Fitzgerald S, Mathews G, Morris C, & Zhulyn O (2012) Using NLP techniques for file fragment classification. *Digital Investigation* 9:S44-S49.
46. Cox JA, James CD, & Aimone JB (2015) A Signal Processing Approach for Cyber Data Classification with Deep Neural Networks. *Complex Adaptive Systems, 2015* 61:349-354.
47. Lee H, Battle A, Raina R, & Ng AY (2007) Efficient sparse coding algorithms. *Advances in Neural Information Processing Systems*, pp 801-808.
48. Hochreiter S & Schmidhuber J (1997) Long short-term memory. *Neural Computation* 9(8):1735-1780.
49. Bugmann G, Christodoulou C, & Taylor JG (1997) Role of temporal integration and fluctuation detection in the highly irregular firing of a leaky integrator neuron model with partial reset. *Neural Computation* 9(5):985-1000.
50. Rothganger F, James CD, & Aimone JB (2016) Computing with dynamical systems. *Rebooting Computing (ICRC), IEEE International Conference on*, (IEEE), pp 1-3.
51. Severa W, Parekh O, Carlson KD, James CD, & Aimone JB (2016) Spiking network algorithms for scientific computing. in *International Conference on Rebooting Computing* (ICRC), San Diego, CA, pp 1-8.
52. Aimone JB, Parekh O, & Severa W (2017) Neural computing for scientific computing applications: more than just machine learning. *Neuromorphic Computing Symposium* (Knoxville, TN), in press.

53. Verzi SJ*, et al.* (2017) Optimization-based computation with spiking neurons. *Neural Networks (IJCNN), 2017 International Joint Conference on*, (IEEE), pp 2015-2022.
54. Follett DR*, et al.* (2017) Neuromorphic data microscope. *Neuromorphic Computing Symposium* (Knoxville, TN), in press.
55. Maass W, Natschläger T, & Markram H (2002) Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation* 14(11):2531-2560.
56. Smith MR*, et al.* (2017) A novel digital neuromorphic architecture efficiently facilitating complex synaptic response functions applied to liquid state machines. *Neural Networks (IJCNN), 2017 International Joint Conference on*, (IEEE), pp 2421-2428.
57. Hill AJ*, et al.* (2017) A Spike-Timing Neuromorphic Architecture. in *International Conference on Rebooting Computing* (ICRC), Washington, D.C., in press.
58. Holler M, Tam S, Castro H, & Benson R (1989) An electrically trainable artificial neural network (etann) with 10240 'floating gate' synapses. *Neural Networks, 1989. IJCNN., International Joint Conference on*, (IEEE), pp 191-196.
59. Le QV (2013) Building high-level features using large scale unsupervised learning. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, (IEEE), pp 8595-8598.
60. Edwards AH*, et al.* (2015) Reconfigurable Memristive Device Technologies. *Proceedings of the IEEE,* 103(7):1004-1033 (collaboration).
61. Rothganger F, Evans BR, Aimone JB, & DeBenedictis EP (2015) Training neural hardware with noisy components. *Neural Networks (IJCNN), 2015 International Joint Conference on*, (IEEE), pp 1-8 (collaboration).
62. Merkel C*, et al.* (2016) Neuromemristive Systems: Boosting Efficiency through Brain-Inspired Computing. *Computer* 49(10):56-64 (collaboration).
63. Mickel PR*, et al.* (2013) Measurement of the polarization vector in BiMnO3 multiferroic thin films using surface and embedded microelectrodes. *Journal of Applied Physics* 114(9):094104.
64. Mickel PR*, et al.* (2013) A physical model of switching dynamics in tantalum oxide memristive devices. *Applied Physics Letters* 102(22):223502.
65. Landon CD*, et al.* (2015) Thermal transport in tantalum oxide films for memristive applications. *Applied Physics Letters* 107(2):023108.
66. Bondi RJ, Fox BP, & Marinella MJ (2016) Role of atomistic structure in the stochastic nature of conductivity in substoichiometric tantalum pentoxide. *Journal of Applied Physics* 119(12):124101.
67. Bondi RJ, Fox BP, & Marinella MJ (2017) Non-metallic dopant modulation of conductivity in substoichiometric tantalum pentoxide: A first-principles study. *Journal of Applied Physics* 121(21):214102.
68. Agarwal S*, et al.* (2015) Energy Scaling Advantages of Resistive Memory Crossbar Based Computation and Its Application to Sparse Coding. *Front Neurosci* 9:484.

69. Agarwal S*, et al.* (2015) The energy scaling advantages of RRAM crossbars. *Energy Efficient Electronic Systems (E3S), 2015 Fourth Berkeley Symposium on*, (IEEE), pp 1-3.

70. Agarwal S*, et al.* (2016) Resistive memory device requirements for a neural algorithm accelerator. *Neural Networks (IJCNN), 2016 International Joint Conference on*, (IEEE), pp 929-938.

71. Fuller EJ*, et al.* (2017) Li-Ion Synaptic Transistor for Low Power Analog Computing. *Advanced Materials* 29(4).

72. van de Burgt Y*, et al.* (2017) A non-volatile organic electrochemical device as a low-voltage artificial synapse for neuromorphic computing. *Nature Materials* 16(4):414-418 (collaboration).

73. Agarwal S*, et al.* (2017) Achieving ideal accuracies in analog neuromorphic computing using periodic carry. *VLSI Technology, 2017 Symposium on*, (IEEE), pp T174-T175.

74. Agarwal S, Schiek RL, & Marinella MJ (2017) Compensating for Parasitic Voltage Drops in Resistive Memory Arrays. *Memory Workshop (IMW), 2017 IEEE International*, (IEEE), pp 1-4.

75. Burr GW*, et al.* (2015) Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Transactions on Electron Devices* 62(11):3498-3507.

76. Jacobs-Gedrim RB*, et al.* (2017) Impact of linearity and write noise of analog resistive memory devices in a neural algorithm accelerator. in *International Conference on Rebooting Computing* (ICRC), Washington, D.C., in press.

DISTRIBUTION

1	MS1169	J. Charles Barbour	1300
1	MS1167	Bryan V. Oliver	1340
1	MS1179	Leonard Lorence	1341
1	MS1320	Samuel S. Collis	1400
1	MS1324	John T. Feddema	1460
1	MS1327	John S. Wagner	1462
1	MS1322	Kenneth F. Alvin	1420
1	MS1319	James A. Ang	1421
1	MS1322	John B. Aidun	1425
1	MS1318	James R. Stewart	1440
1	MS1321	Veena Tikare	1444
1	MS0887	Terrence L. Aselage	1800
1	MS0885	Michael T. Valley	1810
1	MS0886	Blythe Clark	1819
1	MS0887	Shawn M. Dirk	1830
1	MS0959	Deidre Hirschfeld	1832
1	MS0889	Wahid L. Hermina	1850
1	MS1421	Jeffrey S. Nelson	1870
1	MS1415	Carlos Gutierrez	1874
1	MS0440	Reno L. Sanchez	2290
1	MS0980	John C. Rowe	5000
1	MS1079	David R. Sandison	5200
1	MS1071	Clinton A. Boye	5220
1	MS1082	Shanalyn A. Kemme	5228
1	MS1071	Frederick B. McCormick	5260
1	MS1084	Alan Mitchell	5268
1	MS1205	James J. Hudgens	5800
1	MS1209	Leann A. Miller	5860
1	MS0620	James L. Novak	5830
1	MS0621	Nathan D Schwade	5836
1	MS0620	Kevin R. Dixon	5850
1	MS1027	Curtis M. Johnson	5852
1	MS1027	Kristina R. Czuchlewski	5853
1	MS0831	John D. Zepper	6300
1	MS0827	Eunice R. Young	6360
1	MS0810	Aaron D. Niese	6367
1	MS0810	Lisa K. Wilkening	6362
1	MS9054	Robert Q. Hwang	8300
1	MS9161	Christian Mailhiot	8340
1	MS0721	Carol L. J. Adkins	8800
1	MS1138	Stephen D. Kleban	8832
1	MS0736	Richard O. Griffith	8850
1	MS0748	Fredrick M. McCrory	8851
1	MS0801	David R. White	9300
1	MS0801	Julie K. Perich	9310

1	MS0813	Han Wei Lin	9315
1	MS0933	Marcus C. Chang	9360
1	MS0932	Michael J. Haass	9365
1	MS0899	Technical Library	9536 (electronic copy)
1	MS0359	D. Chavez, LDRD Office	1911


Sandia National Laboratories