On the Impact of Approximate Computation in an Analog DeSTIN Architecture

Steven Young, *Student Member, IEEE*, Junjie Lu, *Student Member, IEEE*, Jeremy Holleman, *Member, IEEE*, and Itamar Arel, *Senior Member, IEEE*

***Abstract*— Deep machine learning (DML) holds the potential to revolutionize machine learning by automating rich feature extraction, which has become the primary bottleneck of human engineering in pattern recognition systems. However, the heavy computational burden renders DML systems implemented on conventional digital processors impractical for large-scale problems. The highly parallel computations required to implement large-scale deep learning systems are well suited to custom hardware. Analog computation has demonstrated power efficiency advantages of multiple orders of magnitude relative to digital systems while performing nonideal computations. In this paper, we investigate typical error sources introduced by analog computational elements and their impact on system-level performance in DeSTIN—a compositional deep learning architecture. These inaccuracies are evaluated on a pattern classification benchmark, clearly demonstrating the robustness of the underlying algorithm to the errors introduced by analog computational elements. A clear understanding of the impacts of nonideal computations is necessary to fully exploit the efficiency of analog circuits.**

***Index Terms*— Analog circuits, analog computation, deep machine learning, feature extraction, floating gates.**

I. Introduction

THE mainstream approach for coping with high-dimensional observations has been to employ feature extraction to preprocess the data, reducing its dimensionality to a level that can be effectively processed. As a result, it can be argued that the intelligence behind many pattern recognition systems has shifted to the human-engineered feature extraction process, which can be challenging and highly application-dependent. Moreover, if incomplete or erroneous features are extracted, performance will suffer.

A key factor in realizing deep learning architectures is the resource complexity involved in their training [1]. The computational complexity and storage requirements from deep reinforcement learning systems limit the scale at which they may be implemented using standard digital computers. Custom analog circuitry presents a means of overcoming the limitations of digital very-large-scale integration (VLSI) technology. In order to achieve the largest possible learning system within any given constraints of cost or physical size, it is critical that the basic building blocks of the learning system be as dense as possible. Many operations can be realized in analog circuitry with a space saving of 1–2 orders of magnitude compared to a digital realization. Analog computation also frequently comes with a significant reduction in power consumption, which will become critical as powerful learning systems are migrated to battery-operated platforms.

This massive improvement in density is achieved by utilizing the natural physics of device operation to carry out computation. The benefits in density and power come with certain disadvantages, such as offsets and noise. However, in many cases the feedback inherent to the learning algorithms naturally compensates for inaccuracies introduced by the analog circuits. Similarly, the brain is known to be built from noisy inaccurate neurons. For example, many behavioral responses, such as a fly making a course correction after a disturbance, occur over a period of around 30 ms [2]. Neural signals integrated over comparable time windows typically exhibit a signal-noise ratio (SNR) in the range 1–10 [2]–[4], which is much lower than what can be easily achieved in moderate precision analog electronics (e.g., 8-bit analog-to-digital converters are widely available and exhibit an SNR of over 256). Comparisons between the noise and power tradeoffs in analog and digital circuits and biological systems have also been explored in [5]. The low SNR and outstanding power efficiency of neural systems suggest that relaxed accuracy requirements for electronic computational primitives could allow aggressive optimization for area and power consumption.

This paper describes a study involving the utilization of analog circuits in implementing deep spatiotemporal inference network (DeSTIN) [6], [7]—a state-of-the-art deep machine learning (DML) architecture. The aim of this paper is to demonstrate the scalability potential of realizing DML using analog VLSI. The challenge in designing analog circuits for computational purposes lies in accounting for their various inaccuracies and inconsistencies on the overall system performance. We thus focus our attention on a detailed analysis of the implications of realizing each of the key functions performed by the DeSTIN nodes. Through the application of the proposed architecture on a common large-scale classification problem, very promising implementation characteristics are demonstrated. Moreover, such designs can be fabricated using existing VLSI technology, which results in fabrication time and costs that are the same as digital ASIC designs.

Manuscript received November 19, 2012; revised September 13, 2013; accepted September 17, 2013. Date of publication October 10, 2013; date of current version April 10, 2014. This work was supported in part by the Intelligence Advanced Research Projects Activity via Army Research Office under Grant W911NF 12-1-0017 and in part by NSF under Grant CCF-1218492.

The authors are with the Department of Electrical Engineering and Computer Science, The University of Tennessee, Knoxville, TN 37996 USA (e-mail: syoung22@eecs.utk.edu; jlu9@eecs.utk.edu; jhollema@eecs.utk.edu; itamar@eecs.utk.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2283730

Although the focus of this paper is the implementation of the DeSTIN architecture in analog circuitry, the results are relevant to many other DML architectures. Since many DML architectures do not rely on an extremely precise calculation of some final result and perform many similar mathematical operations to those explored here, the results are useful to explore the feasibility of implementing other DML methods in analog circuitry.

The remainder of this paper is structured as follows. Section II provides an overview of DML and the DeSTIN architecture. In Section III, the key analog circuits constituting the DeSTIN implementation are described, along with their typical inaccuracies. Section IV provides a discussion on modeling the inaccuracies at a system level and provides simulation results that quantify the impact on overall performance in the context of a large-scale classification task. In Section V, we present our conclusions and some thoughts regarding future work.

II. Overview of Deep Machine Learning

DML architectures have recently emerged as promising biologically inspired frameworks for effectively modeling complex signals. The efficacy of machine learning methods has a strong dependence on the features they are applied to. Currently, these features are mostly chosen and created using labor-intensive feature engineering. Deep learning methods try to learn transformations of the data that form these useful features. DML methods can be split into three main approaches: the probabilistic models (e.g., sparse coding and Boltzmann machines), the reconstruction-based algorithms (e.g., auto-encoders), and manifold-learning approaches [8]. They have achieved state-of-the-art results on benchmarks spanning a broad range of application domains [9], [10]. In DML, a hierarchical architecture for information representation is employed whereby higher layers of the hierarchy represent broader and more abstract concepts pertaining to the signal modeled. In addition to the natural high dimensionality of real-life data, the temporal component also plays a key role. A sequence of patterns that a human observes often conveys a meaning to the observer, whereas independent fragments of this sequence would be hard to decipher in isolation. Humans often infer meaning from events or observations that are received close in time. To that end, modeling the temporal component of the observations plays a critical role in effective information representation. Capturing spatiotemporal dependencies, based on regularities in the observations, can therefore be viewed as a fundamental goal for deep learning systems.

Compositional deep learning architectures are a particular family of DML systems. They are characterized by hosting multiple instantiations of a basic cortical circuit (or node) which populate all layers of the architecture. Each node is tasked with learning to represent the sequences of patterns that are presented to it by a unique subset of nodes in the layer that precede it. The nodes at the very lowest layer of the hierarchy receive as input raw data (e.g., pixels of the image) and continuously construct a belief state that attempts to compactly characterize the sequences of patterns observed. The second layer, and all those above it, receive as input the belief states of nodes at their corresponding lower layers, and attempt to construct their own belief states that capture regularities in their inputs. Information flows both bottom up and top down. Bottom-up processing essentially constitutes a feature extraction process in which each layer aggregates data from the layer below it. Top-down signaling helps nodes in lower layers improve their representation accuracy by assisting in correctly disambiguating distorted observations.

The goal of the learning process in an unsupervised DML architecture is to form a hierarchical feature space that can be employed by classification or regression models. The learning process at each node is unsupervised, guided by exposure to a large set of observations and allowing the salient attributes of these observations to be captured across the layers. This extracted information should exhibit invariance to common distortions and variations in the observations, leading to representational robustness. In the context of visual data, robustness refers to the ability to exhibit invariance to a diverse range of transformations, including mild rotation, scale, different lighting conditions, and noise.

It should be noted that, although deep architectures may appear to completely solve or overcome the curse of dimensionality, in reality they do so by hiding the key assumption that some structure or locality exists in the data, which allows sparsity to exist in the architectures. The latter means that the dependencies that may exist between two signals (e.g., pixels) that are spatially close are captured with relative detail, whereas relationships between signals that are distant (e.g., pixels on opposite sides of a visual field) are represented with very little detail. This is a direct result of the nature of the contracting hierarchical architecture, in which fusion of information from inputs that are distant to the hierarchy occurs at the higher layers. It is also important to emphasize that deep learning architectures are not limited by any means to visual data, but to any data where there is some kind of structure or locality such as music audio [11]. These architectures are modality agnostic, and attempt to discover the underlying structure in data of any form [10].

A. Scalability Issues With Software Realizations of DML

Deep layered architectures offer excellent performance attributes. However, the computation requirements involved in training and utilizing them grow dramatically as the dimensionality of the input space increases. Compositional deep layered architectures compose multiple instantiations of a common cell—the node. In CPU-based platforms, in contrast to the inherent concurrency of deep layered architectures, processing is performed sequentially, thereby greatly increasing the execution time. This is the reason why many recent efforts have been directed toward implementing DML architectures using graphic processing units (GPUs) [12]. While GPUs outperform CPU-based realizations, both in computation time and cost/performance ratio, they consume a large amount of power. Custom hardware presents the opportunity to achieve large power savings over a GPU implementation. GPUs are designed to perform 32-bit accurate computations. However,

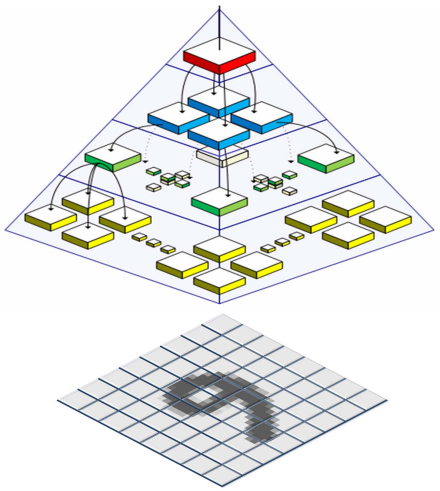

Fig. 1. DeSTIN hierarchical architectures employed for visual sensory processing.

as we explore later, the actual accuracy needed in a DML architecture may be much less than that. Thus, by taking advantage of these relaxed accuracy constraints and leveraging a physical circuit designed specifically for the task at hand, orders of magnitude in power savings can be realized.

B. DeSTIN Architecture

The DML architecture considered here is the DeSTIN architecture, first introduced in [6]. DeSTIN comprises multiple instantiations of an identical cortical circuit, or node. Each node is a parameterized model which learns by means of an unsupervised learning process. These nodes populate all layers of the hierarchy where every node attempts to capture the salient spatiotemporal regularities exhibited by patterns it is presented with. The nodes at the lowest layer of the hierarchy receive as input raw sensory data (e.g., pixels of an image) and continuously construct a belief state that attempts to characterize the sequences observed. The inputs to all layers other than the first are the belief states of nodes at their corresponding lower layers. The beliefs formed across the hierarchy are then used as rich features provided to a classifier or regression learner that can be trained using supervised learning.

Hence, the core process performed at each node is updating of its belief construct. This update rule is expressed through the following conditional probability function, which is based on the probability of an observation given a particular system state, $\Pr(o|s')$; the probability of that system state given a previous state and some higher layer advice, $\Pr(s'|s,a)$; and the belief state of a previous state, $b(s)$:

$$b'(s') = \frac{\Pr(o|s')\sum_{s\in S}\left[\Pr(s'|s,a)b(s)\right]}{\sum_{s''\in S}\left[\Pr(o|s'')\sum_{s\in S}\left[\Pr(s''|s,a)b(s)\right]\right]} \tag{1}$$

which maps observation o, belief b (which is a function of the system state s), and a high-layer node's belief (i.e., advice) a to a new (updated) belief state $b'(s')$. The denominator is a normalization factor. This equation can be viewed as a posterior over the observations $\Pr(o|s')$ that is modulated by a construct that reflects the system dynamics $\sum_{s\in S}\Pr(s'|s,a)b(s)$. These two building blocks of the architecture are the pieces of information which must be learned from the data.

An online clustering algorithm based on the winner-take-all (WTA) competitive learning approach is employed in order to learn $\Pr(o|s')$ [13]. This clustering algorithm captures regularities in the inputs by forming centroids characterized by a mean μ and variance σ^2 in each dimension.

In order to facilitate an analog implementation, several simplifications have been made to the original version of the clustering algorithm used in DeSTIN. The winning centroid is chosen based solely on the Euclidean distance. The distance d_x between a centroid x and an observed input o is expressed as

$$d_x = \|o - \mu_x\|\,\psi_x. \tag{2}$$

The starvation trace ψ is a technique employed by the clustering algorithm to include centroids which are initialized too far from regions of density in the observation space, and are thus at risk of never being selected for update. This allows idle or starved centroids to shrink their apparent distance to all input vectors over time when they are not the selected centroid, and increase the apparent distance when they are the selected centroid. A starvation trace value ψ_c is maintained for each centroid and is decayed by a small constant $0 \leq \gamma \leq 1$ each time that centroid is not updated and increases once the centroid is selected, as reflected by

$$\psi_c = \gamma\,\psi_c + (1-\gamma)\,\mathbb{1}_{x=c} \tag{3}$$

where x represents the chosen centroid. The starvation trace modulates the distance calculation and thus gives "starved" centroids an opportunity to move toward data samples. The estimated mean of the winning centroid, μ_x, is updated toward the current input in conjunction with estimated variance σ_x^2 such that

$$\mu_x \leftarrow \mu_x + \alpha(o - \mu_x) \tag{4}$$

$$\sigma_x^2 \leftarrow \sigma_x^2 + \beta\left[(o-\mu_x)^2 - \sigma_x^2\right] \tag{5}$$

where α and β are positive numbers close to 0. The posterior distribution $\Pr(o|s')$ is then obtained using the normalized Euclidean distance between the input and each centroid s, such that

$$n_s = \sum_{i=1}^{d}\frac{(o_i - \mu_{i,s})^2}{\sigma_{i,s}^2} \tag{6}$$

$$p_s = \frac{n_s^{-1}}{\sum_{s'\in S}(n_{s'}^{-1})}. \tag{7}$$

Most relevant to this paper is the fact that DeSTIN offers low sensitivity to many of the typical errors inherent in analog computation hardware. For example, it is of low importance that each node partitions the inputs into a specific set of centroids. Rather, it is imperative that the centroids found represent some regularities pertaining to the underlying structure of the observations. As a result, many of the gain (i.e., multiplicative) and offset (i.e., additive) errors that are

inherent in analog devices do not prevent the nodes from forming meaningful beliefs, as discussed in more detail below. It is also not important that the centroids be located perfectly in the computed mean point they represent or that they are paired with precise estimates of the variances. Thus, updates in the estimates of the means and variances can be subject to error while still allowing the centroids to yield meaningful beliefs. Since no external mechanism has direct access to the centroids, the consequences of a bias errors on the inputs are marginal given that the error will map all inputs to a new space. Inaccuracies that result in gain differences between the various dimensions of the centroids may alter the centroids that are formed, but do not prevent the system from discovering and representing regularities in the data. Sensitivity analysis with regard to gain and bias errors is important in mapping an algorithm to an analog architecture. Since the primary goal of DeSTIN is to form meaningful beliefs that capture regularities in the data, the architecture is relatively insensitive to the errors an analog implementation typically injects into the computation processes involved.

III. Analysis of Analog Computation

As mentioned above, large-scale DML systems have the potential to be uniquely effective at solving complex high-dimensional problems. However, their computational complexity limits the scale at which they may be implemented in standard digital computers. We next focus on the design and study of custom integrated circuits which facilitate the realization of a large-scale DML architecture. In order to achieve the largest possible learning system within any given constraints of cost, power consumption, or physical size, it is critical that the basic building blocks of such system be as compact and efficient as possible.

Many operations can be realized in analog circuitry with a space and power saving of up to three orders of magnitude compared to a digital implementation [14]–[17]. For example, simulations of the analog circuit shown in Fig. 8, which implements the function x^2/y, and a digital circuit with equivalent resolution and functionality indicate that the analog circuit consumes more than an order of magnitude less energy per operation than the digital circuit. Such savings can be critical in large-scale data-mining operations, where data throughput is often limited by power dissipation [18]. Additionally, ultralow-power learning systems will enable the embedding of powerful learning functionality into battery-operated portable devices.

The cost of these power and area savings is computation that is altered by error sources intrinsic to analog operation of the transistors. While designing computational circuits for minimum power and area, the circuit designer must avoid introducing errors that will severely degrade the learning performance of the larger system. In this section, we examine the impact of analog circuit nonidealities on learning performance to provide guidance to circuit and system designers, allowing them to aggressively optimize circuit design for density or power consumption while maintaining sufficient learning performance. We next outline the key computational building blocks and their analog hardware considerations.

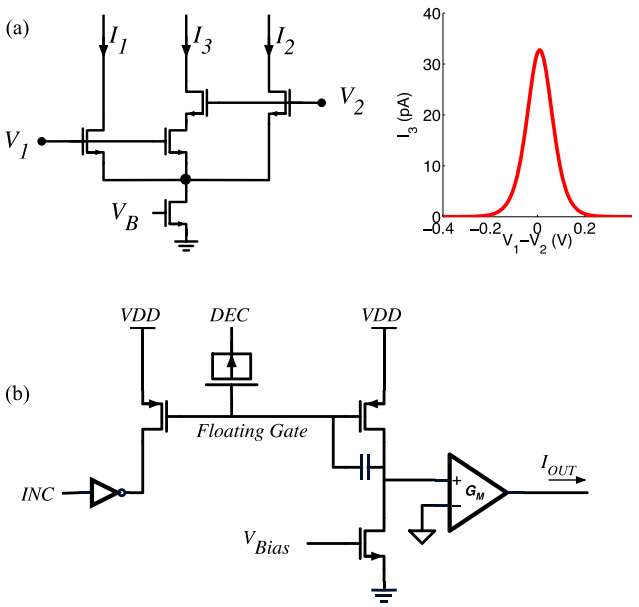

Fig. 2. (a) Bump circuit, which computes $\tanh(V_1 - V_2)$ and its derivative simultaneously. (b) Floating-gate memory using feedback for improved update linearity.

A. Analog Computational Elements

The basic computations needed by machine learning algorithms include storage, multiplication, addition, activation functions, and similarity calculations, all of which can be realized in a compact and power-efficient manner with analog circuits. For example, the differential pair, a mainstay of classical analog design, provides a $\tanh(\cdot)$ function at its output current. The addition of two more transistors yields a "bump circuit" [19], shown in Fig. 2, which computes the derivative of $\tanh(\cdot)$. The bump circuit simultaneously provides a measure of similarity between two inputs and the $\tanh(\cdot)$ of their difference. The bump function can also be used as a probability distribution, as it peaks with zero difference and saturates to zero for large differences.

The bump circuit illustrates the power advantage that analog computation holds over digital methods. A bump circuit, biased at 200 fA, can evaluate the similarity between a stored value and about 200 observations per second, according to simulations using 0.24-μm transistors. A single inverter consumes about four times that much current when switching at 200 Hz, and about half that much statically, without switching at all. To perform a comparable computation digitally would require dozens more transistors and one to two orders of magnitude more current. Higher clock frequencies can allow digital systems to perform more operations per unit of energy, but system-level issues such as latencies in memory access and delay variations result in faster digital circuits waiting—and leaking current—for significant amounts of time, thus reducing efficiency.

Summation is trivial in the current domain, as it is accomplished by joining the wires with the currents to be summed. In the voltage domain, a feedback amplifier with $N+1$ resistors in the feedback path can compute the sum of N inputs. A Gilbert

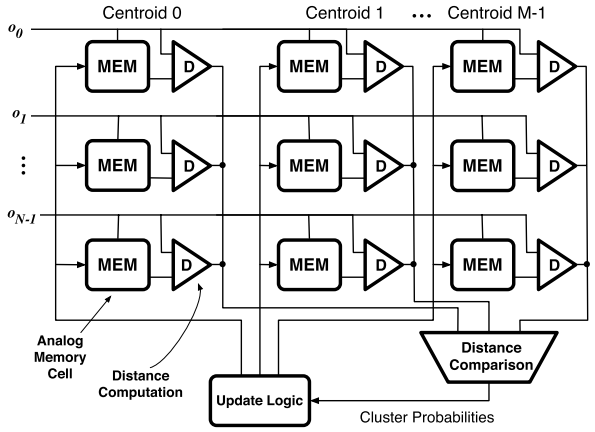

Fig. 3. Block diagram of an analog clustering circuit. o_i refers to the dimension i of the input.

cell [20] provides four-quadrant multiplication while using only seven transistors.

B. Floating Gate Storage and Adaptation

Floating gates have proved themselves useful in many different applications, such as programmable switches [21], threshold matching in transistor circuits [22], and as a storage element in analog adaptive systems [23]–[26]. In learning systems, a floating gate can be used to store a weight or learned parameter. Since floating gates are electrically isolated by silicon dioxide, they retain a fixed quantity of charge under normal conditions, providing an analog nonvolatile memory. Through Fowler–Nordheim tunneling, or hot electron injection, charge can be added to or removed from the floating gate. The stored value can be read out as a voltage or current. Nonlinearities in the update mechanisms can be mitigated using feedback linearization [25], [27] or predistortion [16]. In this paper, we propose to employ floating gate memories to store learned centroids.

C. Mapping Architecture to Circuits

A generic clustering architecture is illustrated in Fig. 3, showing an N-dimensional input and M centroids. For the purpose of this analysis, we will assume a discrete-time clustering process. The inputs are the observations $o_{0:N-1}$ and the outputs are the class probabilities $p_{0:M-1}$. The N-dimensional input is classified on the basis of the distance between the observation and the respective centroids $\mu_{0:M-1}$. Each input o_i is compared to the corresponding component $\mu_{i,j}$ of every centroid to produce a 1-D distance $d_{i,j}$. The 1-D distances are then squared and summed across the input dimensions to yield a total distance D_j between the observation and each centroid. The distances are then compared to each other and converted into class probabilities $p_{0:M-1}$. A variety of distance functions, or equivalently similarity functions, may be used. The memories constituting each centroid are then updated based on the respective class probability, such that the new value of a given memory element is a function of the previous value and the set of centroid-observation distances. One step of the complete clustering process can be represented using general mapping functions by the following steps:

$$d_{i,j} = f_1(o_i, \mu_{i,j}) \tag{8}$$

$$D_j = f_2(d_{0:N-1,j}) \tag{9}$$

$$p_{0:M-1} = f_3(D_{0:M-1})\ (eqs.\ 6\&7) \tag{10}$$

$$\mu_{i,j}[t+1] = f_4(\mu_{i,j}[t], o_i[t], D_{0:M-1}[t]). \tag{11}$$

In order to analyze the effects of no-ideal computations, we will use representative cases for the general equations shown in (8)–(11). Employing the Euclidean distance and a moving average update rule for estimating the variances, as mentioned above, we obtain the following set of equations:

$$d_{i,j} = \frac{(o_i - \mu_{i,j})^2}{\sigma_{i,j}^2} \tag{12}$$

$$\sigma_{i,j}^2[t+1] = \sigma_{i,j}^2[t] + \lambda_\sigma \left(\left(o_i - \mu_{i,j}\right)^2 - \sigma_{i,j}^2[t] \right) \tag{13}$$

$$D_j = \sum_{i=0:N-1} d_{i,j} \tag{14}$$

$$p_{0:M-1} = f_3(D_{0:M-1})\ (eqs.\ 6\&7) \tag{15}$$

$$\mu_{i,j}[t+1] = \begin{cases} \mu_{i,j}[t] + \lambda(o_i - \mu_{i,j}), & \text{if } D_j = \min(D_{0:M-1}) \\ \mu_{i,j}[t], & \text{otherwise.} \end{cases} \tag{16}$$

Here $d_{i,j}$ is the 1-D distance between the ith component of the observation and the corresponding component of the jth centroid. D_j is the distance from the observation and the jth centroid, aggregated across dimensions. The standard deviation of the ith dimension of all observations attributed to the jth centroid is represented by $\sigma_{i,j}$, such that $\sigma_{0:\vec{N}-1,j}$ is the diagonal of the covariance matrix of the data attributed to centroid j. The memory value $\mu_{i,j}$ is the learned value of the ith element of the jth centroid. The learning rates are represented by λ and λ_σ for the centroid location and the centroid standard deviation, respectively. Note that the operations involved in realizing these equations are merely addition, subtraction, multiplication, division, and absolute value. The first two operations are trivial in current domain, while the others can be implemented quite effectively, as discussed next.

D. Analog Error Sources

The analog error sources of primary concern are mismatch and noise. Random variation or mismatch occurs as a result of manufacturing variation in circuit components [28, Sec. 9.7]. Variation in the threshold voltage of MOS transistors is the predominant source of mismatch. Because a modification to the threshold voltage has the same effect on circuit behavior as a shift in the gate voltage, threshold variation results in an bias error for voltage-mode signals. Typical values range from a few millivolts to tens of millivolts. In the case of current-mode signals, the effect of threshold variation depends on the transistor's operating region. However, analog circuits most often employ weak inversion (subthreshold) operation due to the computational flexibility of the exponential current–voltage relationship in weak inversion [29, Ch. 3]. $I_{\mathrm{DS}} = I_S e^{V_{\mathrm{GS}}/U_T}$,

where I_S is a device-specific constant, U_T is the thermal voltage $k_B T/q$ (about 25 mV at room temperature), and V_{GS} is the gate–source voltage. Even though the notion of a "threshold voltage" is not precisely applicable to subthreshold operation, variations in the physical properties that are manifested in the threshold voltage still have the same effect as a shift in the gate voltage. Therefore, we can represent the current in a transistor with a shift in the threshold (relative to the mean value) as

$$I_{\mathrm{DS}} = I_S e^{\frac{V_{\mathrm{GS}}-\Delta V}{U_T}} = I_S e^{\frac{V_{\mathrm{GS}}}{U_T}} e^{\frac{-\Delta V}{U_T}}$$

where ΔV represents the variation in threshold voltage. It can be seen that the term $\exp -\Delta V/U_T$ results in a gain error that will be applied to any input signal. Threshold voltage variations in the range 5–50 mV will result in gain errors ranging from 1.2 to 7.4 (or 0.14–0.85 for negative shifts).

We can represent the gain errors typical of current mode signals as $x' = \alpha x$, where x is the ideal value and x' is the actual value. Similarly, the bias errors common to voltage-mode signals can be represented as $x' = x + \beta$. The gain and offset, represented by α and β, are random variables, with means of unity and zero, respectively.

We now describe the effects of transistor mismatch on the clustering process. Most of the error sources will, in general, not be correlated across either the dimensions or the centroids because the errors are generated by independent transistors, so the error term for each unit may be considered as an independent and identically distributed variable. The standard deviation of threshold voltage is inversely proportional to the transistor's area, so performance can be improved by using large transistors. Thus, a reduction in the errors comes at the cost of decreased density and increased capacitance, which in turn degrades bandwidth and computational throughput.

Input Variation: Each dimension of the input signal must be copied to each of the centroids. In current-mode signaling, this will be accomplished via a current mirror, and the input will be multiplied by a gain error. With voltage signaling, the same input can be physically shared between centroids without explicit copying, and this error source is avoided. This variation will in general not be correlated either across dimensions or across centroids. This results in modifications to (12) and (16) such that for a gain variation $d_{i,j} = f(\alpha^i_{i,j} o_i, \mu_{i,j})$ and $\mu_{i,j}[t+1] = g(\mu_{i,j}[t], \alpha_{i,j} o_i[t], D_j[t])$. In this case, the factor $\alpha^i_{i,j}$ is the same in both equations but will be different for every centroid and for every dimension within a centroid.

Fig. 4 shows the variation of signals distributed through a current mirror as both current and transistor area are varied. As can be seen, current has only a modest effect on mismatch in the subthreshold regime. As the bias current is increased into the microampere range, the transistor will enter strong inversion and the effect of threshold variation will diminish. However, to minimize power consumption in systems with many thousands of nodes, individual transistor currents must be maintained at a much lower level. In contrast, current mismatch displays a nearly ideal inverse relationship with transistor area, as predicted by [30], with errors below 20% attainable with an area of 1 μm^2. Current mismatch can occur in many locations throughout a computational circuit,

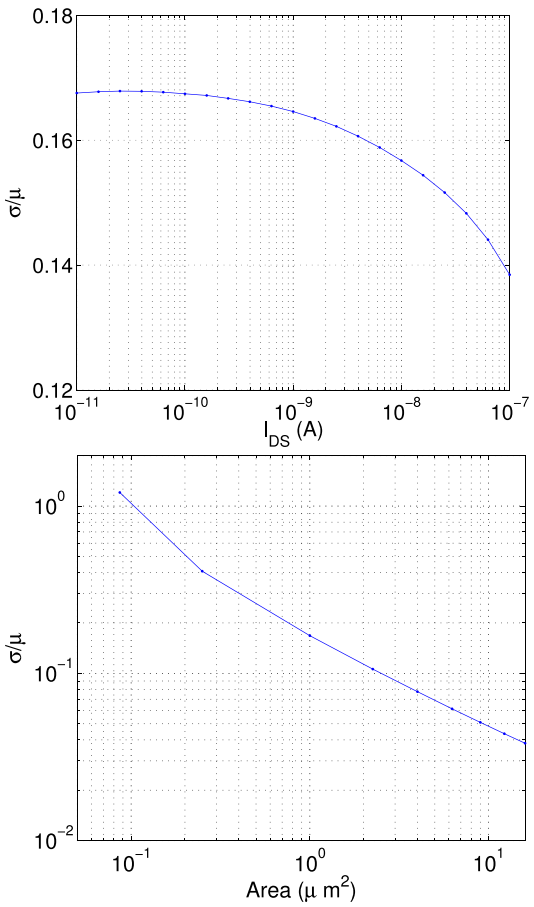

Fig. 4. Simulated current mirror mismatch as a function of (top) bias current with area fixed at 1 μm^2 and (bottom) transistor area with current fixed at 1 nA. Mirror mismatch is plotted as the ratio of the standard deviation of a current signal to its mean.

as described below. Because the underlying mechanism is the same as that for a current mirror, the error magnitudes can be expected to be similar to that shown in Fig. 4.

If this input variation is modeled as a bias error, it has no system-level impact. In this case it will simply shift the location of the origin of the input space. Since the clustering is concerned solely with the distance between the centroids and the input, this error will have no effect on this relationship and thus no effect on the ultimate output of the clustering node, the belief state. If this error is modeled as a gain error, it will have the effect of giving certain dimensions of the input more or less weight in the distance calculations and in the calculation of the belief state. The impact of this error will be dependent on the characteristics of the true data clusters. If they are well separated, no significant impact should be expected. If they are not, the system could characterize the input in a different manner than expected if each input dimension were given equal weight as demonstrated in Fig. 5.

Update Asymmetry and Variation: Mismatch in the update mechanisms may result in asymmetric updates, such that increments to a given memory are of a different magnitude than decrements. Also, because of variation in the update mechanism, the update rate may vary from one memory cell to another.

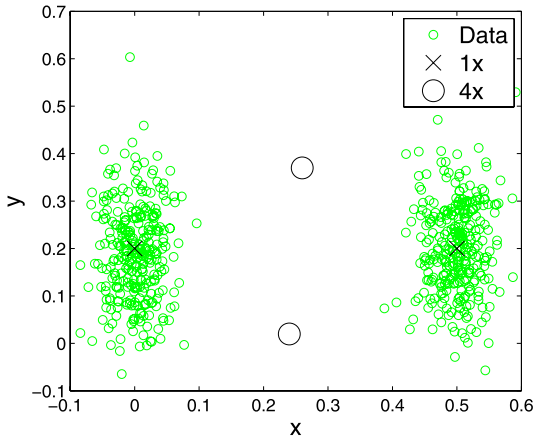

Fig. 5. Input gain. Gain errors can cause the input to form centroids in a skewed space. This figure demonstrates the effect of a gain error that is too large, $4x$, in the second dimension of each centroid versus a case with no gain error.

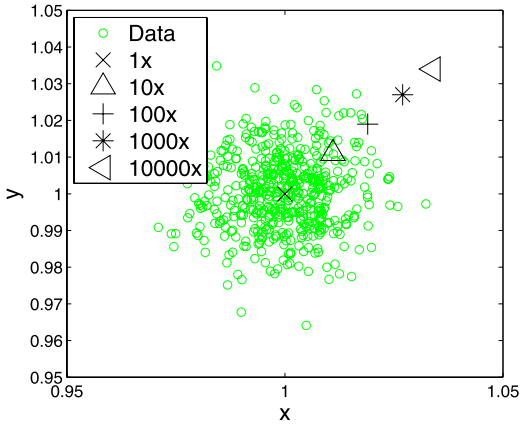

Fig. 6. Centroid offset. As the size of the increments relative to the decrements becomes larger, the centroid becomes more offset from the true center of the data. Depending on the nature of the data being clustered over, the amount of error that is acceptable may vary. It is also important to remember that extremely accurate clustering is not needed, and even with fairly inaccurate clustering meaningful beliefs can be calculated.

An asymmetric error will have some system-level impact. During the transient stage of centroids moving toward their clusters, this error will simply cause the learning rate to be modified. During the steady-state stage of centroids learning the mean and variance of their clusters, this will cause the mean to be offset within the cluster and the calculated variance to be larger. The expected offset is expected to be proportional to the mismatch between increment and decrement learning rates, λ^+ and λ^-, respectively. Assuming the increment is larger than the decrement, if the number of increments and decrements are equal, the centroid will move upward. However, as the centroid moves upward, the number of decrements will increase and cause the centroid to reach equilibrium

$$\lambda^+ Pr(X > \mu) = \lambda^- Pr(X < \mu). \tag{17}$$

A demonstration of this effect can be seen in Fig. 6. Variation in the update rate between memory cells will have the effect of some centroids converging toward their equilibrium faster than others. This will have no effect on the equilibrium state of the centroids and no significant effect on the learning transient except in extreme cases.

Memory Adaptation Variation: Each analog memory cell will tend to converge toward the inputs applied to it when it is updated. Input-referred offset or gain error here will cause the memory to converge to a value different than the actual mean of the cluster it is learning. This error, $\alpha^m_{i,j}$, can be represented by modifying the error term applied to the observation for the update equation with respect to that applied for the distance measurement

$$\mu_{i,j}[t+1] = \begin{cases} \mu_{i,j}[t] + \lambda \alpha^m_{i,j}(o_i - \mu_{i,j}), & \text{if } D_j = \min(D_{0:M-1}) \\ \mu_{i,j}[t], & \text{otherwise.} \end{cases} \tag{18}$$

If the memory adaptation variation is modeled as a gain error as in (18), it will have the effect of increasing or decreasing the learning rate. Much like the update variation error, it will have little to no effect except in extreme cases. However, if this error is modeled as an bias error, it will have the effect of shifting the learned centroid by the amount of the error. For small errors, this will have a small effect on the calculation of the belief state. When the error becomes larger, it can cause a centroid to walk far away from the data cluster it is supposed to represent and possibly toward another data cluster. The effect of such a case is demonstrated in Fig. 7.

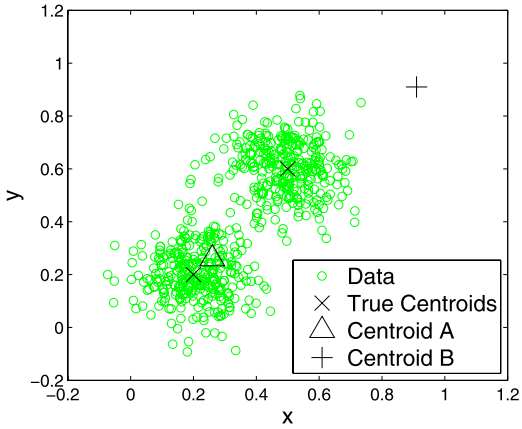

Fig. 7. Memory adaptation bias. This figure demonstrates what happens when one centroid has a bias error in the memory adaptation. In this figure, Centroid B has a bias error toward the top right of the plot. As it moves toward the other data cluster, it swaps positions with Centroid A before reaching a steady-state position offset from its data cluster by approximately the magnitude of the bias error.

Distance Error: The distance measurement circuits may exhibit gain and/or offset errors. Variation within a given centroid could result in one dimension contributing disproportionately to the overall distance between an observation and an input. Circuitry accepting the individual 1-D inputs will contribute to error uncorrelated across dimension within a given centroid. Circuitry operating on the aggregated distance will contribute to an error that affects each dimension identically. We will represent gain and offset error in the 1-D distances by $\alpha^d_{i,j}$ and $\beta^d_{i,j}$, respectively, which will include components that are uncorrelated across dimensions as well as correlated components. As a point of reference, the output of

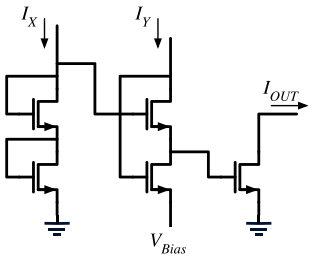

Fig. 8. Translinear circuit for computing x^2/y needed to obtained the Mahalanobis distance and belief calculation from (6) and (7).

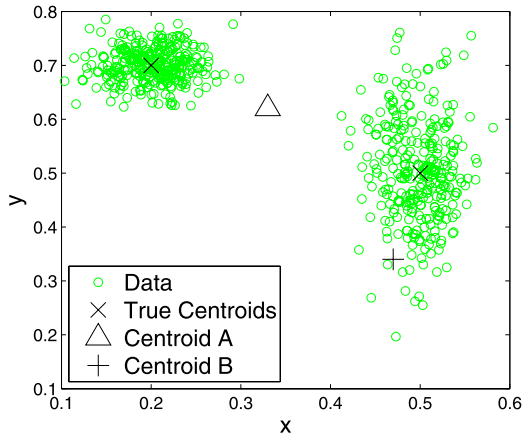

Fig. 9. Illustration of a distance calculation error within the clustering process. . Centroid B has an error causing it to appear artificially more distant from any input. This results in Centroid A to claiming some observations that should belong to Centroid B.

the distance computation shown in Fig. 8 has a normalized standard deviation (σ/μ) of about 2.7%.

For reasonable error levels, distance error has the effect of giving one dimension more or less importance is the centroid selection process much like the case of the input gain error demonstrated in Fig. 5.

Distance Comparison: There may be input-referred offset in the distance comparison block, typically implemented as either a WTA circuit for similarity or a loser-take-all for difference. The effect of this can be expected to be similar to a distance error that is identical for all dimensions of a given centroid but varies across centroids. Gain errors in the distance comparison circuitry are denoted as α_j^D.

On a system level, distance comparison error will have the effect of causing inaccurate belief state calculations and making a centroid appear artificially further from (or closer to) all inputs in the selection algorithm. If this error is small, it will have no effect since the distance comparison is only used to select the centroid to update. If the error is larger, a centroid could become starved until this distance comparison error comes into equilibrium with the starvation trace. Once this equilibrium is reached, the centroid will still claim fewer input vectors than it should, since its starvation trace cannot remain small enough to claim input vectors without allowing other centroid to claim more input vectors. This effect is demonstrated in Fig. 9.

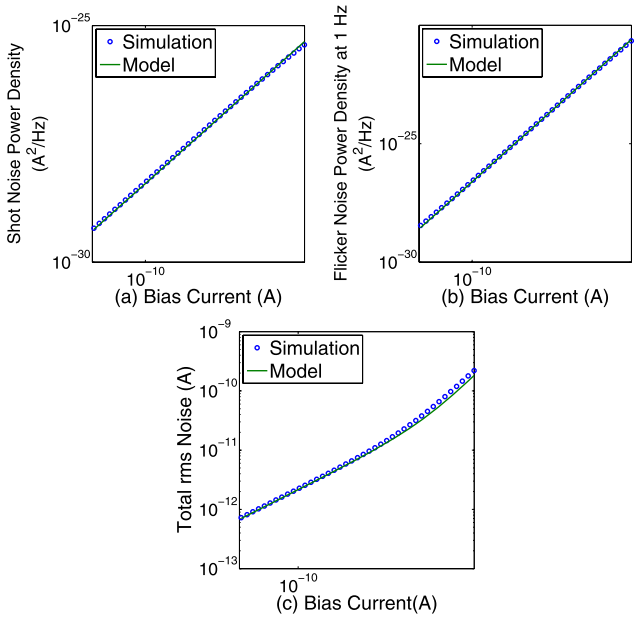

Fig. 10. Modeling of the current noises in a transistor as a function of bias current. (a) Shot noise. (b) Flicker noise. (c) Total integrated RMS noise.

Memory Leakage: Some analog memory cells, such as switched-capacitor cells or floating-gate memories with thin gate oxides [31], gradually lose charge from the storage node, resulting in an undesired change to the stored centroid locations. The stored value may drift toward either of the supply rails or, less frequently, toward an equilibrium level in the middle of the storage range. Because the leakage occurs even in the absence of intentional updates to the memory, its magnitude is dependent on the frequency of updates; increased time between updates results in increased leakage. Assuming that the operating frequency is constant, memory leakage $\gamma_{i,j}^{lk}$ results in a modification to (16): $\mu_{i,j}[t+1] = g(\mu_{i,j}[t], o_i[t], D_j[t]) + \gamma_{i,j}^{lk}$. Because floating-gate memories utilizing thick-oxide transistors [25], [32] can effectively eliminate leakage, we will not further address it in this paper.

Noise: Each of the signals is an analog current or voltage and is thus subject to additive noise. The noise is typically Gaussian. It may be spectrally white, such as shot noise and thermal noise, or pink concentrated at low frequencies, such as flicker noise. In a metal–oxide–semiconductor field-effect transistor biased in the subthreshold region, the primary noise sources are shot noise and flicker noise. Shot noise stems from individual electrons flowing through a junction at random intervals. It is a broadband white noise with power density given by $I_{n,\text{shot}}^2 = 2qI_D$, where I_D is the dc bias current. As bias current increases, the shot noise power increase with I_D while the signal power increase with I_D^2, making it possible to trade power consumption with SNR. At low frequencies, flicker noise dominates. It has a $1/f$ power law and can be attributed to the fluctuation of number of carriers and/or their effective mobility in the channel. Its power density is inversely proportional to the size of the transistor and proportional to the square of bias current, $I_{n,\text{flicker}}^2 = K_f I_D^2/A$, where K_f is a process-

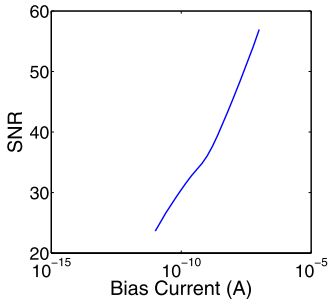

Fig. 11. SNR as a function of bias current in the analog floating-gate memory.

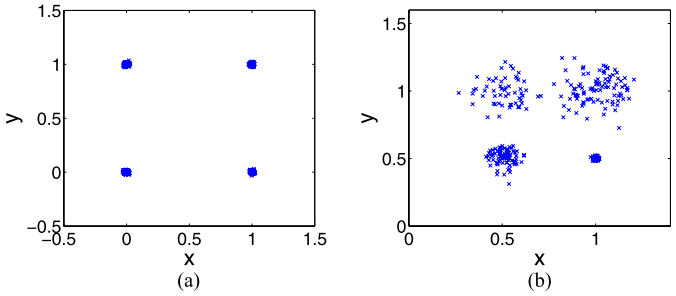

Fig. 12. (a) Clean and (b) noisy synthetic clustering data used for evaluation of analog computation inaccuracies.

dependent fit constant and A is the area of the transistor. In Fig. 10(a) and (b), the expected noise currents as a function of bias current in a $1-\mu\text{m}^2$ sized transistor based on the models described above are compared to the simulation results obtained using foundry-provided models and show good agreement. The high-frequency drain current noise simulations indicated excess noise beyond that predicted by the shot current model. This excess noise is fitted using a correction factor of 1.2 applied to the shot noise model. The total root-mean-square (RMS) noise current can be obtained by integrating shot and flicker noise respectively over the system noise bandwidth and then adding them in power. The result from our noise model is compared to the simulation and shows a good agreement, as in Fig. 10(c).

Based on the analysis on a single transistor above, it can be seen that there exists a tradeoff between the SNR and the power consumption. This can be illustrated in Fig. 11. In the figure, the SNR of the floating-gate memory shown in Fig. 2 is plotted against its bias current, which is proportional to the power consumption. From this noise versus power tradeoff, designers can estimate the available room for power scaling down given the lower boundary of SNR acceptable by the learning algorithm. The continuous-time noise in the current or voltage signal is converted into a discrete-time noise process when the signal is effectively sampled by the distance comparison to determine the class probabilities and centroid updates. For the purposes of this investigation, we will assume that this sampling results in a white noise sequence.

Noise should have no effect on the calculated centroid means since it is a zero-mean process. For clusters with a smaller variance than the noise level, the calculated variance will be similar to the noise level rather than similar to the true variance. If the true variances are less than the noise level for all centroids, then the beliefs will be calculated on the basis of a distance measure that is approximately the Euclidean distance than the normalized Euclidean distance because of the inaccurate calculated variances. The beliefs will then have less information than they otherwise would, but the beliefs will still be meaningful because of the accurate mean values.

Incorporating the nonideal effects into (19)–(23) yields the following relationships, where any n^x denotes additive noise occurring at the same location in the circuit where α^x occurs:

$$d_{i,j} = \frac{(\alpha^i_{i,j}\alpha^d_{i,j}o_i - \mu_{i,j} + n^o_{i,j}[t])^2}{\sigma^2_{i,j}} + n^d_{i,j}[t] \tag{19}$$

$$D_j = \sum_{i=0:N-1} \alpha^D_j d_{i,j} + n^D_{i,j}[t] \tag{20}$$

$$p_{0:M-1} = f_3(D_{0:M-1})\ (eqs.\ 6\&7) \tag{21}$$

$$\delta_{i,j} = (\alpha^i_{i,j}\alpha^m_{i,j}o_i - \mu_{i,j}) \tag{22}$$

$$\mu_{i,j}[t+1] = \begin{cases} \mu_{i,j}[t] + \lambda^+_{i,j}\delta_{i,j} + \gamma^{lk}_{i,j}, & \{\text{j:}D_j<D_k\forall\text{k}\neq\text{j},o_i>\mu_{i,j}\} \\ \mu_{i,j}[t] + \lambda^-_{i,j}\delta_{i,j} + \gamma^{lk}_{i,j}, & \{\text{j:}D_j<D_k\forall\text{k}\neq\text{j},o_i<\mu_{i,j}\} \\ \mu_{i,j}[t] + \gamma^{lk}_{i,j}, & \text{otherwise.} \end{cases} \tag{23}$$

IV. System-Level Impacts: Modeling Inaccuracies

Given the inaccuracies discussed above, in this section we examine the impact of error sources on the algorithmic performance. This is accomplished by using computational models of the inaccuracies present in analog computation extracted from transistor-level simulations of the circuit. First, a metric for algorithmic performance must be defined. Since the common method for using DeSTIN in pattern recognition tasks is to extract the beliefs as features, performance is defined as the mean absolute error (MAE) between what the ideal belief values should be and those calculated considering errors in the system. It is important to note that this means performance is not directly tied to the numerical accuracy of the calculated centroid means and variances, which are being calculated in a space altered by the error sources. In the remainder of this section, we first explore the effects of the analog error sources on a synthetic dataset in order to demonstrate the effect on calculated belief states when the true centroids are known. Then, we present a nonsynthetic classification problem using the MNIST dataset in which classification performance is the best metric to evaluate the success of the system.

To demonstrate the effect of the various errors and mismatches, a simple clustering problem is considered. The data shown in Fig. 12 is clustered using a single DeSTIN node with varying levels of error and noise. The noise is always additive Gaussian noise, while gain errors and bias errors are implemented according to (25) and (25), respectively, where r_x is the difference between the maximum and minimum values x can take. Noise is added to select signals in the system in the same manner as bias error. The use of currents to represent variables generally leads to gain errors, while voltage-based signals lead to offset errors. We modeled the cases of full current-mode signaling and full voltage-mode

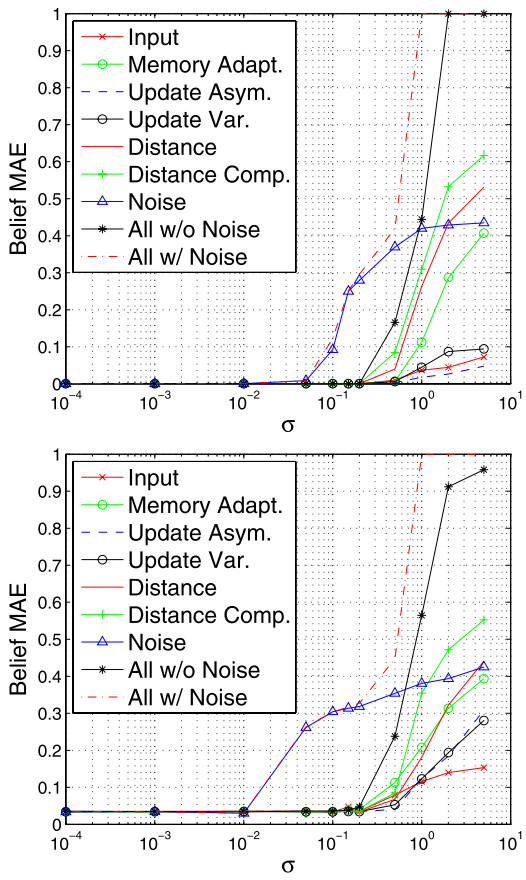

Fig. 13. Accuracy versus level of error (σ). Gain errors on clean dataset (top) and noisy dataset (bottom). This figure illustrates that the update and input errors have the lowest impact on performance, while noise has the most significant impact. In addition to the individual effects of each noise, this figure includes the effect of all the error sources combined and all the error sources and noise combined.

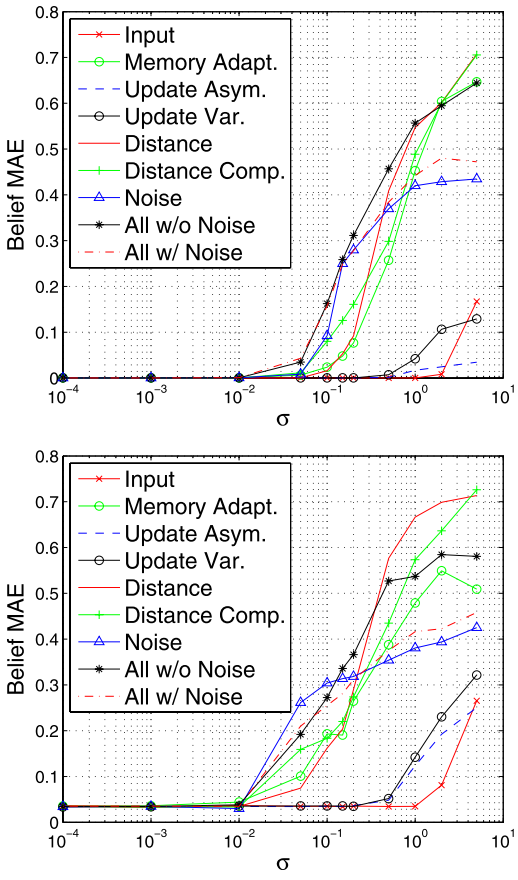

Fig. 14. Accuracy versus level of error (σ). Bias errors on clean dataset (top) and noisy dataset (bottom). The update and input errors have the lowest impact, while the remaining error components have an impact similar to that of the additive noise of the same level. In addition to the individual effects of each noise, this figure includes the effect of all the error sources combined and all the error sources and noise combined.

signaling to explore the different effects. The impact of gain and offset errors on a system using mixed-mode signaling can be expected to fall between these two cases

$$x' = x\mathcal{N}(1, \sigma) \tag{24}$$

$$x' = x + \mathcal{N}(0, \sigma)r_x. \tag{25}$$

The resulting error in the belief is calculated as the MAE between the ideal belief vector and that obtained using analog computation. Figs. 13 and 14 illustrate the effect of the errors discussed in this section on a single node's belief values. As can be observed, none of the errors introduces any notable degradation below a standard deviation value of 10^{-3}.

When modeling all errors as gain errors, additive noise in the system has a much larger impact than even the rest of the errors combined. Thus, it is demonstrated that the inconsistency caused by noise is much more harmful than the consistent gain and bias errors. The most destructive gain errors are the distance comparison, distance, and memory adaptation variation errors. The update asymmetry, input, and update variation errors have much less impact.

When all errors are modeled as bias errors, additive noise has much the same effect as the distance comparison, distance, and memory adaptation variation errors. The update asymmetry, input, and update variation errors still have much less impact.

It is important to remember, however, that "correct" clustering as defined by the metric used for these clustering datasets is not necessary for the DeSTIN hierarchy to produce meaningful beliefs. To test this ability, a full DeSTIN hierarchy is utilized to form features for a standard multilayer perceptron (MLP) neural network classifier. The DeSTIN hierarchy used contains three layers. The top, middle, and bottom layers contain 1, 4, and 16 nodes, respectively, with the nodes in each layer using 25, 18, and 25 centroids. The belief states are then sampled from three movements over the image. These beliefs are then provided to the classifier. Thus, we are testing the ability of our system which contains noise and errors to form a model of the regularities it observes and to produce features for a MLP classifier. The classifier used is a feed-forward neural network with two hidden layers with 128 neurons in the first hidden layer and 64 neurons the second hidden layer. The dataset used is the MNIST hand written digits dataset [33]. Results are shown in Fig. 15, where the robustness of the DeSTIN architecture to the gain errors, bias errors, and additive noise introduced by the analog computation

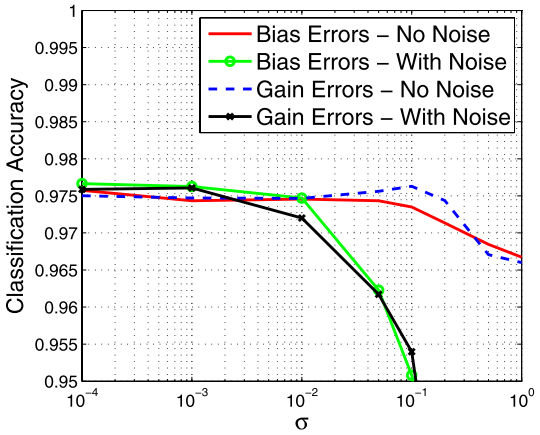

Fig. 15. MNIST performance versus system error/noise level. The results here demonstrate the effect of different error/noise levels when the analog error and noise sources are modeled in different ways. The best results here are 97.6%. However, when using a decorrelated ensemble of classifiers and using elastic distortions [34], results of 99% have been achieved with the DeSTIN architecture, which is more comparable to state of the art. These methods were not used to produce this plot, however, in order to produce results for such a wide range of variables in a timely manner.

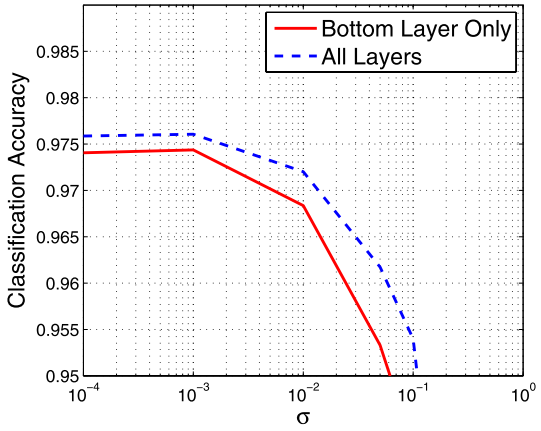

Fig. 16. MNIST performance versus error level. Providing the classifier with belief states from all layers is always better than only providing the belief states from the bottom layer for every error/noise level. These results are created with all errors modeled as gain errors and with additive noise.

models is illustrated. In particular, the errors have little to no effect until the standard deviation of the errors becomes larger than the operating range (0–1). High levels of noise result in inconsistent beliefs that hold less information for the classification module, as one would expect, but the system can be seen to be robust to significant levels of noise.

The results from the clustering performance tests and the MNIST classification test allow some important conclusions to be drawn. A significant amount of error and noise can be introduced to the DeSTIN architecture and its clustering algorithm without having a destructive effect upon performance. It is particularly noteworthy that noise is the most harmful error source by a significant margin. This is intuitively reasonable, as noise represents a dynamic error source to which the learning system cannot adapt. In contrast, the other error sources distort the input space in a static way, leaving relationships in the underlying data intact. Even noise does not significantly degrade MNIST performance for normalized levels below 10^{-2}, corresponding to an SNR of 40 dB. Fortunately, this modest level of noise performance is easily attainable even at extremely low current levels in the nanoampere range as shown in Fig. 11. It is important to consider such implications when designing analog circuits that retain performance while maintaining low power and area profiles.

Another aspect that should be better understood is how noise interacts with the depth of the architecture. It is imperative that global features generated at the higher layers, which are subject to errors introduced in the lower layers, remain meaningful. This is examined in Fig. 16, where classification results using only the bottom layer belief states are compared to classification results using the belief states from all layers. These results clearly demonstrate that features from the upper layer are able to add global information to the local information obtained from the lower layers, across all levels of error and noise.

V. Conclusion

This paper explored the promising implications of realizing deep layered machine learning architectures using custom analog VLSI. It is argued that the homogeneous nature of deep architectures, combined with the inherent simplicity in the computations involved, pave the way for scalable and efficient implementation of such systems using analog circuitry. While digital circuits have become ubiquitous over the last two decades in large part because of their simplicity and repeatability, analog circuits still retain advantages in the context of machine learning systems. Analog computation offers an improvement in power efficiency over equivalent digital systems, ranging from an order of magnitude for the simple x^2/y circuit described above to several orders of magnitude for more complex systems [15], [35]. Such dramatic reductions in power consumption offer the potential to deliver DML to severely power-constrained devices such as micro-scale wireless sensors and implantable medical devices. The analysis presented here demonstrates that the precision requirements of machine learning systems are compatible with ultralow-power analog circuits. In order to leverage the advantages of analog computation while maintaining excellent performance on learning tasks, it is critical that system designers understand the impact of various error sources on system performance. The results discussed in this paper can help guide designers so that the resources of power, chip area, and design effort can be focused on those areas where the impact is greatest.

References

[1] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," in *Proc. 12th Int. Conf. Artif. Intell. Stat.*, Apr. 2009, pp. 153–160.

[2] W. Bialek, F. Rieke, D. Ruyter, and D. Warland, "Reading a neural code," *Science*, vol. 252, no. 5014, pp. 1854–1857, Jun. 1991.

[3] R. R. de Ruyter van Steveninck, G. D. Lewen, S. P. Strong, R. Koberle, and W. Bialek, "Reproducibility and variability in neural spike trains," *Science*, vol. 275, no. 5307, pp. 1805–1809, Mar. 1997.

[4] M. van Rossum, B. J. O'Brien, B. J. O'brien, and R. G. Smith, "Effects of noise on the spike timing precision of retinal ganglion cells," *J. Neurophysiol.*, vol. 89, no. 5, pp. 2406–2419, May 2003.

[5] R. Sarpeshkar, "Analog versus digital: Extrapolating from electronics to neurobiology," *Neural Comput.*, vol. 10, no. 7, pp. 1601–1638, Oct. 1998.
[6] I. Arel, D. Rose, and R. Coop, "DeSTIN: A scalable deep learning architecture with application to high-dimensional robust pattern recognition," in *Proc. AAAI Fall Symp. BICA*, Nov. 2009, pp. 1150–1157.
[7] I. Arel, D. Rose, and T. Karnowski, "A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference," in *Proc. NIPS Workshop Deep Learn. Speech Recognit. Rel. Appl.*, Dec. 2009, pp. 1–8.
[8] Y. Bengio, A. C. Courville, and P. Vincent, *Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives*. Ithaca, NY, USA: Cornell Univ. Press, Jun. 2012.
[9] H. Larochelle, D. Erhan, A. C. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proc. 24th ICML*, 2007, pp. 473–480.
[10] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. 28th ICML*, Jun. 2011, pp. 689–696.
[11] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *Proc. 11th ISMIR*, Aug. 2010, pp. 339–344.
[12] J. Bergstra, F. Bastien, O. Breuleux, P. Lamblin, R. Pascanu, O. Delalleau, *et al.*, "Theano: Deep learning on gpus with python," in *Proc. Big Learning Workshop, NIPS*, vol. 11. 2011, pp. 1–6.
[13] S. Young, I. Arel, T. P. Karnowski, and D. Rose, "A fast and stable incremental clustering algorithm," in *Proc. 7th ITNG*, 2010, pp. 204–209.
[14] C. Twigg and P. Hasler, "Configurable analog signal processing," *Digit. Signal Process.*, vol. 19, no. 6, pp. 904–922, 2009.
[15] R. Chawla, A. Bandyopadhyay, V. Srinivasan, and P. Hasler, "A 531 nw/mhz, 128×32 current-mode programmable analog vector-matrix multiplier with over two decades of linearity," in *Proc. IEEE Custom Integr. Circuits Conf.*, Oct. 2004, pp. 651–654.
[16] P. Hasler and J. Dugger, "An analog floating-gate node for supervised learning," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 5, pp. 834–845, May 2005.
[17] J. Holleman, A. Mishra, C. Diorio, and B. Otis, "A micro-power neural spike detector and feature extractor in 0.13 μm CMOS," in *Proc. IEEE CICC*, Sep. 2008, pp. 333–336.
[18] R. Bhargava, H. Kargupta, and M. Powers, "Energy consumption in data analysis for on-board and distributed applications," in *Proc. Int. Conf. Mach. Learn. Workshop Mach. Learn. Technol. Autonom. Space Appl.*, vol. 3. 2003, pp. 1–6.
[19] T. Delbruck, "'Bump' circuits for computing similarity and dissimilarity of analog voltages," in *Proc. IJCNN*, vol. 1. Jul. 1991, pp. 475–479.
[20] P. Gray, P. Hurst, S. Lewis, and R. Meyer, *Analysis and Design of Analog Integrated Circuits*. New York, NY, USA: Wiley, 2001.
[21] T. Hall, C. Twigg, J. Gray, P. Hasler, and D. Anderson, "Large-scale field-programmable analog arrays for analog signal processing," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 11, pp. 2298–2307, Nov. 2005.
[22] F. Adil and P. Hasler, "Offset removal from floating gate differential amplifiers and mixers," in *Proc. 45th MWSCAS*, vol. 1. Aug. 2002, pp. 251–254.
[23] D. Hsu, M. Figueroa, and C. Diorio, "Competitive learning with floating-gate circuits," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 732–744, May 2002.
[24] P. Hasler and J. Dugger, "An analog floating-gate node for supervised learning," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 5, pp. 834–845, May 2005.
[25] J. Holleman, S. Bridges, B. Otis, and C. Diorio, "A 3 μw CMOS true random number generator with adaptive floating-gate offset cancellation," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1324–1336, May 2008.
[26] D. Hsu, S. Bridges, M. Figueroa, and C. Diorio, "Adaptive quantization and density estimation in silicon," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 15. 2003, pp. 1107–1114.
[27] M. Figueroa, S. Bridges, and C. Diorio, "On-chip compensation of device-mismatch effects in analog VLSI neural networks," in *Advances in Neural Information Processing Systems*, vol. 17, L. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA, USA: MIT Press, 2005, pp. 441–448.
[28] Y. Tsividis and C. McAndrew, *Operation and Modeling of the MOS Transistor*. London, U.K.: Oxford Univ. Press, 2011.
[29] S. Liu, J. Kramer, G. Indiveri, and T. Delbruck, *Analog VLSI: Circuits and Principles*. Cambridge, MA, USA: MIT Press, 2002.
[30] M. Pelgrom, A. Duinmaijer, and A. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1439, Oct. 1989.
[31] J. Holleman, "Micro-power integrated circuits for neural interfaces," Ph.D. dissertation, Dept. Electr. Eng., Univ. Washington, Seattle, WA, USA, 2009.
[32] P. Hasler, C. Diorio, B. Minch, and C. Mead, "Single transistor learning synapses," in *Advances in Neural Information Processing Systems*, vol. 7, G. Tesauro, D. Touretzky, and T. Leen, Eds. Cambridge, MA, USA: MIT Press, 1995, pp. 817–824.
[33] Y. LeCun and C. Cortes. (2010). *MNIST Handwritten Digit Database* [Online]. Available: http://yann.lecun.com/exdb/mnist/
[34] P. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, Aug. 2003, pp. 958–963.
[35] M. Figueroa, S. Bridges, D. Hsu, and C. Diorio, "A 19.2 GOPS mixed-signal filter with floating-gate adaptation," *IEEE J. Solid-State Circuits*, vol. 39, no. 7, pp. 1196–1201, Jul. 2004.

Steven Young (S'07) received the B.S. degree in electrical engineering from the University of Tennessee, Knoxville, TN, USA, in 2000, where he is currently pursuing the Ph.D. degree with the Machine Intelligence Laboratory.

His current research interests include scalable machine learning with a focus on deep learning algorithms and architectures.

Junjie Lu (S'12) received the B.Sc. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2007. He is currently pursuing the Ph.D. degree in electrical engineering from the University of Tennessee, Knoxville, TN, USA.

He was a Research Engineer with Philips, Shanghai, China, from 2007 to 2010. His current research interests include low-power, high-performance analog and mixed-signal circuit design, and analog signal processing.

Jeremy Holleman (S'02–M'10) received the bachelor's degree in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 1997, and the master's and Ph.D. degrees in electrical engineering from the University of Washington, Seattle, WA, USA, in 2006 and 2009, respectively.

He was with Data I/O, Redmond, WA, and National Semiconductor, Federal Way, WA. He is currently an Assistant Professor with the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN, USA. His current research interests include low-power integrated circuits for biomedical devices and other wireless sensing applications.

Itamar Arel (S'92–M'03–SM'06) received the B.S., M.S., Ph.D., and M.B.A. degrees in electrical and computer engineering from Ben-Gurion University, Beer-Sheva, Israel, in 1995, 1998, and 2003, respectively,.

He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN, USA, where he directs the Machine Intelligence Laboratory. From 2000 to 2003, he was with TeraCross, Inc., Campbell, CA, USA, where he was a Chief Scientist developing terabit/s switch fabric integrated circuits. His current research interests include large-scale deep machine learning and technologies for decision making under uncertainty.

Dr. Arel is a recipient of the U.S. Department of Energy Early CAREER Award.