Spatial-Temporal Hybrid Neural Network With Computing-in-Memory Architecture

Kangjun Bai, *Student Member, IEEE*, Lingjia Liu, *Senior Member, IEEE*, and Yang Yi, *Senior Member, IEEE*

***Abstract*— Deep learning (DL) has gained unprecedented success in many real-world applications. However, DL poses difficulties for efficient hardware implementation due to the needs of a complex gradient-based learning algorithm and the required high memory bandwidth for synaptic weight storage, especially in today's data-intensive environment. Computing-in-memory (CIM) strategies have emerged as an alternative for realizing energy-efficient neuromorphic applications in silicon, reducing resources and energy required for neural computations. In this work, we exploit a CIM-based spatial-temporal hybrid neural network (STHNN) with a unique learning algorithm. To be specific, we integrate both multilayer perceptron and recurrent-based delay-dynamical system, making the network becomes linear separable while processing information in both spatial and temporal domains, better yet, reducing the memory bandwidth and hardware overhead through the CIM architecture. The prototype fabricated in 180nm CMOS process is built of fully-analog components, yielding an average on-chip classification accuracy up to 86.9% on handprinted alphabet characters with a power consumption of 33mW. Beyond that, through the handwritten digit database and the radio frequency fingerprinting dataset, software-based numerical evaluations offer 1.6–to–9.8× and 1.9–to–4.4× speedup, respectively, without significantly degrading its classification accuracy compared to the cutting-edge DL approaches.**

***Index Terms*— Deep learning, computing-in-memory, spatial-temporal architecture, analog computing, delay-dynamical system, hybrid neural network, on-chip classification.**

I. Introduction

ARTIFICIAL intelligence (AI), an automatic learning system intended to replicate the way that we humans learn, offers an alternative solution to accelerate the computational efficiency with witnessed remarkable progress in a board spectrum of scenarios [1]. The capabilities of AI are exploited through software simulations, which rely on general-purpose computing systems or cloud infrastructures to train a large-scale artificial neural network (ANN). As their drawbacks, the required computational resources (*e.g.*, the storage capacity and the memory bandwidth) and the network performance (*e.g.*, the connectivity and the latency of network communication) significantly degrade the user experience, while themselves exposing other security issues.

Throughout the development history of AI, deep neural networks (DNNs) have been optimized by the gradient-based learning algorithms with the help of high-performance processors, programming frameworks, and big data [2]–[4]. The high level of complexity in DNNs requires the use of high-performance central processing units (CPUs) and graphical processing units (GPUs) to execute neural computations. However, the fact that separating the memory and CPUs/GPUs location requires more resources to be allocated for data transmission, resulting in lower computational efficiency. With the increasing demand in DNNs, the demand for sophisticated hardware has also increased, boosting the power consumption to a higher magnitude.

Application-specific integrated circuit (ASIC) chips, on the other hand, have paved the way for DNNs by accommodating electronic synapses and neural computing strategies with parallel computing capability. By storing the weight matrices (*i.e.*, memory) in electronic synapses between layers, the multiplication-and-accumulation (MAC) operation is made possible by reading the memory information directly with multiple inputs in a single run, reducing the memory bandwidth required for read/write operations and increasing the computational efficiency. Many highly optimal DNNs have been introduced to demonstrate remarkable computing capabilities in a wide range of complex applications [2], [5]–[7], and yet, these structures suffer from the computational cost, leading to inefficient hardware implementation.

On one side, ASIC implementations significantly accelerate the computational efficiency of DNNs. On the other hand, DNNs tend to maximize the accuracy with increments of complexity [8]. In particular, the use of gradient-based backpropagation learning algorithm is a highly complex operation, as the gradient of loss functions needs to be propagated recursively backward to estimate the change of weights required in each hidden layer. More importantly, the complications are amplified by the imperfections in circuit components.

By contrast, the introduction of reservoir computing networks has opened up new possibilities by eschewing the complex gradient-based learning algorithms [9], [10]. Reservoir computing networks, uniquely suited for nonlinear information processing, offer a unique training mechanism by only updating the output weights, significantly reducing the

Manuscript received October 16, 2020; revised February 17, 2021 and March 24, 2021; accepted April 2, 2021. Date of publication April 16, 2021; date of current version June 4, 2021. This work was supported in part by the U.S. National Science Foundation (NSF) under Grant CCF-1750450, Grant ECCS-1811497, and Grant CCF-1937487. This article was recommended by Associate Editor A. Worapishet. *(Corresponding author: Kangjun Bai.)*

The authors are with The Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: kangjun@vt.edu; ljliu@vt.edu; yangyi8@vt.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2021.3071956.

Digital Object Identifier 10.1109/TCSI.2021.3071956

computational and design overhead. In recent years, various applications of reservoir computing networks have been introduced; in particular, reservoir computing networks have been shown to be a promising candidate for sequence learning [11]–[13]. Beyond that, the delay-based reservoir network is a unique reservoir computing paradigm with a delay-feedback structure [14], which satisfies the required high-dimensional mapping as in the original reservoir computing network with a simple processing structure, making the network an outstanding model for power-limited portable devices.

The motivation of our work is to introduce a computing-in-memory (CIM)-based spatial-temporal hybrid neural network (STHNN) with an embedded delay-dynamical system. Specifically, we focus on the optimization of network structure and training algorithm to minimize the implementation complexity of ASIC-based DNN designs. Major contributions of our work are summarized as follows:

1) A spatial-temporal computing structure by integrating both multilayer perceptron and recurrent-based delay-dynamical system.
2) A CIM-based MAC operator with fully-analog components, enabling the high-speed parallel operation without degrading the network's stability.
3) A unique training mechanism by only updating readout weights, potentially reducing the resources and energy required for learning operations in hardware.
4) A prototype fabrication with microcontroller verification, yielding an average on-chip classification accuracy up to 86.9% on handprinted alphabet characters with merely 33mW of power consumption.
5) Software-based evaluations offer up to 9.8× speedup over the cutting-edge DNN models while yielding a competitive classification accuracy.

The rest of this paper is organized as follows: Section II provides an overview of DNNs. The design methodology of the introduced CIM-based STHNN and the demonstration of the fabricated prototype are discussed in Section III and Section IV, respectively, followed by the software-based evaluations in Section V. The paper is then concluded in Section VI.

II. Deep Learning

Deep learning (DL), the cutting-edge of AI, is capable to automatically learn and optimize by extracting key features or finding similarities from data based on a general-propose learning algorithm [15], [16]. This is made possible through multiple hidden layers and a colossal amount of neurons in DNNs, in a similar way to humans; that is, DNNs are what underpins DL. For instance, in a DNN-based image classification task, edges at a particular orientation or location of an image are extracted by the first hidden layer, while particular arrangements of edges and motifs are detected by latter hidden layers [17]. Essentially, by deploying more hidden layers and associated neurons, DNNs can represent functions of increasing complexity, exploring themselves to more sophisticated machine learning applications.

A. Deep Learning Models

The general development of DNNs can be categorized into two aspects, *i.e.*, the feedforward neural networks (FNNs), representing the spatial-based DNN structures, and the recurrent neural networks (RNNs), representing the temporal-based DNN structures. The former targets at extracting key features from static data while the latter aims at finding similarities from dynamic information.

In FNNs, multilayer perceptron (MLP) is only made of dense layers, while convolutional neural networks (CNNs) are made of convolutional layers, pooling layers, and dense layers. By concatenating multiple hidden layers in a processing pipeline, both MLP and CNNs have become quintessential DNN models used in computer vision and pattern recognition [5], [18]–[20]. Despite that CNNs have broken many performance records in computer vision, the number of convolution kernels and layers significantly impacts the training time and the inference accuracy. The recent progress in DL aims to bridge the gap between neuroscience and machine learning, creating a network that closely replicates the behavior of brain cortex. Spiking neural networks (SNNs) incorporate event-driven processing structure, in which neurons propagate information through biologically-realistic signals (*i.e.*, spikes), exhibiting favorable properties in neural circuits like the brain [21]. SNNs have been proven to be a powerful tool in domains of classification and recognition tasks [22]–[24].

By contrast, in RNNs, hidden layers are built of feedback connections, having similar temporal dynamics as in SNNs [25]. In general, RNNs are designed to take a series of input without predetermined limit on size, adding an interesting twist over FNNs. With the recurrent structure, RNNs have the capability to process one or more input vectors and to produce one or more output vectors, whose outputs are influenced not only by internal weights but also by the context based on historical information. Among various RNN models, both long short-term memory (LSTM) [26] and gated recurrent unit (GRU) [27] have additional interacting gates to determine how much of each information should be removed or passed forward, yielding a better control-ability and performance. Due to the sequential nature of recurrent connections, RNNs have widely deployed in time series prediction, speech recognition, and natural language processing [28]–[31]. Nevertheless, the training operation with historical information presented in earlier computing cycles has become the major factor that limits the efficient hardware implementation. More importantly, all internal weight matrices and bias vectors need to be trained, leading to significant computational and hardware overhead, and impeding such powerful computing modules for deployment to portable devices.

B. Learning Algorithms and Design Challenges

In general, both FNNs and RNNs utilize some version of gradient descent for calculating weight updates during the training, that is, the bigger the gradient, the bigger the adjustment, or vice versa. However, this method is complicated by the vanishing gradient problem [32], which is inevitable in gradient-based learning algorithms. In such approaches,

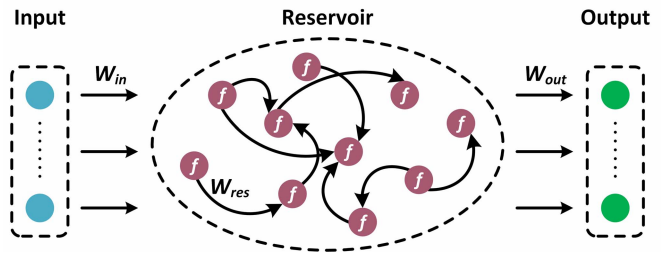


Fig. 1. General structure of echo state network (ESN).

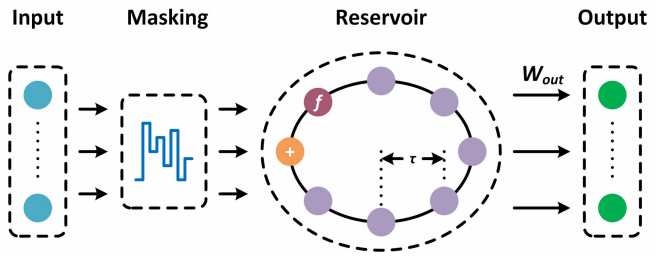


Fig. 2. General structure of time-delay reservoir (TDR) network.

gradients of error function will gradually shrink as it propagates recursively backward. As the network gets deeper, the gradient presented in earlier hidden layers will be plainly small or even non-existent, preventing the weight from changing its value or even completely stopping the neural network from training. On the other hand, by directly accessing the neural activation in the forgetting gate with a unique additive gradient structure, LSTMs enable the network to prevent the error gradients from vanishing [33]. Despite that both CNNs and LSTMs with the use of gradient-based learning algorithms are powerful, such structures are complicated to be realized in hardware.

C. Reservoir Computing Network

Echo State Networks (ESNs) [9], the well-known reservoir computing paradigm, sidestep this issue by only training the output weights with a unique learning mechanism. As depicted in Fig. 1, ESNs consist of 3 major computing layers, in which the reservoir layer is formed by a group of sparsely-connected neurons, offering a high-dimensional mapping for sequential inputs and enhancing the separability of networks. The general state of reservoir dynamics can be expressed as

$$h_t = f(x_t \cdot W_{in} + h_{t-1} \cdot W_h + b_h), \tag{1}$$

where $f()$ is the hyperbolic tangent ($tanh$) activation, x_t denotes the input at present computing cycle, W_{in} and W_{res} represent input weights and internal weights, respectively, h_{t-1} indicates the historical information from the previous computing cycle, and b_h is bias vectors. The actual output can be then calculated as

$$\hat{y}_t = h_t \cdot W_{out} + b_{out}, \tag{2}$$

where W_{out} indicates output weights. In practice, the reservoir layer must satisfy the echo state property (ESP) for computing the ESN principle, relating asymptotic properties of reservoir dynamics to the driving signal [34]. For such an operation, W_{res} is first initialized with a uniform distribution between -0.5 to 0.5. This weight matrix is then updated only once according to the ESP with a spectral radius below 1, which can be denoted as

$$W_{res} := \frac{|\lambda_{max}(W_{res})|}{\sigma} \cdot W_{res}, \tag{3}$$

where $|\lambda_{max}(W_{res})|$ is an eigenvalue of $W_{res} \in [-0.5, 0.5]$ with the largest absolute value, and σ represents the spectral radius. Once the network is initialized, W_{in} and W_{res} remain fixed for the entire simulation.

In the recent progress of reservoir computing, a delay-based reservoir computing is proposed with the introduction of time-multiplexing, as shown in Fig. 2. The reservoir dynamics are hereby given by a delay-feedback system with a single nonlinear neural activation, which can be simplified as

$$h_t = f(x_t \cdot MK + h_{t-1}), \tag{4}$$

where MK is a mask function, in which the time length per frame is identical to the time interval between neurons in the reservoir layer. The use of delay-feedback system enables the simple implementation of neural networks with analogue hardware in either electronically or optically [14], [35]–[38].

III. Spatial-Temporal Hybrid Neural Network

Hybrid neural networks (HNNs) typically concatenate two or more FNNs and RNNs in a processing pipeline to improve the learning capability. In this work, we build a CIM-based spatial-temporal HNN (STHNN) by integrating both MLP and recurrent-based delay-dynamical system (DDS), making the network becomes linear separable while computing static and dynamic data in both spatial and temporal domains. Fig. 3 demonstrates the general architecture of our CIM-based STHNN, composing of a dense layer as the feature extractor, multiple analog computing units that are built upon DDS as a internal processing layer, and a CIM-based neural classifier as the output layer. More specifically, we utilize the spatial characteristic in the dense layer to extract key features of inputs, while utilizing the temporal characteristic in the DDS to enable the learning capability with historical information. In the meantime, we take advantage of the unique learning mechanism introduced in the reservoir computing network to reduce required resources for training. Lastly, we adopt the spiking information processing capability to diminish the implementation complexity on ASIC chip.

A. CIM-Based MAC Operation

The necessary feature extraction of DL through the MAC operation is commonly carried out through an intensive sum-of-product computation. However, due to the high access latency and bandwidth required for reading/writing memory cells (MCs), a timely response can no longer be realized with the conventional computing architecture [39]. On the other hand, in neuromorphic applications, crossbars with current-based MCs can frankly implement such a MAC operation. Essentially, input vectors can be mapped to voltages and weight matrices can be implemented with crossbars. Consequently, the MAC results can be computed by multiplying input vectors with MCs in crossbars and sampling

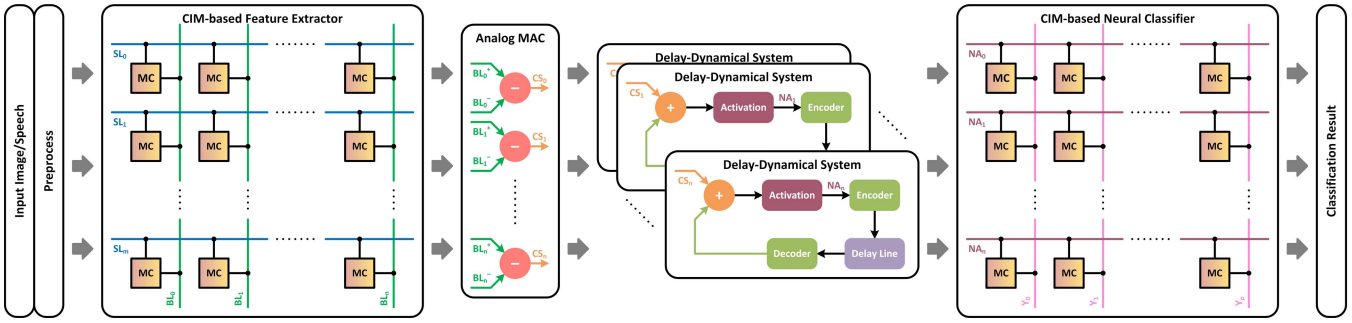


Fig. 3. Overview of computing-in-memory (CIM)-based spatial-temporal hybrid neural network (STHNN) architecture.

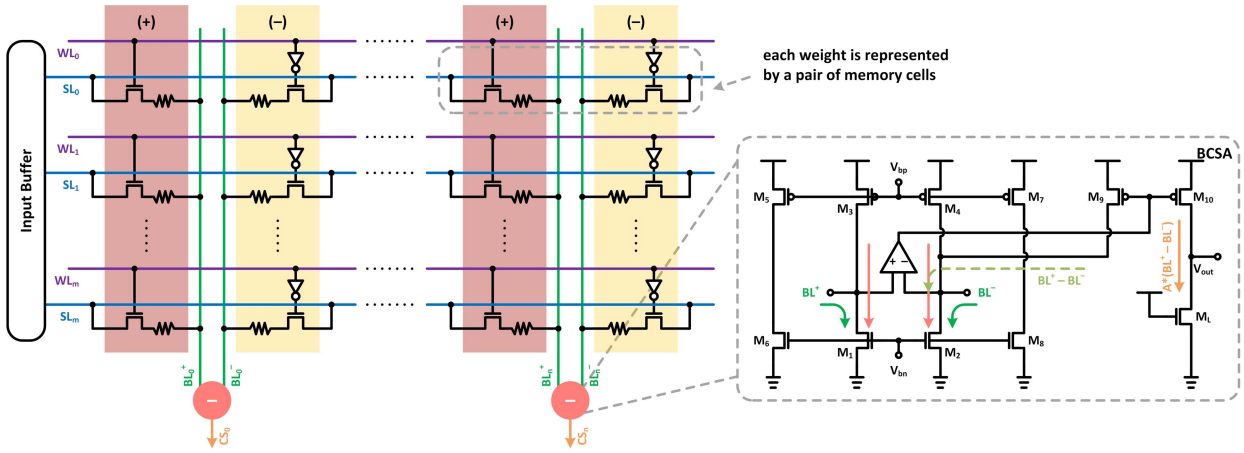


Fig. 4. Deploying the computing-in-memory (CIM)-based multiplication-and-accumulation (MAC) operator on a resistive crossbar.

the accumulated current on each bit-line (BL), which can be expressed as

$$I_j = \sum_{i=1}^{m} V_i \cdot G_{ij}, \tag{5}$$

where V_i is the input voltage, m defines the input dimension, and G_{ij} represents the conductance of an MC located at the i-th source-line (SL) and j-th BL.

The deployment of MAC operation on a memory crossbar is depicted in Fig. 4. In practice, any given patterns (*e.g.*, images) are commonly reshaped into an one-dimensional (1D) matrix, $[1, m]$, before the neural computation takes over. To extract m_f features, a crossbar with the shape of $[m, m_f]$ is deployed. With the consideration of precise modeling, both positive and negative weights are utilized in our work to form a double-column crossbar, in which a single weighted value of W_{in} is represented by a pair of MCs with a shared SL, that is, $G_{ij} = G_{ij}^{+} - G_{ij}^{-}$, where G_{ij}^{+} and G_{ij}^{-} denote the positive and negative conductance of MCs, respectively.

The MCs used in this design are built of resistive memory, in which the high-resistance-state (HRS) and the low-resistance-state (LRS) of each MC are defined as $R \in$ [1k–to–20kΩ]. This particular resistance state is allocated within the range of measured resistance states through the discrete memristor device (BS-AF-W) from the Known Inc. [40] with a linear scaling factor of 50,000. Table I demonstrates a simplified truth table of MAC operation with a binary input, representing a black-and-white image. In such a computation,

TABLE I

SIMPLIFIED TRUTH TABLE OF MULTIPLYING-AND-ACCUMULATING (MAC) OPERATION WITH BINARY INPUT

Input (1-bit)	(+) Weight	(-) Weight	MAC Out
0	HRS	LRS	0
0	LRS	HRS	0
1	HRS	LRS	$I_{high}@BL^{-}$
1	LRS	HRS	$I_{high}@BL^{+}$

a group of voltages, representing the corresponding information of an actual input, is applied to the horizontal SLs synchronously. As the outcome, an accumulated current is generated at each vertical BL where the MAC operation is deployed, which can be expressed as

$$I_j = I_j^{+} - I_j^{-} = \sum_{i=1}^{m} V_i \cdot G_{ij}^{+} - \sum_{i=1}^{m} V_i \cdot G_{ij}^{-}. \tag{6}$$

It can be observed that a subtraction operation is required for the double-column crossbar design as the negative conductance nor current are non-existent. To support such an operation, a bilateral current sensing amplifier (BCSA) with the inlaid neural activation is deployed. During the operation, the accumulated I_j^{+} and I_j^{-} are respectively injected into the BSCA. The transistor M_1 senses the positive input current, I_j^{+}, and the reference current generated through the transistor M_3, such that $I_{M1} = I_j^{+} + I_{M3}$. This current is then replicated through the associated current mirror, M_5–to–M_8, such that $I_{M8} = I_{M6} = I_{M1}$. As the negative input current, I_j^{-}, injects

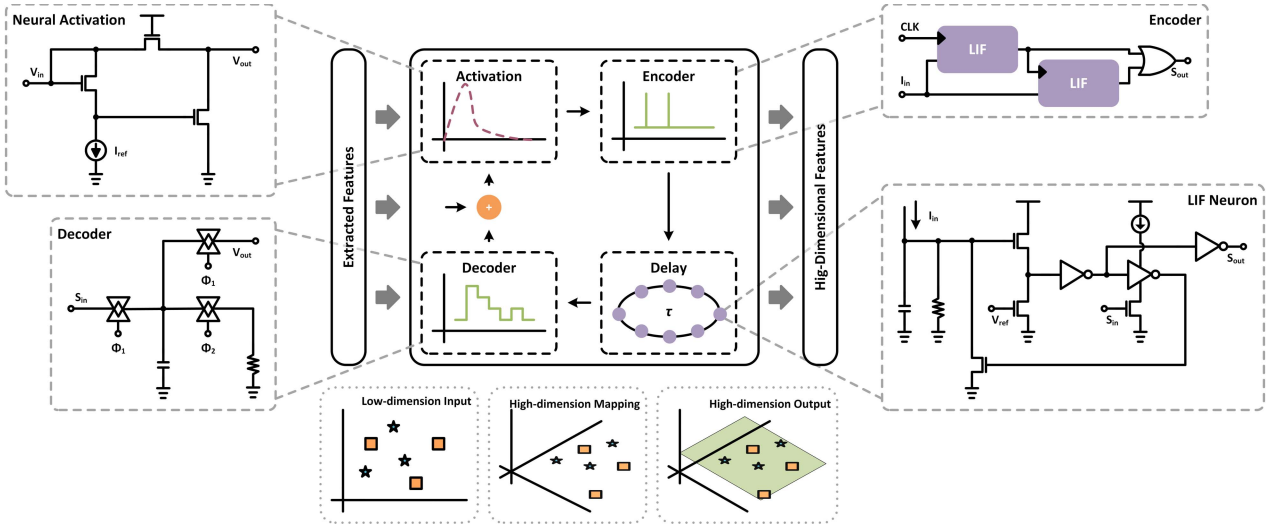


Fig. 5. Computing structure of delay-dynamical system (DDS).

into the transistor M_2, the associated current mirror forces M_2 to replicate the current at M_1, and thus, $I_{M2} = I_{M1}$. By balancing the current of I_{M1} and I_{M2}, the current through the transistor M_9 can be expressed as $I_{M9} = I_j^+ - I_j^-$, such that, $I_{M2} = I_{M9} + I_j^- + I_{M4} = I_j^+ + I_{M4} = I_{M1}$. The high-gain operational amplifier keeps tracking the variation of its positive and negative inputs, and dynamically regulates the driving voltage of M_9. The resulting current of $I_j^+ - I_j^-$ is replicated through the output current mirror, M_9 & M_{10}, with a current gain of A. The output voltage can be then consistently generated through a loading transistor M_L. Such a linear computation is obtained when $I_j^+ - I_j^- > 0$. By contrast, $I_{M9} = 0$ when $I_j^+ - I_j^- < 0$ as the current cannot be propagated reversely through the transistor M_9, and thus, output remains at 0V.

Such a CIM-based network allows MAC operation with multiple inputs in a single reading to enable the high-speed parallel operation, minimizes the output voltage variation under various BL currents, models the neural activation of the rectified linear unit (ReLU) as required in the feature extractor, and isolates the MAC operation and latter computations. More importantly, the MAC operation along with the BCSA generate the desired analog signal as required by DDS, and thus, analog-to-digital and digital-to-analog converters are unnecessary, potentially making the system suitable for large-scale DNN designs.

B. Delay-Dynamical System

In our STHNN, the internal processing layer is built of DDS, allowing the network to gather knowledge and experience from its previous computing cycle without the use of synaptic weights. The DDS is built upon a delay-based reservoir computing network, as demonstrated in Fig. 5, composing of a nonlinear neural activation (NNA), an analog-to-spike (A/S) encoder, a dynamic memory delay line (MDL), and a spike-to-analog (S/A) decoder.

During the operation, BCSA fetches the accumulated MAC results for the NNA to carry out the high-dimensional mapping, and thus, uncorrelated features between inputs can

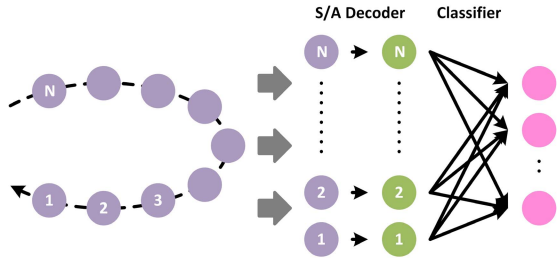


Fig. 6. Operating principle of the dynamic memory delay line (MDL) with dense output weights and neural classifier.

be linearly separated. The selected activation data is then encoded into a temporal spike train and propagated along the MDL with a controllable timing coefficient to express the historical information, that is, forming the short-term memory. As depicted in Fig. 6, the MDL is generally composed of a succession of leaky integrate-and-fire (LIF) neurons, carrying out the linear feedback shifting operation and storing the shifted information for a short period of time, and thus, additional memory storage is unnecessary. Through such a delay-feedback structure, the DDS exhibits a transient response to form a short-term memory, gaining the capability to learn from the context of data. As the outcomes from the DDS, a set of temporal spike trains stored in the MDL, representing the internal state of DDS, are gathered and decoded back into analog signals to compute the final outcomes through the neural classifier.

NNA is the key component in DNNs to carry out the high-dimensional mapping; however, the commonly used NNAs in RNNs (*e.g.*, *sigmoid* and *tanh*) are still suffered from the vanishing gradient problem, in which the gradient towards either end of these functions tends to respond very less to change. By natural, Mackey-Glass (MG) function [41] refers to a family of delay differential equation, showing that the dynamics of a delay-feedback system can depend on both present inputs and historical data, which can be written as

$$y_t = \alpha \cdot \frac{x_{t-\tau}}{1 + x_{t-\tau}^{\beta}} - \gamma \cdot x_t, \tag{7}$$

Algorithm 1 STHNN

Data: $x = [n, m]$, $y = [n, l]$, $scaling = k$, $G.noise$
Result: W_{out}
initialization
$W_{in} := \frac{|\lambda_{max}(W_{in})|}{\sigma} \cdot W_{in}$
for $i \leftarrow 1$ **to** n **do**
  $map_t = ReLU(m_i \cdot W_{in} + b_{in})$
  for $ii \leftarrow 1$ **to** m **do**
    $h_t = MG((1-k) \cdot map_t + k \cdot h_{t-1} + b_h)$
  end
  if $G.noise$ *is not None* **then**
    $h_t = h_t + G.noise$
  end
end
return h_t
$S = [h_0, h_1, \cdots, h_n]$
$Y = [y_0, y_1, \cdots, y_n]$
$W_{out} = Y \cdot S' \cdot (S \cdot S' + \eta \cdot I)^{-1}$

where x_t is the input at present computing cycle, $x_{t-\tau}$ represents the input from τ-th previous computing cycle, α and γ are scaling parameters where γ is commonly defined as $\gamma = 1 - \alpha$, and β is the nonlinear exponent. The use of MG function as NNA was first introduced in [14]; afterward, our previous work in [13] demonstrates the advantages of MG over classical NNA in the application of wireless communication. In the circuit point of view, the characteristic of MG function can be realized by comparing the potential level of input signal and the threshold voltage of sensing transistor. In this work, the electronic circuit model of MG function is optimized based on our previous design in [13].

Likewise, the LIF neuron is an essential processing unit in our STHNN, which is built based upon our previous design in [35]. Such a LIF neuron can be adopted to implement the A/S encoder by directly sensing the raw sensory information and to implement the MDL by inputting a constant current with a controllable triggering signal. The general operation is carried out by firing a spike once the potential across the membrane capacitor exceeds the firing threshold. Ultimately, to compute the final outcomes of network, a S/A decoder is needed to decode the temporal spike train back into analog signals, in which the decoder is optimized based on our previous work in [42]. As the encoded information is carried by time intervals on a temporal spike train, the core idea of the decoding scheme is to extract these time intervals as a driving signal to achieve various levels of analog voltage.

C. Learning Rule

In the training operation, weight matrices of the feature extractor are initialized according to the ESP with a uniform distribution between -1 to 1. Once the input layer is initialized, W_{in} remains fixed for the entire computation. Such an operation allows the input layer to act as a unique filter for feature extraction.

For each computing cycle, the trajectory of reservoir dynamics is computed by feeding the training sample, $\{x_i\}_{i=1}^{m}$. As the outcomes, a set of internal states, $\{h_t\}_{t=1}^{m}$, is obtained by integrating the present feature input and the historical information stored in the MDL. Consequently, optimal output weights can be calculated directly through the Tikhonov regularization, which can be defined as

$$W_{out} = Y \cdot S' \cdot (S \cdot S' + \eta \cdot I)^{-1}, \tag{8}$$

where S' represents the transpose matrix of reservoir dynamics $S = [h_0, h_1, \cdots, h_n]$, $Y = [y_0, y_1, \cdots, y_n]$ is the target output, n is the number of training samples, $\eta \geq 0$ is a constant, and I is the identity matrix with the same size of reservoir state. Classification accuracy is then calculated simply as the fraction of correctly classified inputs. The general learning operation of our STHNN is summarized in Algorithm 1.

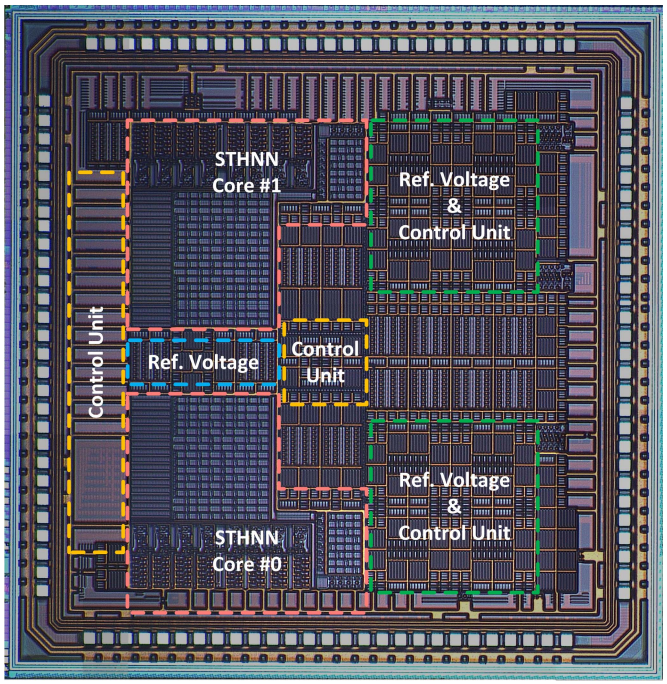


Fig. 7. Die micrograph of fabricated prototype in standard 180nm CMOS process, occupying 9mm^2 die area.

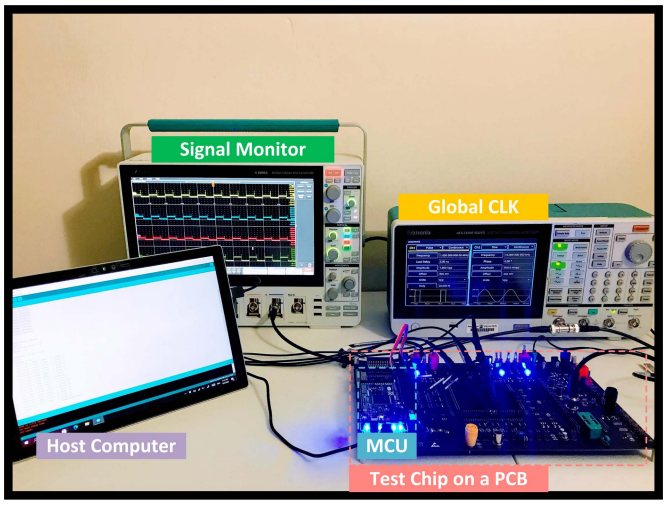


Fig. 8. System-level testbench for performance evaluations.

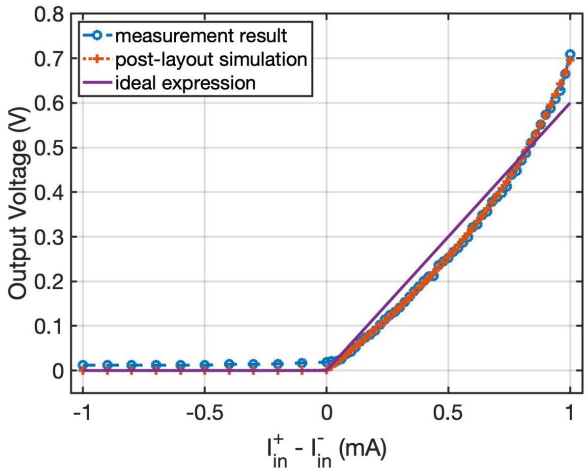


Fig. 9. Measured characteristics of bilateral current sensing amplifier (BCSA) with inlaid rectified linear unit (ReLU) neural activation.

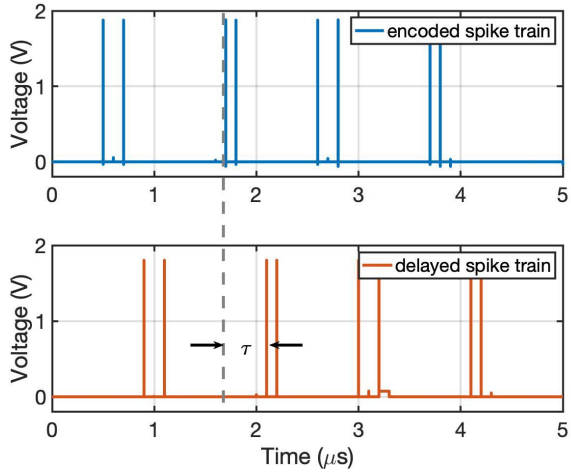


Fig. 11. Illustration of temporal spike trains along the memory delay line (MDL) with an identical delay time of τ.

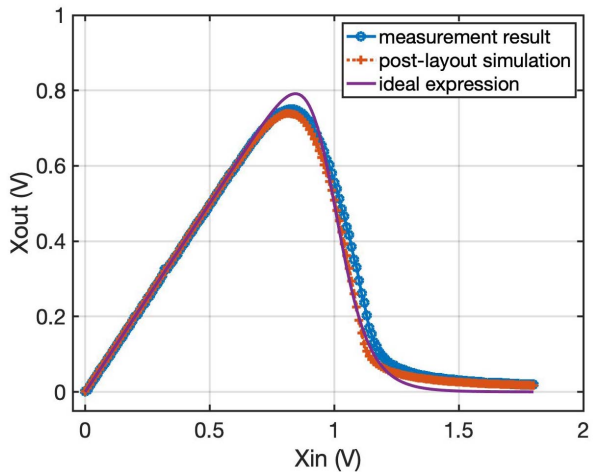


Fig. 10. Measured characteristics of Mackey-Glass (MG) neural activation together with the ideal fit.

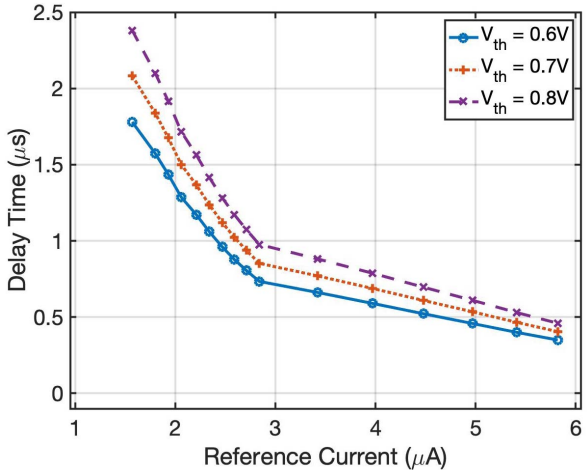


Fig. 12. Measured characteristic of leaky integrate-and-fire (LIF) neuron with respect to various threshold voltages and reference currents.

IV. Performance Evaluation

A prototype of STHNN is fabricated in GlobalFoundries (GF) standard 180nm CMOS process for basic function verification of on-chip classification. Fig. 7 demonstrates the die micrograph of our fabricated STHNN. Two STHNN cores along with peripheries and basic function modules measure 9mm^2, where each STHNN core occupies 0.814mm^2. In this prototype, each STHNN core is built of a 16×8 CIM-based feature extractor with analog MAC operators, $8\times$ DDSs with a total of 64 neurons, and a 8×4 CIM-based neural classifier. During the inference, input weights and output weights are initialized and calculated, respectively, through an offline compiler. Fig. 8 presents the system-level testbench and software interface for the performance evaluation, composing of an ARM-based microcontroller (MCU), a test chip mounted on the printed circuit board (PCB), and a host computer.

A. System Characteristics

1) MAC: With the introduced BCSA, the linearity and the stability between the MAC operation and latter computations can be maintained. To demonstrate such a functionality, input currents, I_{in}^{+} and I_{in}^{-}, respectively collected from BL^{+} and BL^{-} from the crossbar were directly applied to the BCSA. As plotted in Fig. 9, a linear response of output voltage, in a range of 0–to–0.7V, can be obtained when $I_{in}^{+} - I_{in}^{-} > 0$, in which the dynamic range of both I_{in}^{+} and I_{in}^{-} were set to be 0–to–1mA. By contrast, V_{out} remains at 0V when $I_{in}^{+} - I_{in}^{-} < 0$, indicating that more features are correlated with negative weights. Such a distribution closely models the MAC operation along with the ReLU neural activation as required by the feature extractor. Compared to the design in [43], our BCSA is capable to process information collected from both positive and negative synaptic weights with $2\times$ input dynamic range, and naturally achieve a ReLU NNA.

2) NNA: From Eq. (7), the nonlinear characteristic of the MG NNA can be denoted as

$$X_{out} = \alpha \cdot \frac{X_{in}}{1 + X_{in}^{\beta}}. \tag{9}$$

As shown in Fig. 10, a nonlinear response of output voltage can be observed, in which the input voltage was set to be

TABLE II
MEASURED ON-CHIP CLASSIFICATION ACCURACY OF 12 SELECTED CHARACTERS DEPLOYED IN 9 TESTING SETS

		Test 1				Test 2				Test 3			
Character		A	C	I	N	A	F	N	T	A	J	N	X
Accuracy	w/o noise	94.5%	93.0%	95.3%	96.9%	53.1%	46.9%	97.7%	96.9%	90.6%	94.5%	95.3%	93.8%
	w. noise	82.0%	78.9%	79.7%	81.3%	46.1%	40.6%	82.8%	79.7%	76.6%	79.7%	83.5%	82.0%

		Test 4				Test 5				Test 6			
Character		C	F	T	U	C	N	O	P	F	I	J	N
Accuracy	w/o noise	96.9%	96.1%	92.2%	90.6%	93.0%	92.2%	94.5%	93.0%	96.9%	95.3%	95.3%	97.9%
	w. noise	82.8%	84.4%	80.5%	78.1%	79.7%	79.7%	81.3%	82.0%	81.3%	76.6%	82.8%	84.4%

		Test 7				Test 8				Test 9			
Character		F	N	P	X	H	N	U	X	J	N	P	X
Accuracy	w/o noise	48.4%	97.7%	45.3%	96.9%	46.9%	53.1%	90.6%	95.3%	93.8%	92.2%	91.4%	94.5%
	w. noise	39.1%	81.3%	40.6%	79.7%	37.5%	41.4%	76.7%	82.8%	82.0%	81.3%	76.6%	76.6%

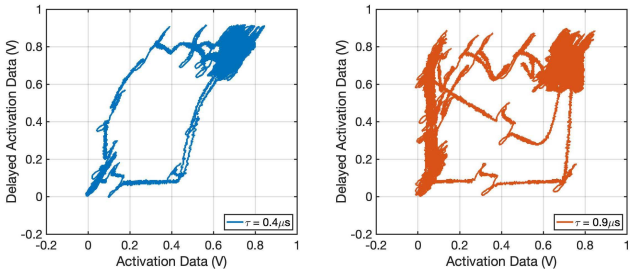


Fig. 13. Measured reservoir dynamics with (*left*) a short delay of 0.4μs and (*right*) a long delay of 0.9μs.

0–to–1.8V. It can be also observed that the measured nonlinear characteristic fits the ideal MG function with a scaling parameter and nonlinear exponent of $\alpha = 1$ and $\beta = 16$, respectively. Moreover, the reported power consumption of 24.55μW in our newly optimal design exhibits 2.66× power reduction over our previous implementation in [13].

3) MDL: Multiple LIF neurons are concatenated in series to form a MDL, where the spike generated from the present neuron is used as the triggering signal for its following. The delay is formed based on the time required to fire the LIF neuron, which can be simplified as,

$$\tau = C_m \cdot \frac{V_{cm}}{I_{ref}}, \tag{10}$$

where C_m is the membrane capacitance, V_{cm} is the voltage potential across C_m, and I_{ref} is the reference input current. As demonstrated in Fig. 11, a delayed temporal spike train along the MDL can be obtained with an identical delay time. Moreover, by controlling the reference current applied to the LIF neuron, an adjustable delay, in a range of 0.35–to–2.38μs, can be obtained, as depicted in Fig. 12. Likewise, compared to the LIF neuron presented in our previous work [35], the reported power consumption of 3.1μW in our newly optimal design expresses 5.37× power reduction even with a less advanced CMOS process and a higher supply voltage.

4) Reservoir Dynamic: In the Lyapunov stability analysis, a delay-feedback system can be either stabilized or destabilized with the introduction of delay, transferring the dynamic of such a system from periodic to chaotic, or vice versa [44]. The phase portraits, as illustrated in Figure 13, depict the dynamics of DDS by using two activation data, in which one of them is collected with time delay. When the timing coefficient is set to be 0.4μs, the delayed activation data keeps tracing its initial path while only a small amount of data is diverged, making the dynamic of DDS close to the periodic regime. As the timing coefficient increases (e.g., 0.9μs), more delayed activation data diverge from the initial path, and yet, the equilibrium point remains to track its original path even in a long run, making the dynamic of DDS close to the chaotic regime.

B. On-Chip Character Classification

The neural classifier in our fabricated STHNN prototype contains four output neurons, which are capable to classify images into four categories. In this experiment, 12 capital characters were drawn from the NIST handprinted alphabet character database [51]. Each MC in the feature extractor and neural classifier were set according to the optimal weights after training a small-scale network through software. During the inference, selected images (128×128 pixels) were cropped (32×32 pixels), down-sampled (4×4 pixels), and reshaped into a 1D column vectors through the MCU. The final outcomes from the MCU were further scaled down to the desired voltage level supported by the fabricated test chip (*e.g.*, 0–to–1.8V) through off-chip voltage dividers and buffers. Input voltage signals were then processed by the on-chip STHNN, in which the corresponding categories of input images were measured simply as the highest voltage amplitude among four output neurons. With the use of down-sampling operation, a large group of characters cannot be represented or differentiated by a 4 × 4 array, and thus, only 12 characters were adopted in this experiment.

During the evaluation, 128 measurements were carried out for each set of random characters to demonstrate the robustness of our fabricated STHNN prototype. Table II depicts the classification accuracy of 12 selected characters deployed in 9 different testing sets. Under the scenario without noise, the average classification accuracy for 4,608 images among 9 testing sets reaches 86.9%. Since the down-sampled images of characters "A, F and P" as well as "H and N" are difficult to be differentiated in a small array, a lower classification

TABLE III
PERFORMANCE COMPARISON TO THE STATE-OF-THE-ART NEUROMORPHIC CHIPS

	[45]	[46]	[47]	[48]	[49]	[50]	This Work
Technology	180nm	65nm	10nm	28nm	130nm	130nm	180nm
Die Area	$51.4mm^2$	$16mm^2$	$1.72mm^2$	$5.95mm^2$	$21.82mm^2$	$1.79mm^2$	$9mm^2$
Algorithm	CNN	CNN	SNN	CNN	MLP	RBM	STHNN
Memory Cell	SRAM	SRAM	SRAM	SRAM	ReRAM	ReRAM	ReRAM
Memory Mode	–	–	–	–	CIM	CIM	CIM
Weight Precision	–	–	7-bit	1-bit	3-bit signed	1-bit	1-bit
On-chip Learning	yes	no	yes	–	yes	no	no
Core Structure	neuron	–	neuron	DAC	ADC	neuron	neuron
Neuron Type	IF	–	LIF	–	–	IF	LIF
Latency	15.4ns	–	–	–	51.1ns	–	185ns
Supply Voltage	1.8V	1V	0.9V	0.8V	5V	1.8V	1.8V
Power Consumption	4mW	278mW	–	0.899mW	–	0.05–to–2.2mW	33mW
Application	DVS Images	CIFAR-1000	MNIST	CIFAR-10	MNIST	MNIST	NIST
Inference Speed	–	34.7frames/s	–	380frames/s	77μs/image	–	10μs/image
Accuracy	≈100%	–	89%	86.05%	94.4%	–	86.9% w/o noise 73.9% with noise

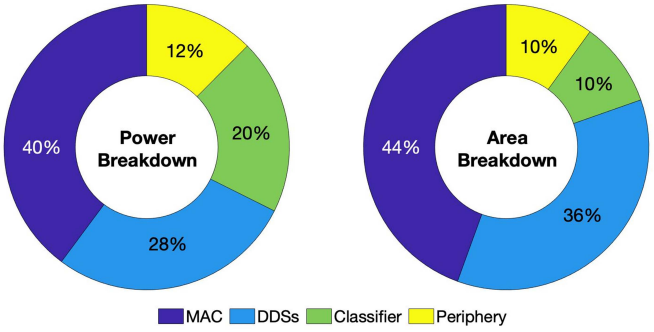


Fig. 14. Power distribution during inference @ 33mW and area breakdown of neural core @ $0.814mm^2$.

accuracy is reported when they are examined under the same group. Beyond that, the classification accuracy was also evaluated with the introduction of noise by randomly overwriting the information of 1–to–3 pixels for each down-sampled images. It can be deduced from measurement that a higher inference error occurs with the introduction of noise, reducing the average classification accuracy to 73.9%.

C. Power and Area Analysis

In this experiment, the supply voltage was fixed at 1.8V and the clock frequency was set to be 1MHz. The power distribution and the area breakdown of our fabricated STHNN prototype are demonstrated in Fig. 14, where the total power consumption of 33mW is reported during the inference operation. The feature extractor together with analog MAC operators occupy 40% of total power and 44% of total area; 8× DDSs occupy 28% and 36% of total power and area, respectively; the neural classifier occupies 20% of total power and 10% of total area, and the rest is occupied by peripheries (*e.g.*, analog/digital buffers, reference voltage/current generators, etc.). Design specifications of our fabricated STHNN prototype and the comparison to the state-of-the-art neuromorphic chips are summarized in Table III. Due to the limitation of a small-scale MAC operator, the reported performance efficiency of 393×10^{-6}TOPS/mm^2 is significantly lower. However, our fabricated STHNN prototype demonstrates the possibility of bridging FNN and DDS in a processing pipeline for neuromorphic applications, potentially yielding a competitive performance with a simplified network structure.

V. APPLICATION EVALUATION

In our hardware modeling, a small-scale of STHNN is implemented on the test chip for the functional verification. To further demonstrate the advantages and the reliability of our STHNN over other DNN models, in this section, a mathematical model of our STHNN was implemented in TensorFlow, and its performance was evaluated through the MNIST handwritten digit database [52] and the ORACLE radio frequency (RF) fingerprinting dataset [53]. The performance was then compared to our baseline DNN models (MLP, CNN, LSTM and GRU) and the cutting-edge DNN models.

A. Experimental Setup

In this experiment, the performance was evaluated through the 4-core Intel i7-6700K CPU and 16G RAM. All weight matrices and bias vectors of baseline models were updated through the gradient-based backpropagation learning algorithm by minimizing the cross-entropy loss through the Adam optimizer. Learning parameters were set as follows: an L2 regularization was utilized to prevent the overfitting, the dropout rate on the readout layer was set to be 0.5, and the learning rate was set to be 1×10^{-3}.

The database used in the evaluation of handwritten digits contains 70,000 28×28-pixel black-and-white images. In this task, 55,000 samples were adopted for training and 10,000 samples were adopted for testing. Likewise, the dataset used in the evaluation of RF devices contains 16 USPR X310 transmitter radios, each of which has 20 million bit-similar in-phase and quadrature (IQ) samples collected from over-the-air WiFi transmission. In this task, 512,000 complex IQ samples were drawn for each RF device,

TABLE IV
PERFORMANCE COMPARISON TO OUR BASELINE MODELS ON MNIST DATABASE

	MLP	CNN	LSTM	GRU	STHNN
Network Structure	2× dense	2× convolution 2× max-pooling 1× dense	1× recurrent	1× recurrent	1× dense 1× DDS 1× dense
# of Kernels	–	$64 \times [3, 3]$	–	–	–
# of neurons	2× 256	256	256	256	various
Training Epochs	10	10	10	10	1
Classification Accuracy	97.37%	98.58%	98.75%	98.89%	97.53% @ 2048 neurons 90.27% @ 256 neurons
Training Time	1min	26mins	5.5mins	4.5mins	4mins @ 2048 neurons 15secs @ 256 neurons

TABLE V
PERFORMANCE COMPARISON TO OUR BASELINE MODELS ON RF FINGERPRINTING DATASET

	MLP	CNN	LSTM	GRU	STHNN
Network Structure	2× dense	2× convolution 2× max-pooling 1× dense	1× recurrent	1× recurrent	1× dense 1× DDS 1× dense
# of Kernels	–	$64 \times [2, 3]$	–	–	–
# of neurons	2× 512	512	512	512	various
Training Epochs	500	50	100	100	1
Classification Accuracy	6.25%	94.56%	95.4%	97.61%	93.9% @ 2048 neurons 76.11% @ 512 neurons
Training Time	53mins	1h24mins	35mins	30mins	19mins @ 2048 neurons 43secs @ 512 neurons

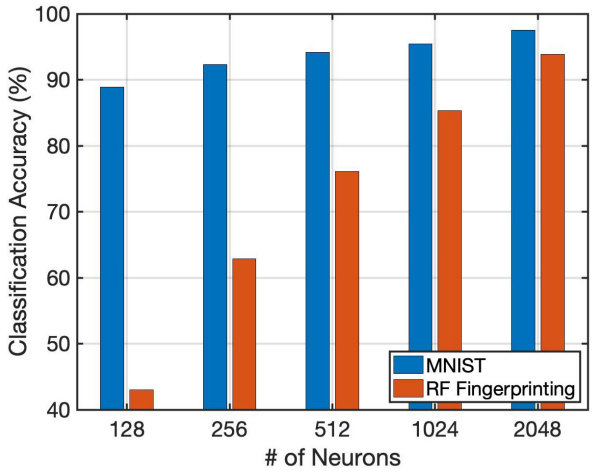


Fig. 15. Classification accuracy of STHNN with respect to various numbers of neurons in the MAC operator.

in which 75% of samples were adopted for training and 25% of samples were adopted for testing.

B. Classification Accuracy

The classification accuracy of our STHNN with respect to various numbers of neurons in the MAC operator is plotted in Fig. 15. It can be observed that a higher classification accuracy can be achieved as the number of neurons increases. However, a longer training time is then required to train the network, *e.g.*, it takes 15secs to run a 256-neuron but 4mins to run a 2048-neuron on the MNIST database; beyond that, it takes 43secs to run a 512-neuron but 19mins to run a 2048-neuron on the RF fingerprinting dataset. The performances of our STHNN and baseline models are summarized in Table IV and V.

In the evaluation of handwritten digits, CNN achieves the highest classification accuracy among our baseline models due to its feature-based learning mechanism, and yet, the longest training time is reported as the convolution operation is computationally expensive. It can be discovered that our STHNN achieves the lowest classification accuracy when the number of neurons is set to be the same as in all other models. To perform the same, the number of neurons in our STHNN had to be increased to 2048. Despite that our STHNN requires more number of neurons to reach an identical classification accuracy, the reported training time is 1.6–to–9.8× faster than alternative neural networks. Moreover, MLP performs 4× faster than our STHNN with a similar classification accuracy due to its simple structure and the simplicity of database, and yet, this is not been the case for other complicated applications.

In the evaluation of RF devices, the reported training time on our STHNN is 1.9–to–4.4× faster than alternative neural networks, while the classification accuracy is merely 1–to–3.5 percentage points poorer. In this task, the MLP is not trainable in such a temporal dataset. The same phenomenon is also reported in our previous work [13] using the transmitted symbol detection task on 5G multiple-input multiple output orthogonal frequency-division multiplexing (MIMO-OFDM) systems, indicating that the simple MLP structure is not capable to execute temporal applications.

The DDS used in our STHNN is mainly built of a single NNA and a dynamic MDL. Such a processing structure does not contain trainable weights, and thus, the learning capability and the classification accuracy are naturally lower than the conventional DNN models. By contrast, such a simple processing structure potentially reduces the design and training complexity of ASIC-based DNN designs without significantly degrading its classification accuracy.

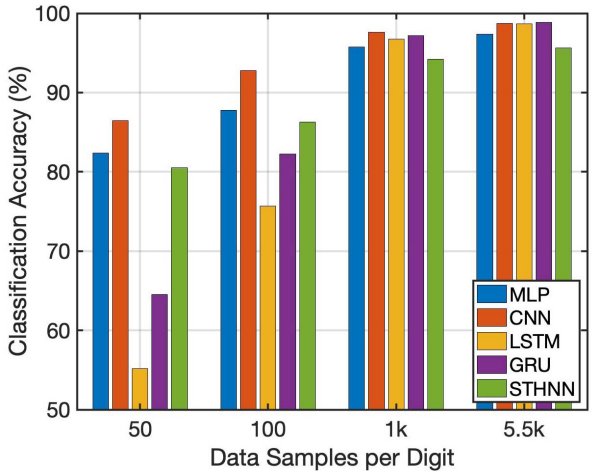


Fig. 16. Classification accuracy with respect to various data samples on MNIST database.

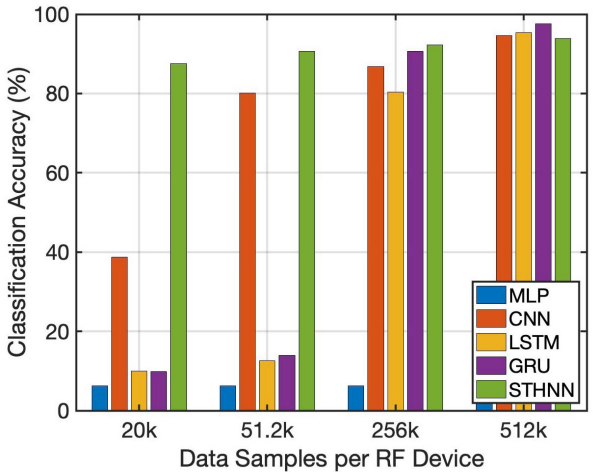


Fig. 17. Classification accuracy with respect to various data samples on RF Fingerprinting dataset.

TABLE VI

CLASSIFICATION ACCURACY COMPARISON TO THE CUTTING-EDGE DNN MODELS USING ON MNIST DATABASE

	Algorithm	Network Structure	Classification Accuracy
[54]	binary MLP	3× dense	95.8%
[55]	spiking CNN	3× convolution 3× max-pooling 1× dense	97.2%
[56]	MLP	7× dense	98.3%
[57]	CNN (LeNET–5)	3× convolution 2× pooling 2× dense	99.1%
[58]	MLP	–	99.4%
This Work	STHNN	1× dense 1× DDS 1× dense	97.53%

TABLE VII

CLASSIFICATION ACCURACY COMPARISON TO THE CUTTING-EDGE DNN MODELS ON RF FINGERPRINTING DATASET

	Algorithm	Network Structure	Classification Accuracy
[53]	CNN	8× convolution 4× max-pooling 2× dense	98.6% @ 16 RF devices
[59]	CNN	2× convolution 1× max-pooling 1× dense	98.6% @ 16 RF devices
[60]	CNN	2× convolution 1× max-pooling 1× dense	98% @ 5 RF devices
[61]	CNN	5× convolution 5× max-pooling 1× dense	93.4% @ 17 RF devices
This Work	STHNN	1× dense 1× DDS 1× dense	93.9% @ 16 RF devices

C. Reliability

Since the output weights of our STHNN are trainable in a single step, it is often possible to successfully train such a network with significantly fewer data. Such a capability is summarized in Fig. 16 and 17 together with our baseline models. In the evaluation of handwritten digits, both MLP, CNN and STHNN are capable to maintain their classification accuracy even with fewer training samples, while the others have a significant reduction. Unsurprisingly, with the recurrent nature embedded in the DDS, our STHNN does not have a strong capability in enhancing the classification accuracy of static data, and yet, the computational efficiency of STHNN is still 9.8× higher than the one with CNN. Table VI summarizes the classification accuracy of our STHNN compared to the cutting-edge DNN models. In the evaluation of RF devices, only our STHNN is capable to maintain its classification accuracy even with fewer training samples, while alternative neural networks have a significant reduction. Likewise, the computational efficiency of STHNN is also 4.4× higher than the others. The comparison to the cutting-edge DNN models are summarized in Table VII.

VI. CONCLUSION

To support the integration of DNNs and neuromorphic computing, we introduce and fabricate a CIM-based STHNN. To be specific, a set of neural computations is implemented to realize the spatial and temporal information processing, wherein CIM-based feature extractor and recurrent-based DDS are incorporated. The use of Tikhonov regularization significantly reduces the training complexity and eschews the gradient of error functions to be recursively propagated backward. A 180nm prototype chip is fabricated with 86.9% on-chip classification accuracy in handprinted alphabet characters at a power consumption of 33mW. Moreover, numerical evaluations demonstrate that our STHNN offers up to 9.8× speedup compared to the state-of-the-art DNN models while yielding a competitive classification accuracy. In conclusion, this work simplifies the neural computing architecture with a unique training mechanism; therefore, a complex machine learning application can be processed with a simple system-level design. Even with a less advanced CMOS process, this work

is still capable to realize a high classification accuracy with an appreciable power consumption, opening the door to efficient neuromorphic applications and mobile edge devices.

REFERENCES

[1] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for Internet of Things data analysis: A survey," *Digit. Commun. Netw.*, vol. 4, no. 3, pp. 161–175, Aug. 2018.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[3] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The microsoft 2017 conversational speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5934–5938.

[4] L. Deng *et al.*, "Tianjic: A unified and scalable chip bridging spike-based and continuous neural computation," *IEEE J. Solid-State Circuits*, vol. 55, no. 8, pp. 2228–2246, Aug. 2020.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3123–3131.

[7] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[8] A. Ardakani, C. Condo, M. Ahmadi, and W. J. Gross, "An architecture to accelerate convolution in deep neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 4, pp. 1349–1362, Apr. 2018.

[9] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks-with an erratum note," German Nat. Res. Center Inf. Technol., Bonn, Germany, GMD Rep. 148, 2001, p. 13, vol. 148, no. 34.

[10] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, no. 11, pp. 2531–2560, Nov. 2002.

[11] M. Han and M. Xu, "Laplacian echo state network for multivariate time series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 238–244, Jan. 2018.

[12] S. Wen, R. Hu, Y. Yang, T. Huang, Z. Zeng, and Y.-D. Song, "Memristor-based echo state network with online least mean square," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 9, pp. 1787–1796, Sep. 2019.

[13] K. Bai, Y. Yi, Z. Zhou, S. Jere, and L. Liu, "Moving toward intelligence: Detecting symbols on 5G systems through deep echo state network," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 10, no. 2, pp. 253–263, Jun. 2020.

[14] L. Appeltant *et al.*, "Information processing using a single dynamical node as complex system," *Nature Commun.*, vol. 2, no. 1, pp. 1–6, Sep. 2011.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[16] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.

[17] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," 2015, *arXiv:1506.06579*. [Online]. Available: http://arxiv.org/abs/1506.06579

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[19] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2553–2561.

[20] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

[21] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers Neurosci.*, vol. 12, p. 774, Oct. 2018.

[22] C. Eliasmith *et al.*, "A large-scale model of the functioning brain," *Science*, vol. 338, no. 6111, pp. 1202–1205, Nov. 2012.

[23] N. K. Kasabov, "NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data," *Neural Netw.*, vol. 52, pp. 62–76, Apr. 2014.

[24] M. Hu, Y. Chen, J. J. Yang, Y. Wang, and H. H. Li, "A compact memristor-based dynamic synapse for spiking neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 8, pp. 1353–1366, Aug. 2017.

[25] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*. [Online]. Available: http://arxiv.org/abs/1506.00019

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*. [Online]. Available: http://arxiv.org/abs/1409.1259

[28] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[29] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[30] K. Xu *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," *Comput. Sci.*, vol. 2015, pp. 2048–2057, Feb. 2015.

[31] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," 2017, *arXiv:1704.02971*. [Online]. Available: http://arxiv.org/abs/1704.02971

[32] B. Hanin, "Which neural net architectures give rise to exploding and vanishing gradients?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 582–591.

[33] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1310–1318.

[34] I. B. Yildiz, H. Jaeger, and S. J. Kiebel, "Re-visiting the echo state property," *Neural Netw.*, vol. 35, pp. 1–9, Nov. 2012.

[35] K. Bai and Y. Yi, "DFR: An energy-efficient analog delay feedback reservoir computing system for brain-inspired computing," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 14, no. 4, pp. 1–22, Dec. 2018.

[36] K. Bai, Q. An, L. Liu, and Y. Yi, "A training-efficient hybrid-structured deep neural network with reconfigurable memristive synapses," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 1, pp. 62–75, Oct. 2020.

[37] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, "Parallel photonic information processing at gigabyte per second data rates using transient states," *Nature Commun.*, vol. 4, no. 1, pp. 1–7, Jun. 2013.

[38] L. Larger *et al.*, "Photonic information processing beyond Turing: An optoelectronic implementation of reservoir computing," *Opt. Exp.*, vol. 20, no. 3, pp. 3241–3249, 2012.

[39] H. Zhang, G. Chen, B. C. Ooi, K.-L. Tan, and M. Zhang, "In-memory big data management and processing: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 7, pp. 1920–1948, Jul. 2015.

[40] T. W. Molter and M. A. Nugent, "The generalized metastable switch memristor model," in *Proc. 15th Int. Workshop Cellular Nanosc. Netw. Appl.*, 2016, pp. 1–2.

[41] M. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, no. 4300, pp. 287–289, Jul. 1977.

[42] C. Zhao, K. Hamedani, J. Li, and Y. Yi, "Analog spike-timing-dependent resistive crossbar design for brain inspired computing," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 1, pp. 38–50, Mar. 2018.

[43] C. Liu, Q. Yang, C. Zhang, H. Jiang, Q. Wu, and H. H. Li, "A memristor-based neuromorphic engine with a current sensing scheme for artificial neural network applications," in *Proc. 22nd Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2017, pp. 647–652.

[44] K. Ikeda and K. Matsumoto, "High-dimensional chaotic behavior in systems with time-delayed feedback," *Phys. D, Nonlinear Phenomena*, vol. 29, nos. 1–2, pp. 223–235, Nov. 1987.

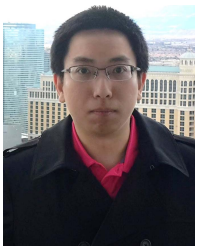
[45] G. Indiveri, F. Corradi, and N. Qiao, "Neuromorphic architectures for spiking deep neural networks," in *IEDM Tech. Dig.*, Oct. 2015, pp. 2–4.

[46] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, "14.5 Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Jan. 2016, pp. 127–138.

[47] G. K. Chen, R. Kumar, H. E. Sumbul, P. C. Knag, and R. K. Krishnamurthy, "A 4096-neuron 1M-synapse 3.8-pJ/SOP spiking neural network with on-chip STDP learning and sparse weights in 10-nm FinFET CMOS," *IEEE J. Solid-State Circuits*, vol. 54, no. 4, pp. 992–1002, Apr. 2019.

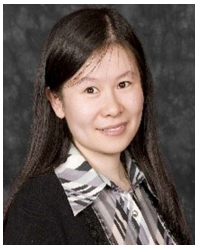
[48] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An always-on $3.8\mu j$/86% cifar-10 mixed-signal binary CNN processor with all memory on chip in 28-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 158–172, Oct. 2018.

[49] Q. Liu *et al.*, "33.2 A fully integrated analog ReRAM based 78.4TOPS/W compute-in-memory chip with fully parallel MAC computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 500–502.
[50] W. Wan *et al.*, "33.1 A 74 TMACS/W CMOS-RRAM neurosynaptic core with dynamically reconfigurable dataflow and *in-situ* transposable weights for probabilistic graphical models," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 498–500.
[51] P. Grother and K. Hanaoka, "Nist special database 19 handprinted forms and characters 2nd edition," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep., 2016.
[52] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Dec. 1998.
[53] K. Sankhe *et al.*, "No radio left behind: Radio fingerprinting through deep learning of physical-layer hardware impairments," *IEEE Trans. Cognit. Commun. Netw.*, vol. 6, no. 1, pp. 165–178, Mar. 2020.
[54] Y. Umuroglu *et al.*, "FINN: A framework for fast, scalable binarized neural network inference," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, Feb. 2017, pp. 65–74.
[55] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, S. J. Thorpe, and T. Masquelier, "Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks," 2018, *arXiv:1804.00227*. [Online]. Available: http://arxiv.org/abs/1804.00227
[56] D. Pedamonti, "Comparison of non-linear activation functions for deep neural networks on MNIST classification task," 2018, *arXiv:1804.02763*. [Online]. Available: http://arxiv.org/abs/1804.02763
[57] C. Yadav and L. Bottou, "Cold case: The lost mnist digits," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 13443–13452.
[58] H. Guo, S. Wang, J. Fan, and S. Li, "Learning automata based incremental learning method for deep neural networks," *IEEE Access*, vol. 7, pp. 41164–41171, 2019.
[59] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized radio clAssification through convolutional neuraL nEtworks," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2019, pp. 370–378.
[60] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, "Deep learning convolutional neural networks for radio identification," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 146–152, Sep. 2018.
[61] Y. Shi, K. Davaslioglu, Y. E. Sagduyu, W. C. Headley, M. Fowler, and G. Green, "Deep learning for RF signal classification in unknown and dynamic spectrum environments," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Nov. 2019, pp. 1–10.



Kangjun Bai (Student Member, IEEE) received the B.S. and M.S. degrees in electrical engineering and embedded electrical and computing systems from San Francisco State University (SFSU), San Francisco, USA, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with The Bradley Department of Electrical and Computer Engineering (ECE), Virginia Tech, Blacksburg, USA. His research interests include emerging technologies for artificial intelligence, including very large scale integration (VLSI) circuits and systems, neuromorphic computing, deep learning, machine learning algorithms, and machine intelligence applications.

Lingjia Liu (Senior Member, IEEE) is currently an Associate Professor with The Bradley Department of Electrical Engineering and Computer Engineering, Virginia Tech. He is also an Associate Director of the Wireless@VT. Prior to joining VT, he was an Associate Professor with the EECS Department, The University of Kansas (KU). He spent more than four years working with the Mitsubishi Electric Research Laboratory (MERL) and the Standards and Mobility Innovation Laboratory, Samsung Research America (SRA), where he received Global Samsung Best Paper Award in 2008 and 2010. He was leading Samsung's efforts on multiuser MIMO, CoMP, and HetNets in LTE/LTE-Advanced standards. His general research interests include emerging technologies for Beyond 5G cellular networks, including machine learning for wireless networks, massive MIMO, massive MTC communications, and mmWave communications.



Yang Yi (Senior Member, IEEE) is currently an Associate Professor with The Bradley Department of Electrical Engineering and Computer Engineering, Virginia Tech. Her research interests include very large scale integrated (VLSI) circuits and systems, computer-aided design (CAD), neuromorphic architecture for brain-inspired computing systems, and low-power circuits design with advanced nano-technologies for high-speed wireless systems.