Towards Temporal Information Processing – Printed Neuromorphic Circuits with Learnable Filters

Haibin Zhao*
haibin.zhao@kit.edu
Karlsruhe Institute of Technology

Priyanjana Pal*
priyanjana.pal@kit.edu
Karlsruhe Institute of Technology

Michael Hefenbrock
michael.hefenbrock@revoai.de
RevoAI GmbH

Michael Beigl
michael.beigl@kit.edu
Karlsruhe Institute of Technology

Mehdi B. Tahoori
mehdi.tahoori@kit.edu
Karlsruhe Institute of Technology

ABSTRACT

Many consumer products such as wearable and disposable electronics require flexibility, biocompatibility and ultra low-costs, which can hardly be matched by silicon electronics. Therefore, printed electronics (PE) becomes a competitive candidate thanks to its additive manufacturing. To address fundamental signal-processing tasks, printed neuromorphic circuits (pNCs) have received increasing attention, as they can achieve promising computational capabilities by assembling simple circuit primitives. However, many target domains of PE are based on processing temporal sensory data, which can not be reached by existing pNCs, since they lack components with time dependencies. This paper proposes a printed temporal processing block that combines existing pNCs with learnable filters. We model the proposed circuit and proposed the corresponding training objective to enable their bespoke design. Simulations on 15 benchmark time-series datasets reveal that, the proposed circuits can effectively process temporal data by using 1.5× and 1.3× of device counts and power respectively. The classification accuracy reaches 98% of that from classic Elman recurrent neural networks.

CCS CONCEPTS

• **Hardware** → **Flexible and printable circuits**; • **Computing methodologies** → *Machine learning approaches.*

KEYWORDS

printed electronics, neuromorphic computing, temporal information processing, recurrent neural network

ACM Reference Format:
Haibin Zhao, Priyanjana Pal, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2023. Towards Temporal Information Processing – Printed Neuromorphic Circuits with Learnable Filters. In *18th ACM International Symposium on Nanoscale Architectures (NANOARCH '23), December 18–20, 2023, Dresden, Germany*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3611315.3633249

*Authors contributed equally to this work.

NANOARCH '23, December 18–20, 2023, Dresden, Germany

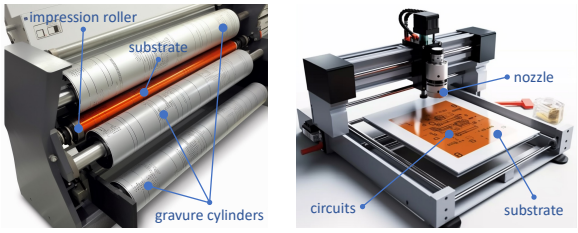
ACM ISBN 979-8-4007-0325-6/23/12…$15.00
https://doi.org/10.1145/3611315.3633249

1 INTRODUCTION

With the growth of the Internet of Things (IoT), next-generation smart electronics, such as smart labels [2], smart band-aids [20], and smart packaging [1], will be enabled by lightweight, flexible, and cost-efficient technologies. Although silicon-based electronics based on lithography processes have been continuously improved in terms of power consumption and integration density, matching the demands of the aforementioned emerging technologies remains challenging. This is mainly due to bulky substrates and requirement of complicated manufacturing equipment. To address the demand of these emerging electronics, progress has been made in developing innovative materials and fabrication methods. One of the most promising approaches in this realm is printed electronics (PE) [3]. Analogous to color printing, printed electronic devices are fabricated through additive printing processes, therefore, PE allows for highly customized electronic devices. Moreover, in contrast to the costly lithography-based manufacturing, the simple manufacturing processes in PE not only significantly reduces manufacturing costs, but also allows depositing functional materials with different properties, facilitating various desired properties, such as softness [12], non-toxicity [11], and bio-degradability [13].

Despite these advantages, PE also come with their own weaknesses. For example, printed devices generally have larger feature sizes, lower integration density, and lower performance compared to silicon counterparts. To this end, printed analog neuromorphic circuits have emerged as a promising computing paradigm in PE. It allows computing directly in the analog domain, and does not necessitate analog-to-digital converters (ADCs), which significantly reduces device counts. Additionally, neuromorphic computing has been proven to have strong computational capability, even through only a series of simple operation primitives [18]. Specifically, the fundamental idea of neuromorphic circuits is to execute the computational operations, i.e., weighted-sums and nonlinear activation functions, in artificial neural networks (ANNs), also known as multilayer perceptrons [6], through specific analog circuit designs, namely resistor crossbars and nonlinear transfer circuits.

Existing research on printed neuromorphic circuits (pNCs) has predominantly focused on bridging the gap between realistic printed circuit design and ANN models. From a circuit design perspective, [7] and [15] enable different activation functions. From the algorithmic level, [26] incorporates the manufacturability and the printing variation into the circuit optimization process, while [25] extended



(a) gravure printing (b) ink-jet printing

Figure 1: Exemplary typical printing technologies: (a) gravure printing and (b) ink-jet printing.

the learnable parameters from weights to nonlinear functions. Notably, the advent of novel functional materials and devices, e.g., memristors [21], may also broaden the possibilities for pNCs.

Nevertheless, the previous research is constrained to the context of implementing multilayer perceptions, where the pNCs are assumed to operate with constant DC or time-independent input signals. This weakness considerably restricts the prospective applications of pNCs, because in many perceived application scenarios (e.g., stress detection [24]), absolute values of temporal sensory data matter less, while their trends and changes are more informative.

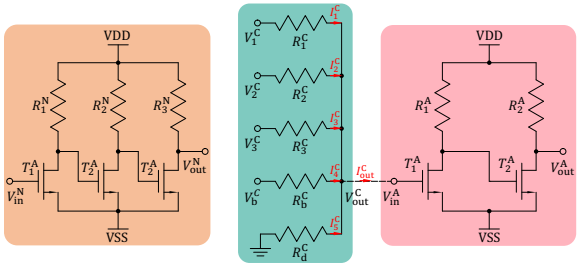
To address this limitation, we propose a learnable and tunable printed temporal processing block (pTPB). By introducing capacitors, the circuits are capable to memorize (or store) information about previous inputs. This means, the circuit outputs are no longer solely dependent on the signal at the current time step, but rather on the signals at all previous time steps. Consequently, by combining existing pNCs with proposed pTPBs, which are referred to as printed temporal processing neuromorphic circuits (pTPNCs), the circuits are capable of processing time-series sensory inputs. By modeling the pTPBs with consideration of the circuit decoupling, we construct a time-domain machine learning model of the pTPBs, which further enables the training of pTPNCs for target tasks. Subsequently, we show that the pTPNC forms an instance of recurrent neural network (RNN) models. Finally, simulations with 15 time-series benchmark datasets validates our design.

The rest of this paper is organized as follows: Sec. 2 introduces the concept of PE, the basis of pNCs and RNNs. Sec. 3 motivates this research, describes the modeling of the pTPBs as well as the pTPNCs, and propose the corresponding training objective. Subsequently, Sec. 4 validates the proposed method and discusses the experiment results. Finally, Sec. 5 summarizes this work.

2 PRELIMINARIES

2.1 Printed Electronics

Printed electronics (PE) is an emerging fabrication technique for next-generation electronic devices. It involves an additive manufacturing process that fabricates circuits by depositing various functional inks onto the substrates. In contrast to conventional lithography-based silicon electronics, PE enables to produce circuits at considerably low costs, owing to the absence of complicated fabrication equipment and steps. Moreover, various fabrication methods can be adapted to specific production scenarios (as shown in Fig. 1), accommodating both small-quantity prototypes during product development phase (via ink-jet printing) and high-volume production (through gravure printing).



(a) (b) (c)

Figure 2: Circuit primitives of the printed neuromorphic circuit (pNC). (a) Inverter-based negative weight circuit. (b) Example of a 3-input, 1-output printed resistor crossbar. (c) Inverter-based printed tanh-like (ptanh) circuit.

Nevertheless, since additive manufacturing typically results in large feature sizes and high variation, PE does not aim to replace silicon-based electronics in precise or complex scenarios, but rather to complement them in edge computing scenarios such as disposable or wearable electronics. In these scenarios, the advantages of PE can be fully leveraged, as the computational demands tends to be simple, so that small circuits are good enough; meanwhile, these tasks are often highly specific and thus require a highly flexible and custom manufacturing process.

Most of the advanced ink-jet printed field-effect transistors (FETs) utilize organic materials and incorporate lithographically structured organic semiconductors as channels between the source and drain electrodes. Typically, organic FET (OFET) structures rely on P-type materials, which exhibit a notably low field-effect mobility [8] and operate at a high supply voltage range ($\geq$ 25V). As a result, OFET technology is not suitable for the intended applications. In this regard, low-voltage and low-power devices are favored, with inorganic oxide semiconductors being more feasible candidates. Current inorganic PE research focuses on ink-jet printed N-type Electrolyte-Gated Transistor (nEGT), whose band structure of metal oxides facilitates high electron mobility, enabling nEGTs to operate at sub-1V supply voltage.

2.2 Printed Analog Neuromorphic Circuits

To respect the large feature size of PE, processing information directly in the analog domain becomes favorable, as it eliminates the conversion between digital and analog signals. Among computational paradigms, neuromorphic computing has attracted considerable attention due to its ability to implement complicated computing functionalities [18]. Moreover, this impressive ability requires only the combination of simple computing primitives, providing significant advantages in terms of circuit design, optimization, and manufacturing. These primitives are introduced in the following.

Resistor crossbar. Fig. 2(b) depicts a resistor crossbar in a pNC, emulating the weighted-sum operations utilized in ANNs. This circuit is also prevalently used in other fields such as in-memory computing [19]. The behavior of the crossbar is characterized by

$$\sum_i \frac{V_i^{\mathrm{C}} - V_{\mathrm{out}}^{\mathrm{C}}}{R_i^{\mathrm{C}}} + \frac{V_{\mathrm{b}}^{\mathrm{C}} - V_{\mathrm{out}}^{\mathrm{C}}}{R_{\mathrm{b}}^{\mathrm{C}}} - \frac{V_{\mathrm{out}}^{\mathrm{C}}}{R_{\mathrm{d}}^{\mathrm{C}}} = I_{\mathrm{out}}^{\mathrm{C}}.$$

Here, the superscript $(\cdot)^{\mathrm{C}}$ indicates the variables within the crossbar. Due to the high resistivity of the succeeding circuit, i.e., the printed tanh-like circuit, the current $I_{\mathrm{out}}^{\mathrm{C}}$ can be neglected [22]. Furthermore,

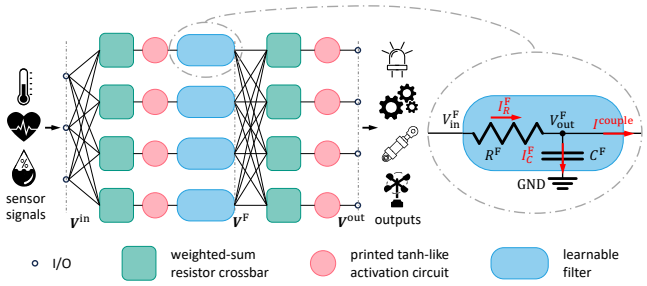


Figure 3: Schematic of a 3-input 4-output printed temporal processing block (pTPB) that receives sensor signals and yields outputs to subsequent devices.

by expressing the resistance R as the corresponding conductance $g = 1/R$ and fixing $V_b = 1$V, the equation can be rearranged to

$$V_{out}^{C} = \sum_i \frac{g_i^C}{G} V_i^C + \frac{g_b^C}{G}, \tag{1}$$

where G refers to the summed conductance of all the conductances in the crossbar, i.e., $\sum_i g_i^C + g_b + g_d$. It can be seen that the output voltage of the crossbar V_{out}^C can be interpreted as the weighted-sum of the corresponding input voltages V_i^C, where the weights and bias are represented by the ratios of the conductances. Consequently, by appropriate designing and printing of the conductances, desired weights and biases can be achieved. Notably, different from ANNs, the embodied weights and biases in Eq. 1 are limited to less than 1. Moreover, due to the hardware constraints, they can only be positive values, therefore, when negative weights or biases are required, inverter-based negative weight circuits (Fig. 2a) are prepended to the corresponding crossbar resistors to transform the corresponding input voltages into negative ones.

Printed tanh-like circuit. As shown in Fig. 2(c), succeeding the resistor crossbar, an inverter-based circuit, namely printed tanh-like (ptanh) circuit, is connected to resemble the nonlinear activation functions in the ANNs. The transfer characteristic of this circuit is expressed by an appropriately parameterized tanh(·) function, i.e.,

$$V_{out}^{A} = \text{ptanh}(V_{in}^{A}) = \eta_1 + \eta_2 \cdot \tanh\left((V_{in}^{A} - \eta_3) \cdot \eta_4\right),$$

where the superscript $(\cdot)^A$ refers to the variables in this activation circuit. Note that, the η_i modify the translation and scaling of the original tanh(·) function and are only auxiliary parameters that are implicitly determined by physical quantities of the components in the circuit, namely $q^A = [R_1^A, R_2^A, T_1^A, T_2^A]$.

There are already technologies to take the variation (e.g., printing errors) sensitivity of analog computing paradigms into account during training to improve the robustness of pNCs [26]. Moreover, considering the target application of PE like disposable electronics for specific use cases, these circuits do not employ costly reconfigurable components such as memristors, rather, they are printed as fixed devices after their design and parameter optimization.

2.3 Recurrent Neural Networks

Recurrent neural networks (RNNs) were primarily proposed to handle inputs with variable lengths, as the input dimensionality for an MLP is pre-determined. By incorporating an internal hidden state h, an RNN is then capable of processing variable length inputs through repeated state updates. RNNs have demonstrated remarkable success in areas such as handwriting recognition [9]. Notably, RNNs are theoretically Turing-complete, meaning that they can execute arbitrary programs to process any given input sequences [10]. A general formulation of RNN state equations is given by

$$\begin{aligned} h_k &= f_1(f_2(h_{k-1}) + f_3(\boldsymbol{x}_k)), \\ \boldsymbol{y}_k &= f_4(h_k), \end{aligned} \tag{2}$$

where the subscript $k \in \{0, 1, \cdots, K\}$ refer to the iteration (time step), h_k is the internal hidden state at the k-th step, $\boldsymbol{x}_k$ denotes the input at the k-th step, and $\boldsymbol{y}_k$ represents the output at the k-th step. Moreover, the functions $f_1(\cdot), \cdots, f_4(\cdot)$ are classic operations in ANNs, such as learnable affine mappings and/or activation functions. The specific choices of them vary across different network architectures, e.g., in Elman RNNs [5], $f_2(\cdot)$ and $f_3(\cdot)$ are weighted-sum operations with biases, while $f_1(\cdot)$ and $f_4(\cdot)$ are functions of learnable linear mappings with activation functions.

3 PRINTED TEMPORAL PROCESSING BLOCK

Given their distinct strengths and weaknesses, pNCs aim to complement silicon-based electronics. Thereby, they are typically envisioned to serve as cheap edge computing units in IoT systems, such as wearables devices or smart packaging. However, in some scenarios like stress detection [24], the concrete signal values might be less informative due to physiological variation among individuals. In contrast, the temporal changes in the signal often exhibits more meaningful information. However, this kind of time-series data processing is beyond the reach of the existing pNCs, due to their lack of component to memorize historical signals.

To address this limitation, we propose to introduce learnable printed capacitors into the existing pNCs, which allow the circuit to memorize, and thus process, temporal sensory data. By combining the capacitors with existing primitives in pNCs, we construct the printed temporal processing blocks (pTPB). Additionally, by stacking multiple pTPBs, more sophisticated functions can be realized.

Different application scenarios (i.e., tasks) generally necessitate different temporal processing behaviors. Conveniently, the highly flexible additive printing techniques of PE can enable such task-specific fabrication. To design the bespoke signal processing behaviors for target tasks, we have developed the mathematical model of the proposed pTPB and the corresponding pTPNC, along with an optimization objective. With this approach, the components in pTPBs (e.g., the capacitance) can be optimized alongside the resistances in the crossbars (representing weights and biases).

3.1 Modeling of Temporal Processing Unit

The structure of the pTPB is represented in Fig. 3. It can be seen that the most essential part in the circuit are framed by the blue boxes, which consist of capacitors and resistors, resembling RC low-pass filters. Therefore, we will first model these filters while considering their coupling with the rest of the circuit. Afterwards, we will develop the model to cover the entire pTPB.

Modeling of single filter. Initially, we concentrate on modeling the filter without considering its coupling to the successive circuit. Taking the first unit in Fig. 3 as an example, we obtain:

$$\begin{aligned} I_R^{\mathrm{F}} &= \left(V_{\mathrm{in}}^{\mathrm{F}} - V_{\mathrm{out}}^{F}\right)/R^{\mathrm{F}}, \\ I_C^{\mathrm{F}} &= C^{\mathrm{F}}\mathrm{d}V_{\mathrm{out}}^{\mathrm{F}}/\mathrm{d}t, \\ I_R^{\mathrm{F}} &= I_C^{\mathrm{F}}, \end{aligned} \tag{3}$$

where the superscript $(\cdot)^{\mathrm{F}}$ indicates the values in this filtering unit. Therefore, the differential equation of the capacitor voltage $V_{\mathrm{out}}^{\mathrm{F}}$ with respect to time can be expressed by

$$\frac{\mathrm{d}V_{\mathrm{out}}^{\mathrm{F}}}{\mathrm{d}t} = -\frac{1}{R^{\mathrm{F}}C^{\mathrm{F}}}V_{\mathrm{out}}^{\mathrm{F}} + \frac{1}{R^{\mathrm{F}}C^{\mathrm{F}}}V_{\mathrm{in}}^{\mathrm{F}}.$$

By using backward Euler integration, we obtain the update of the V^{F}:

$$\begin{aligned} V_{\mathrm{out}k}^{\mathrm{F}} &= \underbrace{\frac{R^{\mathrm{F}}C^{\mathrm{F}}}{R^{\mathrm{F}}C^{\mathrm{F}}+\Delta t}}_{=:\beta} V_{\mathrm{out}k-1}^{\mathrm{F}} + \underbrace{\frac{\Delta t}{R^{\mathrm{F}}C^{\mathrm{F}}+\Delta t}}_{=:1-\beta} V_{\mathrm{in}k}^{\mathrm{F}} \\ &= \beta V_{\mathrm{out}k-1}^{\mathrm{F}} + (1-\beta)V_{\mathrm{in}k}^{\mathrm{F}}, \end{aligned} \tag{4}$$

where Δt refers to the step size of the temporal discretization, $V_{\mathrm{in}k}^{\mathrm{F}}$ and $V_{\mathrm{out}k}^{\mathrm{F}}$ are the input and output of the filter at time step k. Evidently, $\beta \in (0,1)$ depends on R^{F} and C^{F}, thus, by finding suitable R^{F} and C^{F}, a desired filtering behavior can be achieved. In this work, as these values will be learned jointly with the crossbar resistors to fit the specific tasks, we refer to these units as *learnable filters*.

Modeling of coupled filter. To connect the learnable filters with the resistor crossbars, it is imperative to take their coupling into account. This coupling primarily results from the fact that the current flowing through the resistor R^{F} does not fully feed into the capacitor C^{F}, but is partially shunted towards the crossbar, see red arrows in Fig. 3. To reflect this coupling in our model, we modify Eq. 3 to

$$I_R^{\mathrm{F}} = I_C^{\mathrm{F}} + I^{\mathrm{couple}} =: \mu I_C^{\mathrm{F}},$$

with I^{couple} refers to the coupling current flows towards crossbar, and $\mu := 1 + I^{\mathrm{couple}}/I_C^{\mathrm{F}}$ is a decoupling factor. Consequently, Eq. 4 is reformulated to

$$\begin{aligned} V_{\mathrm{out}k}^{\mathrm{F}} &= \underbrace{\frac{\mu R^{\mathrm{F}}C^{\mathrm{F}}}{\mu R^{\mathrm{F}}C^{\mathrm{F}}+\Delta t}}_{=:\beta'} V_{\mathrm{out}k-1}^{\mathrm{F}} + \underbrace{\frac{\Delta t}{\mu R^{\mathrm{F}}C^{\mathrm{F}}+\Delta t}}_{=:1-\beta'} V_{\mathrm{in}k}^{\mathrm{F}} \\ &= \beta' V_{\mathrm{out}k-1}^{\mathrm{F}} + (1-\beta')V_{\mathrm{in}k}^{\mathrm{F}}, \end{aligned} \tag{5}$$

It is notable that μ is contingent on the values of R^{F}, C^{F} and R^{C}, which vary continuously during the training process. Additionally, μ is also determined by the frequency of the input signal, which is generally agnostic in the design stage. Therefore, to consider the circuit coupling in the training process, we determine the general range of μ through the SPICE simulation with the printed Process Design Kit (pPDK) [17]. Specifically, by performing SPICE simulations on the target datasets (see Sec. 4) with general printable resistances and capacitances, we empirically determined $\mu \in [1, 1.3]$ for the given applications. More details can be found in Sec. 3.4.

3.2 Modeling of Temporal Processing Block

Although Eq. 5 emulates Eq. 2, it possesses only one learnable parameter, i.e., β'. To expand the design space, and better mimic the expressiveness of classic RNNs, a more sophisticated combination of the learnable filters, crossbars, and ptanh circuits should be designed.

As sketched in Fig. 3, to match analogous computational capabilities of classic Elman RNNs, we first pass the input voltages through resistor crossbars followed by ptanh activation circuits, before feeding them to the filters. Here, the ptanh circuits are introduced to decouple the learnable filters from the preceding crossbars, because proper weighted-sum computation through the crossbar necessitates a negligible output current $I_{\mathrm{out}}^{\mathrm{C}}$. However, the resistivity of the filter circuit is much lower than the crossbars. Additionally, we also apply an identical process to output voltages from learnable filters. In the rest of the work, we refer to this stack of primitive layers as a pTPB, i.e., the whole circuit exemplified in Fig. 3. Consequently, the mathematical model of a pTPB can be described as

$$\begin{aligned} \mathrm{V}_k^{\mathrm{F}} &= \boldsymbol{\beta'} \odot \mathrm{V}_{k-1}^{\mathrm{F}} + (1-\boldsymbol{\beta'}) \odot \mathrm{ptanh}(\mathrm{W}_1\mathrm{V}_k^{\mathrm{in}} + \mathrm{b}_1), \\ \mathrm{V}_k^{\mathrm{out}} &= \mathrm{ptanh}(\mathrm{W}_2\mathrm{V}_k^{\mathrm{F}} + \mathrm{b}_2), \end{aligned} \tag{6}$$

where $\mathrm{V}_k^{\mathrm{in}} \in \mathbb{R}^{N_{\mathrm{in}}}$ and $\mathrm{V}_k^{\mathrm{out}} \in \mathbb{R}^{N_{\mathrm{out}}}$ vectorize the input and output voltages of the pTPB at time point k. $\mathrm{V}_k^{\mathrm{F}} \in \mathbb{R}^{N_{\mathrm{F}}}$ summarizes the output voltages of the filter layer and $\boldsymbol{\beta'} \in \mathbb{R}^{N_{\mathrm{F}}}$ collects the β' values of each filter. Moreover, $\mathrm{W}_1 \in \mathbb{R}^{N_{\mathrm{F}} \times N_{\mathrm{in}}}$, $\mathrm{b}_1 \in \mathbb{R}^{N_{\mathrm{F}}}$, $\mathrm{W}_2 \in \mathbb{R}^{N_{\mathrm{out}} \times N_{\mathrm{F}}}$, and $\mathrm{b}_2 \in \mathbb{R}^{N_{\mathrm{out}}}$ denotes the weighted-sum operations emulated by the corresponding crossbars. Additionally, "$\odot$" indicates element-wise multiplication.

By comparing Eq. 6 with Eq. 2, we conclude that, the designed circuit layer represents an instance of an RNN with $f_1(\cdot)$ and $f_2(\cdot)$ being identity functions, while $f_3(\cdot)$ and $f_4(\cdot)$ are weighted-sums followed by activation functions.

Notably, a pTPNC may consist of multiple pTPBs connected successively for accomplishing more intricate computational tasks. In case of multiple pTPBs, we denote the initial input voltages (typically sensor signals) by $\boldsymbol{x}_k$, and represent the final output of the last layer in the circuit by $\hat{\boldsymbol{y}}_k(\boldsymbol{\beta'}, \mathrm{g}, \boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \cdots, \boldsymbol{x}_0)$, which is a function of $\boldsymbol{\beta'}$ in the learnable filters, the crossbar conductances g, and the input voltages at all time steps $\boldsymbol{x}_k, \cdots, \boldsymbol{x}_0$.

3.3 Training of pTPNCs

In case of training basic pNCs (without pTPB), the cross-entropy loss $L(\cdot)$ can be minimized with respect to crossbar conductances g to decrease the mismatch between the label $\boldsymbol{y}$ and the circuit prediction $\hat{\boldsymbol{y}}(\mathrm{g}, \boldsymbol{x})$ for an input $\boldsymbol{x}$, and thus improve, e.g., the classification accuracy. In contrast, the pTPNCs is time-dependent and allows to obtain predictions for each time step $\boldsymbol{y}_k$ based on the current input $\boldsymbol{x}_k$ and previous inputs $\boldsymbol{x}_{k-1}, \cdots, \boldsymbol{x}_0$. We thus consider the temporal dynamics of the circuit output and, to encourage consistent correct classification at every point in time, the objective function can be modified to

$$\underset{\boldsymbol{\beta'}, \mathrm{g}}{\mathrm{minimize}} \underbrace{\frac{1}{K}\sum_{k=0}^{K} L\left(\hat{\boldsymbol{y}}_k(\boldsymbol{\beta'}, \mathrm{g}, \boldsymbol{x}_k, , \boldsymbol{x}_{k-1}, \cdots, \boldsymbol{x}_0), \boldsymbol{y}\right)}_{\mathcal{L}(\boldsymbol{\beta'}, \mathrm{g}, \boldsymbol{x}_K, \cdots, \boldsymbol{x}_0, \boldsymbol{y})}. \tag{7}$$

Additionally, it is necessary to consider the dependency of the decoupling factor μ and the initial voltages of the capacitors. The former has been previously mentioned in Sec. 3.1, while the latter is generally caused by the preceding input signal. To reduce the dependencies of the circuit coupling (μ) and the initial voltage V_0^F on the results, we integrate our loss function over the value ranges for both variables, assuming $[1, 1.3]$ for μ, and $[0, 1]$ for V_0^F. Through this, we should achieve a configuration of that learnable parameters g and $\boldsymbol{\beta}'$ that is robust to the choice of either value, which leads to the training objective of

$$\underset{\boldsymbol{\beta}', \mathrm{g}}{\text{minimize}} \int \mathcal{L}(\boldsymbol{\beta}', \mathrm{g}, \boldsymbol{x}_K, \cdots, \boldsymbol{x}_0, \boldsymbol{y}, \mu, V_0^F)\, \mathrm{d}\mu\, \mathrm{d}V_0^F.$$

Unfortunately, no analytical solution for the integral (or its gradient with respect to the learnable parameters) exists. We thus rewrite the minimization of the training objective using equivalent formulation as an expected value

$$\underset{\boldsymbol{\beta}', \mathrm{g}}{\text{minimize}}\ \mathbb{E}_{p(\mu), p(V_0^F)} \left\{ \mathcal{L}(\boldsymbol{\beta}', \mathrm{g}, \boldsymbol{x}_K, \cdots, \boldsymbol{x}_0, \boldsymbol{y}, \mu, V_0^F) \right\}, \quad (8)$$

which allows to obtain Monte Carlo estimates of the function value (and its gradients) whenever needed. Consequently, based on the ranges for V_0^F and μ, we choose $p(V_0^F) = \mathcal{U}[0, 1]$ and $p(\mu) \sim \mathcal{U}[1, 1.3]$, i.e., uniform densities over their assumed ranges.

Notably, given that the circuit operates continuously on input signals rather than performing a one-time computing, the circuit latency is implicitly incorporated in the training objective Eq. 8. By encouraging more correct classifications at each time step, the circuit should be trained to achieve correct output as fast as possible.

3.4 Discussion

In this section, we modeled the learnable filters with consideration of the circuit coupling. Subsequently, we proposed the pTPB for temporal data processing, and shown that the proposed pTPB forms an instance of RNN. Finally, we formulated the optimization objective for the pTPNCs. While this work does not currently consider variations in printed circuits into the training objective, similar variation-aware training as proposed in [26] could be introduced in the future work to overcome the sensitivity of analog computing systems to the variations.

Another challenge of this process pertains to the values of the decoupling factor μ. To minimize the coupling effect, the resistances in the filters are designed with lower values (<1kΩ) than that of the resistors in crossbars (100kΩ-10MΩ), while the capacitances are designed as high as the printing technology allows (100nF-100μF). In this work, we determined the range of μ by analyzing of signal frequencies in the experimental datasets (see Sec. 4) and conducting SPICE simulations with pPDK [17]. Notably, this range of μ has taken into account the cases in which the filters might be connected to multiple (up to five) crossbars, which will aggravate the circuit coupling by elevating the coupling current I^{couple}.

4 EVALUATION

To evaluate the pTPNCs, we implemented the proposed approach[1] with PyTorch [16] and conduct experiments on 15 benchmark time-series datasets.

[1]The code is available at https://github.com/Neuromophic/LearnableFilters.

Table 1: Result on 15 benchmark time-series datasets: mean and standard deviation of accuracy from random guess (RG), previous printed neuromorphic circuit (pNC), hardware-agnostic Elman recurrent neural network (RNN), and printed temporal processing neuromorphic circuit (pTPNC).

Dataset	RG	pNC	Elman RNN	pTPNC
CBF	0.335	0.456 ± 0.038	0.683 ± 0.036	0.907 ± 0.015
DPTW	0.441	0.507 ± 0.006	0.764 ± 0.012	0.654 ± 0.007
FRT	0.520	0.597 ± 0.120	0.795 ± 0.030	0.761 ± 0.076
FST	0.492	0.509 ± 0.066	0.798 ± 0.068	0.765 ± 0.015
GPAS	0.390	0.452 ± 0.003	0.768 ± 0.023	0.682 ± 0.106
GPMVF	0.567	0.637 ± 0.054	0.829 ± 0.108	0.891 ± 0.163
GPOVY	0.557	0.540 ± 0.007	1.000 ± 0.000	1.000 ± 0.000
MPOAG	0.483	0.560 ± 0.042	0.708 ± 0.035	0.712 ± 0.006
MSRT	0.283	0.261 ± 0.008	0.625 ± 0.068	0.503 ± 0.208
PowerCons	0.445	0.651 ± 0.010	0.982 ± 0.008	0.801 ± 0.040
PPOC	0.655	0.711 ± 0.001	0.724 ± 0.006	0.743 ± 0.005
SRSCP2	0.464	0.489 ± 0.011	0.742 ± 0.010	0.782 ± 0.010
Slope	0.501	0.559 ± 0.002	0.963 ± 0.036	0.898 ± 0.159
SmoothS	0.268	0.447 ± 0.011	0.648 ± 0.010	0.694 ± 0.063
Symbols	0.152	0.141 ± 0.002	0.660 ± 0.049	0.670 ± 0.052
Average	**0.437**	**0.501 ± 0.025**	**0.779 ± 0.033**	**0.764 ± 0.062**

As the functionality of the printed neuromorphic hardware has been experimentally validated in [14, 23, 24], the experiment is conducted at simulation level based on the pPDK [17].

4.1 Experiment Setup

4.1.1 Dataset preparation. First, we sourced all datasets from the UCR time-series classification archive [4]. Afterwards, we filtered out datasets based on their complexity. Only datasets with N_{in} and N_{out} below 10 are kept to match the typical complexity of the target applications of PE. Subsequently, we preprocess the datasets by resizing the series lengths uniformly to 128, normalized the signal values to the range of $[-1, 1]$, reshuffle and split the datasets into training (60%), validation (20%), and test (20%) sets. Then, we leveraged a 2-layer RNNs as a baseline to remove datasets whose difficulty surpassed the capabilities of general RNNs. Ultimately, the top 15 datasets with optimal RNN performance are retained for the further experiment.

4.1.2 Training Setup. For each dataset, we adopt a 2-layer pTPNC (consisting of two pTPBs) with the number of learnable filters N_{F} equaling to N_{out}. To minimize the objective function, we use the gradient-based Adam optimizer with default parameterization to conduct full-batch training to update the optimization parameters. The initial learning rate is set to 0.1 and get halved after every 100-epoch patience (consecutive updates without improvement) on the validation loss. The training process stops once the learning rate has been reduced below 10^{-5}. The entire training procedure is replicated 10 times, employing different random seeds ranging from 0 to 9. This aims to mitigate the potential impact of unfavorable initialization, and thus to ensures a sufficiently good solution.

4.1.3 Baselines. For comparison, we consider 2-layer basic pNCs without pTPB as a baseline. The topology mirrors that of the pTPNCs, i.e., N_{in}-N_{F}-N_{out} with $N_{\text{F}} = N_{\text{out}}$. This comparison intends to assess the temporal processing ability of both pNCs and pTPNC. Since pNCs are unable to process temporal sensory data, the classification results should form random guesses (RG), which refers to always predicting the most probable class in training data. Besides, we also compare the pTPNCs with the RNNs that we strived to mimic. Specifically, we adopt the Elman RNNs provided in PyTorch, and analogously, we utilize 2-layer RNNs with the number of hidden states (equivalent to the number of learnable filters N_{F}) being equal to N_{out}. After hyperparameter tuning, we initiate the learning rate for RNNs to 0.01, while all other training setups are kept identical to those of the pTPNCs.

4.1.4 Evaluation Result. After training, we select the top three models (trained with three different random seeds) for each dataset according to the

Table 2: Hardware costs of basic printed neuromorphic circuit (pNC) and printed temporal processing neuromorphic circuit (pTPNC).

Dataset	#Transistor		#Resistor		#Capacitor		**#Total Device**		**Power** (mW)	
	pNC	pTPNC	pNC	pTPNC	pNC	pTPNC	pNC	pTPNC	pNC	pTPNC
CBF	18	24	57	84	–	6	75	114	0.471	0.653
DPTW	36	48	150	222	–	12	186	282	1.069	1.501
FRT	12	16	34	50	–	4	46	70	0.272	0.372
FST	12	16	34	50	–	4	46	70	0.276	0.342
GPAS	12	16	34	50	–	4	46	70	0.221	0.374
GPMVF	12	16	34	50	–	4	46	70	0.302	0.389
GPOVY	12	16	34	50	–	4	46	70	0.289	0.324
MPOAG	18	24	57	84	–	6	75	114	0.454	0.625
MSRT	30	40	115	170	–	10	145	220	0.862	1.188
PowerCons	12	16	34	50	–	4	46	70	0.312	0.363
PPOC	12	16	34	50	–	4	46	70	0.226	0.381
SRSCP2	12	16	44	60	–	4	56	80	0.294	0.472
Slope	12	16	34	50	–	4	46	70	0.320	0.388
SmoothS	18	24	57	84	–	6	75	114	0.436	0.610
Symbols	36	48	150	222	–	12	186	282	1.143	1.526
Average	**18**	**23**	**60**	**88**	-	**6**	**78**	**118**	**0.463**	**0.634**

accuracy on the validation set. Note that, in accordance with the objective proposed in Sec. 3.3, we have computed the classification accuracy at every time step, and subsequently average these accuracies over time to yield the overall classification accuracy on a dataset. These selected models are then evaluated on the test set. Ultimately, for each dataset, we summarize its mean accuracy with respect to random seeds and the corresponding standard deviation. The result is presented in Tab. 1. To obtain a straightforward insight on the effectiveness of each models in various scenarios (datasets), we also averaged the accuracy and standard deviation with respect to datasets. The averaged values are reported in the last row of Tab. 1.

4.1.5 Hardware Cost. To investigate the additional hardware resources required by the new circuit design, we collect the device counts and total power consumption of both the previous pNCs and the proposed pTPNCs in different application scenarios (i.e., datasets). Analogously, we averaged the hardware costs across all datasets to report a comprehensive comparison regarding the hardware costs between the pTPNC and its pNC counterpart. The results can be seen in Tab. 2.

4.2 Discussion

As can be seen in Tab. 1, basic pNCs without pTPB are unable to process temporal data, and thereby only achieve similar classification accuracy to that of the random guess. However, by comparison of averaged performance between pTPNCs and basic pNCs, it reveals that pTPNCs are indeed capable of processing time-series data. Moreover, such capability for temporal signal processing requires only approximately 1.5× more devices and 1.3× more power consumption. By comparing the performance of pTPNCs with RNNs, we conclude that the pTPNCs can attain a comparable (98%) classification accuracy to their completely hardware-agnostic Elman RNN counterparts. Interestingly, a closer observation of Tab. 1 reveals that pTPNCs and RNNs do not consistently yield comparable performance on every dataset. Their accuracy differs significantly on datasets such as CBF, DPTW, PowerCons, and SmoothS. This may be due to the physical limitations of the circuits (and consequently their distinct computational models).

5 CONCLUSION

With the development of next-generation electronics, PE, including pNCs, have gained substantial research attention, as they can offer unique features such as flexibility and bio-degradability, that can not be matched by conventional silicon-based electronics. However, the existing pNCs, consisting of only printed resistors and transistors, are unable to process temporal sensory inputs, due to the absence of components with time dependencies. This limitation significantly hinders the utility of pNCs in domains where temporal information needs to be processed. To address this problem, this work proposed a printable pTPB with learnable filters. By modeling the proposed circuit with the consideration of circuit coupling and designing a learning objective, we enable the proposed pTPNC to handle time-series process. Simulations performed on 15 benchmark time-series datasets reveal that pTPNCs rectify the inherent inability of previous pNCs to process time-related signals, while having only 1.5× higher device counts and 1.3× increased power consumption. Moreover, through the comparison with Elman RNNs, we show that the proposed pTPNCs achieve an almost comparable (98%) classification accuracy to hardware-agnostic RNNs.

ACKNOWLEDGMENTS

This work is partially supported by the Carl-Zeiss-Foundation as part of "stay young with robots" (JuBot) and the European Research Council (ERC).

REFERENCES

[1] Arif U Alam et al. 2021. Fruit Quality Monitoring with Smart Packaging. *Sensors* 21, 4 (2021), 1509.
[2] Leonardo Weiss Ferreira Chaves and Christian Decker. 2010. A Survey on Organic Smart Labels for the Internet-of-Things. In *2010 Seventh International Conference on Networked Sensing Systems (INSS)*. IEEE, 161–164.
[3] Zheng Cui. 2016. *Printed Electronics: Materials, Technologies and Applications*. John Wiley & Sons.
[4] Hoang Anh Dau et al. 2018. The UCR Time Series Classification Archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.
[5] Jeffrey L Elman. 1990. Finding Structure in Time. *Cognitive science* 14 (1990).
[6] Matt W Gardner and SR Dorling. 1998. Artificial Neural Networks (The Multilayer Perceptron) - A Review of Applications in the Atmospheric Sciences. *Atmospheric environment* 32, 14-15 (1998).
[7] Steven D. Gardner et al. 2022. An Inkjet-Printed Artificial Neuron for Physical Reservoir Computing. *IEEE Journal on Flexible Electronics* 1, 3 (2022), 185–193.
[8] Suresh Kumar Garlapati et al. 2015. Ink-Jet Printed CMOS Electronics from Oxide Semiconductors. *Small* 11, 29 (2015), 3591–3596.
[9] Alex Graves et al. 2009. A Novel Connectionist System for Unconstrained Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 5 (2009), 855–868.
[10] Heikki Hyötyniemi. 1996. Turing Machines are Recurrent Neural Networks. *Proceedings of step* 96 (1996).
[11] Altynay Kaidarova et al. 2019. Wearable Multifunctional Printed Graphene Sensors. *NPJ Flexible Electronics* 3, 1 (2019), 1–10.
[12] Isidoro Ibanez Labiano et al. 2021. Flexible Inkjet-Printed Graphene Antenna on Kapton. *Flexible and Printed Electronics* 6, 2 (2021), 025010.
[13] Jiameng Li et al. 2022. Micro and Nano Materials and Processing Techniques for Printed Biodegradable Electronics. *Materials Today Nano* 18 (2022).
[14] Morten Mikolajek et al. 2019. Fabrication and Characterization of Fully Inkjet Printed Capacitors Based on Ceramic/Polymer Composite Dielectrics on Flexible Substrates. *Scientific reports* 9, 1 (2019), 13324.
[15] Robert A Nawrocki et al. 2014. Neurons in Polymer: Hardware Neural Units Based on Polymer Memristive Devices and Polymer Transistors. *IEEE Transactions on Electron Devices* 61, 10 (2014), 3513–3519.
[16] Adam Paszke et al. 2019. Pytorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035.
[17] Farhan Rasheed et al. 2018. Variability Modeling for Printed Inorganic Electrolyte-gated Transistors and Circuits. *IEEE transactions on electron devices* 66 (2018).
[18] Catherine D Schuman et al. 2017. A Survey of Neuromorphic Computing and Neural Networks in Hardware. *arXiv preprint arXiv:1705.06963* (2017).
[19] Abu Sebastian et al. 2020. Memory Devices and Applications for In-Memory Computing. *Nature nanotechnology* 15, 7 (2020), 529–544.
[20] Qizeng Sun et al. 2022. Smart Band-Aid: Multifunctional and Wearable Electronic Device for Self-Powered Motion Monitoring and Human-Machine Interaction. *Nano Energy* 92 (2022), 106840.
[21] Pauliina Vilmi et al. 2016. Fully Printed Memristors for a Self-Sustainable Recorder of Mechanical Energy. *Flexible and Printed Electronics* 1, 2 (April 2016), 025002.
[22] Dennis D Weller et al. 2020. Programmable Neuromorphic Circuit based on Printed Electrolyte-Gated Transistors. In *2020 Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 446–451.
[23] Dennis D Weller et al. 2021. Realization and Training of an Inverter-Based Printed Neuromorphic Computing System. *Scientific reports* 11 (2021).
[24] Haibin Zhao et al. 2022. Printed Electrodermal Activity Sensor with Optimized Filter for Stress Detection. In *International Symposium on Wearable Computers, Atlanta, GA and Cambridge, UK, September 11-15, 2022*.
[25] Haibin Zhao et al. 2023. Highly-Bespoke Robust Printed Neuromorphic Circuits. In *Design, Automation & Test in Europe Conference & Exhibition*. IEEE.
[26] Haibin Zhao et al. 2023. Highly-Dependable Printed Neuromorphic Circuits Based on Additive Manufacturing. *Flexible and Printed Electronics* 8, 2 (2023), 025018.