Neural Architecture Search for Highly Bespoke Robust Printed Neuromorphic Circuits

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

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

Tara Gheshlaghi
tara.gheshlaghi@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

1 INTRODUCTION

Despite the continual advancement in terms of power efficiency and transistor density, silicon-based electronics can hardly be introduced into many consumer edge applications, which includes smart packaging [1], smart bandages [2, 3], wearables or other disposable electronics [4–7] for consumer products. In such edge scenarios, devices are required to be flexible, customized, bio-compatible, and on-demand fabricated, with manufacturing costs expected to be under a few cents. However, these properties pose substantial challenge to silicon-based VLSI due to limitations such as bulky substrates and highly complicated lithography-based processing.

In this regard, printed electronics (PE) stands out as an adaptable and cost-effective alternative for such applications. The technology’s standout characteristic lies in its ability to provide bespoke application-specific customization requirements, regardless of high or low volumes, owing to its low-cost nature of the additive printing process over conventional lithography-based fabrication.

The main benefit of additive printing process is the significant cost reduction achieved through maskless manufacturing. However, there are also intrinsic drawbacks in this printing methods: Due to the limited operational precision of the printing equipments and the dispersion of ink droplets [8], the geometric (thus electronic) features of the printed components inevitably deviate from their designed values, leading to greater variability in the printing process compared to the high-resolution lithography-based methods. Meanwhile, bespoke architectures used in the realization of printed neuromorphic circuits, along with the use of analog computing [9] or digital approximation [10], can drastically affect the vulnerability of these printed circuits to variation. While the variation tolerance of generic (model-agnostic) neural network (NN) hardware accelerators and other machine learning models has been extensively investigated [9, 11, 12], this aspect remains highly unexplored for custom hardwired (bespoke) activation functions (AFs) in printed analog neuromorphic circuits (pNCs) commonly used in PE.

Thus, many efforts, from manufacturing technology to algorithmic circuit design, have been devoted to mitigating this issue. In this
context, our work have adopted a variation-aware evolutionary algorithm (EA), which utilizes various learnable AF circuits to achieve a variation-robust pNC. In short, the contributions are:

1. This work proposes the design of printed bespoke learnable AF circuits at both circuit and algorithmic level.
2. This research, for the first time, introduces an innovative EA approach to optimize pNCs, capable of not only tuning the AF circuits’ component values but also selecting the most appropriate type of AF for each neuron per dataset to mitigate printing error while performing Neural Architecture Search (NAS) and training crossbar conductances.

In experiment based on SPICE simulation, the proposed method demonstrated an improvement in the normalized classification error rate by 55.38% and 25.11% under ±5% and ±10% variation respectively compared to the baseline. Additionally, it achieved = 21% higher robustness when ReLU family is selected as the AF.

The rest of this paper is structured as follows: Sec. 2 introduces PE, pNC, and other preliminaries. Sec. 3 describes the design of learnable AF circuits and formulation of variation-aware EA. In Sec. 4, the proposed approach is evaluated with extensive simulations. Finally, Sec. 5 summarizes this work.

2 PRELIMINARIES

2.1 Printed Electronics

PE represents a rapidly evolving electronic fabrication technology poised to revolutionize domains like wearables, smart sensors, and the Internet of Things (IoT) [4–6]. Different from traditional silicon-based VLSI, PE focuses instead on lowering the production cost through maskless additive manufacturing at low processing temperatures. Due to abundant functional materials, such electronics boast unique advantages including mechanical flexibility, porosity, non-toxicity, and biodegradability outperforming conventional lithography-based silicon electronics.

Optimal device performance in PE is typically attained with vacuum-deposited, highly purified molecular substrates [13]. However, solution-based fabrication techniques, like spin-coating and inkjet printing, have gained enough attention for their potential to optimize manufacturing efficiency and reduce costs [14]. Printing technologies (as shown in Fig. 1) are generally divided into two main categories: (i) replication printing, exemplified by gravure printing (Fig. 1 (a)), which is optimized for high-throughput manufacturing, and (ii) jet printing, with inkjet printing (Fig. 1 (b)) serving as a key paradigm, for the bespoke fabrication of electronic circuits in smaller quantities. PE utilizes either additive or subtractive manufacturing methods. As illustrated in Fig. 1 (d), subtractive processes involve alternating deposition and etching, and are typically costlier due to specialized equipment requirements. Additive processes, in contrast (Fig. 1 (e)), sequentially deposit materials to form electronic components, as seen in the inkjet printing process. Despite the lower resolution and increased variability of additive methods, they offer substantial cost advantages.

Current research in inorganic PE is focused on the development of N-type Electrolyte-Gated Transistors (n-EGTs) (as shown in Fig. 1 (c)), where the band structure of metal oxides enables significantly higher electron mobility, allowing these n-EGTs to function efficiently at sub-1V supply voltages [15].

However, additive manufacturing also introduces challenges to PE, such as low device counts, large device dimensions, and high variability. Hence, PE does not target to compete with silicon-based devices in terms of performance within VLSI applications, but rather to complement them in resource-limited and cost-sensitive edge computing areas, like disposable electronics or wearable devices. In these scenarios, computational tasks are typically simple and have a relatively high tolerance to compute imprecision and thus require only small-scale circuits. Nonetheless, the high printing variation caused by ink dispersion [8] and printer inaccuracy pose a substantial influence on printed computing systems.

2.2 Printed Analog Neuromorphic Circuits

With the progression of artificial intelligence, neuromorphic computing has emerged as an effective approach for solving complex and nonlinear tasks. Small NNs, as enabled by pNCs, are sufficiently powerful enough for real-time and sensor data processing, without requiring vast computational power and resources associated with larger NN architectures commonly used in deep learning. This compatibility is particularly beneficial for various PE target applications, such as wearable sensors, flexible displays, and smart bandages [4–6], having specific resource constraints and functional requirements. Their capacity for on-demand fabrication facilitates the embedding of pNCs directly onto flexible substrates—something unattainable with conventional silicon-based hardware, thus making them an ideal candidate for applications requiring low device counts and bespoke customized configurations.

Notably, this computational capacity generally requires only basic operations such as weighted-sum and nonlinear activation. Therefore, this streamlined but efficient computing paradigm becomes attractive for circuit design, as desired functionalities can be achieved by the interconnection of simple circuit primitives. In the context of pNCs, by printing appropriate resistor crossbars (for weighted-sum) and nonlinear transformation circuits (for AFs), pNCs can achieve bespoke design for the computational requirements of the target applications of PE.

2.2.1 Hardware Primitives. Fig. 2 exemplifies the circuit schematics of a neuron in printed analog neuromorphic circuits. Fig. 2(a) shows a neuromorphic circuit resembling an artificial neural network (ANN) with a 6-4-3 topology. Fig. 2(b) show the schematic of a
printed neuron. Some negative weight circuits (Fig. 2 (c)) are also incorporated in case required.

Resistor Crossbars. The left part of Fig. 2(b) shows the circuit schematics of a resistor crossbar in printed neuron. Following Ohm’s Law and Kirchhoff’s Laws, the resulting output voltage of the crossbar $V_z$ can be calculated as

$$V_z = \frac{g_1}{G}V_1 + \frac{g_2}{G}V_2 + \frac{g_3}{G}V_3 + \frac{g_0}{G}V_b,$$  \hspace{1cm} (1)

where $g_i$ signifies the conductances of the resistors $R_i$ and $G$ represents the aggregate conductance $\sum_i g_i + g_0 + g_4$. Evidently, the output voltage $V_z$ is the weighted-sum of the input voltages $V_i$, with the weights embodied by the ratio of conductance values between $g_i$, $g_0$, and $G$. So, analogous to training ANNs, by designing and printing suitable conductances, desired weights can be achieved.

Printed negative weight circuit. As the conductances in the crossbar resistor array are limited to representing only positive weights, certain resistors, as shown in Fig. 2(b), are augmented with inverter-based circuits to introduce negative weight capabilities, as shown by the detailed circuit schematics in Fig. 2(c). This facilitates the emulation of multiplication operation with negative weights through the inversion of input voltages. The transfer characteristic of these negative weight circuits is characterized by a modified negative tanh function.

$$\text{neg}(V_z) = -\left(\frac{\eta_i^N}{G} + \frac{\eta_j^N}{G} \cdot \tanh \left(\frac{V_z - \eta_3^N}{\eta_4^N}\right)\right),$$  \hspace{1cm} (2)

where $\eta_i^N = [\eta_1^N, \eta_2^N, \eta_3^N, \eta_4^N]$ are auxiliary parameters that modify the original tanh function, which is ultimately determined by the physical quantities $q^N = [R_1^N, R_2^N, R_3^N, W_1^N, W_2^N, W_3^N, L_1^N]$ in the circuit. Here, $W_1^N$ and $L_1^N$ are geometric features (width and length) of the transistor $T_i^N$. Here, the superscript $(\cdot)^N$ denotes the variables in the negative weight circuit. Moreover, by optimizing $q^N$, the shape of negative weight function can be tuned to better fit specific target tasks. A surrogate circuit model for training $q^N$ was introduced in [9].

Printed activation circuits. Following the crossbar, the signals are passed through a printed activation circuit to resemble the AFs in ANNs. The circuit diagram, in Fig. 2(c), is taken from [9]. Analogous to the negative weight circuit, the characteristic curve of the printed tanh (p-tanh) activation circuit can be represented by a parameterized tanh function, specifically,

$$V_z = \text{ptanh}(V) = \eta_1^T + \eta_2^T \cdot \tanh \left(\left(V - \eta_3^T\right) \cdot \eta_4^T\right),$$  \hspace{1cm} (3)

with the auxiliary parameters $\eta^T = [\eta_1^T, \eta_2^T, \eta_3^T, \eta_4^T]$ determined by $q^T = [R_1^T, R_2^T, W_1^T, L_1^T]$. Similarly, $q^T$ can also be trained to fit specific target tasks. Also, the superscript $(\cdot)^T$ denotes the variables in the tanh circuits.

In addition to this circuit, we designed three additional activation circuits mimicking other AFs, i.e., a printed sigmoid (p-sigmoid) circuit, a printed clipped ReLU (p-clipped ReLU) circuit, and a printed ReLU (p-ReLU) circuit. With these novel designs, we explore their abilities in improving the circuit robustness with respect to printing variations. Detailed descriptions are introduced in Sec. 3.

2.3 Algorithmic Level Design and Optimization

The additive manufacturing process enables the capability of PE to engineer components with predetermined values through the specification of their geometric attributes. Thus, by printing customized conductance values in the resistor crossbar array, which correspond to the requisite synaptic weights in ANNs, pNCs are allowed to have bespoke design for target computational functionalities. Exploiting this capability of pNCs necessitates precise modeling of the circuit architecture and hardware-aware training process.

Modeling of Printed Neuromorphic Circuit. Based on printed neurons described in Sec. 2.2, the mathematical model of a printed neuron, taking p-tanh as the example for activation circuit, is given by

$$\text{ptanh}_i \left[ \sum_j w_j \left(V_j \cdot \mathbb{I} \{\theta_j > 0\} + \text{neg}_q(V_j) \cdot \mathbb{I} \{\theta_j < 0\} \right) \right],$$  \hspace{1cm} (4)

where $\theta_i$ is the learnable parameter encoding conductance by $g_i = |\theta_i|$ and the presence of a negation circuit via its sign. Moreover, the indicator function $\mathbb{I}(\cdot)$ returns 1 if its condition is true, else 0. Additionally, the weights $w_i$ are given by

$$w_i = \frac{|\theta_i|}{\sum_j |\theta_j|}.$$

Note that, $q^T$ and $q^N$ are also factors that can influence the output of the printed neuron. To facilitate their training, a NN-based surrogate model is employed, transforming the activation
circuit parameters to their corresponding $\eta$. For training, these surrogate models are used to approximate the complex mappings of the learnable circuits [9].

Gradient-Based Optimization. Gradient-based learning with backpropagation [16] forms the backbone of training modern ANNs. However, straightforward gradient-based optimization is unable to handle problems that encompass differentiable operations. Additionally, some functions do not provide useful update information through their gradient, an example being piece-wise constant functions. Another example that relates to this work is that the type of AF in Eq. (4) cannot be selected via gradient-based methods, as it amounts to a discrete decision. While the topic of NAS has received increasing attention, existing methods [17, 18] may be beneficial for optimizing circuit topologies but have not been adapted for pNCs yet.

Evolutionary-Based Optimization. EAs draw inspiration from natural selection and biological evolution. These algorithms utilize operations such as crossover and mutation to evolve candidate solutions over time, thereby optimizing them in an incremental fashion. One of the key benefits of EAs is their high degree of versatility and adaptability; for instance, they do not require the problem at hand to be differentiable. This characteristic makes EAs particularly suitable for a wide range of problems that are otherwise challenging for traditional optimization methods.

Although it is generally known that EAs may not match the efficiency of gradient-based optimization methods, particularly in the context of large-scale problems, this limitation is less significant when it comes to the design and optimization of small-scale NNs. Such NNs are often found in many computing applications where the flexibility and the computational resources are limited. Fortunately, these applications are exactly the target domains of the PE. Moreover, as EAs can be parallelized efficiently, their computational time can be substantially reduced. In sum, despite their perceived inefficiency in large-scale scenarios, the practical use of EAs in optimizing small-scale PE for target applications devices underscores their relevance.

2.4 Related Work

Despite the unique advancements of the printing technology and PE, the nature of additive manufacturing process introduces multiple sources of variation in printed components, such as ink dispersion on substrates, irregularities in droplet jetting, satellite droplets, and missing droplets [8, 19]. These variations are sometimes modeled by a uniform distribution of the electrical characteristics of printed devices with in the printed resolution, specifically the parameters of the electrolyte-gated transistors (n-EGTs) and is addressed by a Gaussian Mixture Model at the device level [20, 21].

Efforts from multiple aspects has been paid to mitigate the impact of printing variations, including quality control engineering [22] and material science [23]. From the algorithmic level, advanced design strategies that incorporate variation-aware training [9, 24] are employed. These strategies involve designing circuits with robustness [25–27] in mind, ensuring that they can tolerate a certain degree of variation without any performance degradation.

In this work, we focus on device variability in PE and analog computing by using an EA. In short, our research aims to bridge this gap of variation-aware training by developing not only bespoke pNCs, but also more resilient bespoke AF circuits that maintain high robustness even with the inherent additive manufacturing printing variation.

3 DESIGN AND METHODOLOGY

In the following, we will provide a detailed description of learnable AFs and their transfer characteristic curves based on our circuit design in PE. Fig. 2 (d, e, f) depicts the schematic of different learnable activation circuits, i.e., printed sigmoid (p-sigmoid), printed clipped ReLU (p-clipped ReLU) and printed ReLU (p-ReLU) design.

3.1 Learnable Activation Function (AF) Circuits

Printed Sigmoid Circuit. Similar to the learnable p-tanh AF circuit design in Fig. 2 (c), the p-sigmoid AF can also be obtained from this configuration by replacing the supply voltage $V_{SS} = 0$ and adding a small 100 $\Omega$ resistance as shown in Fig. 2 (d) and can be modeled by a suitable mathematical equation with its own distinct parameters.

**Table 1: FEASIBLE DESIGN SPACE OF P-SIGMOID CIRCUIT**

<table>
<thead>
<tr>
<th>Range</th>
<th>$R_S^1$ (kΩ)</th>
<th>$R_S^2$ (kΩ)</th>
<th>$W_{CR}^1$ (μm)</th>
<th>$L_{CR}^1$ (μm)</th>
<th>$W_{CR}^2$ (μm)</th>
<th>$L_{CR}^2$ (μm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimal</td>
<td>350</td>
<td>40</td>
<td>80</td>
<td>80</td>
<td>500</td>
<td>40</td>
</tr>
<tr>
<td>maximal</td>
<td>750</td>
<td>80</td>
<td>600</td>
<td>200</td>
<td>800</td>
<td>80</td>
</tr>
</tbody>
</table>

The equation for a printed sigmoid function is given by

$$V_a = \eta_1^S + \eta_2^S \cdot \text{sigmoid} \left( \left( V_z - \eta_3^S \right) \cdot \eta_4^S \right),$$

where $\text{sigmoid}(\cdot)$ function is defined by

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}},$$

where $\eta^S = [\eta_1^S, \eta_2^S, \eta_3^S, \eta_4^S]$ are auxiliary parameters determined by the physical quantities $q^S = [R_S^1, R_S^2, W_{CR}^1, L_{CR}^1, W_{CR}^2, L_{CR}^2]$ in the circuit. Adjusting these parameters allows to adapt the shape of the AF within the printed circuit.

Printed Clipped ReLU Circuit. A clipped ReLU activation circuit uses only one transistor and one resistor in series as shown in Fig. 2 (e) and extends the basic ReLU by imposing an upper bound to the activation. The output voltage, $V_a$, is constrained between 0 and a predefined maximum value $V_{max}$.

**Table 2: FEASIBLE DESIGN SPACE OF P-CLIPPED RELU CIRCUIT**

<table>
<thead>
<tr>
<th>Range</th>
<th>$R_{CR}^1$ (MΩ)</th>
<th>$W_{CR}^1$ (μm)</th>
<th>$L_{CR}^1$ (μm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimal</td>
<td>1</td>
<td>40</td>
<td>80</td>
</tr>
<tr>
<td>maximal</td>
<td>10</td>
<td>100</td>
<td>200</td>
</tr>
</tbody>
</table>

The mathematical representation of the clipped ReLU is:

$$V_a = \begin{cases} V_z, & V_z < \eta_{CR}^1 \\ \eta_{CR}^2 - \eta_{CR}^1, & \eta_{CR}^1 < V_z < \eta_{CR}^2 \\ \eta_{CR}^4 - \eta_{CR}^3, & \eta_{CR}^3 < V_z < \eta_{CR}^4 \\ \frac{\eta_{CR}^4 - \eta_{CR}^3}{\eta_{CR}^4 - \eta_{CR}^3} \cdot V_z + \frac{\eta_{CR}^3 \cdot \eta_{CR}^4 - \eta_{CR}^1 \cdot \eta_{CR}^4 - \eta_{CR}^2 \cdot \eta_{CR}^3}{\eta_{CR}^4 - \eta_{CR}^3}, & \text{otherwise} \end{cases}$$

$$\eta_{CR}^1 = \eta_{CR}^3 = \frac{\eta_{CR}^1 + \eta_{CR}^2}{2}, \quad \eta_{CR}^2 = \eta_{CR}^3 = \frac{\eta_{CR}^1 + \eta_{CR}^2}{2}, \quad \eta_{CR}^4 = \eta_{CR}^3 = \frac{\eta_{CR}^1 + \eta_{CR}^2}{2}.$$
Figure 3: Overview of the EA-based training of printed neuromorphic circuits (pNCs). (a) Genes that encode nodes and edges. (b) Crossover from the parent genomes to offsprings. (c) Mutation of the topology and learnable parameters.

3.2 Evolutionary Architecture Search

We propose an innovative EA inspired by the classical NeuroEvolution of Augmenting Topologies (NEAT) [30] to design highly bespoke pNC. The proposed algorithm not only tunes the component values within the AF circuits but also selects the most appropriate type of AF per layer per dataset to mitigate the effects of printing variation and outperform the error rate. In this problem statement, the parameters have certain constraints; conductances in the crossbar resistor arrays are limited to representing only positive weights. EAs are more convenient for considering the constraints on learnable parameters and are also suitable for non-differentiable problems, unlike gradient-based methods. As designing the circuit topology (i.e., the neural architecture) and selecting the optimal activation functional forms represent a discrete decision space, we can leverage the ability of EAs for such discrete problems.

We strategically encode the printed pNCs so that the circuit topology are jointly optimized during evolution, leading to neural architecture search. Additionally, the algorithm trains the crossbar conductances (i.e., weights) and optimizes the type of AF circuits for each printed neuron per dataset, along with the learnable parameters \( q \) in the AF circuits. With this enhanced search space, the resulting pNCs are expected to be more robust against printing variation, leading to a reduction in the post-printing accuracy degradation compared to those trained by gradient methods, where only crossbar conductances and parameters in AF circuits are learned.

The key components of the proposed algorithm are shown in Fig. 3. Each genome represents a pNCs and comprises two types of genes encoding the circuit specifications: node genes (Fig. 3(a-1)) and edge genes (Fig. 3(a-2)). Initially, a population of genomes is created and categorized into multiple species based on node and edge similarities. The number of offspring for each species is determined based on its mean fitness. Top-performing genomes are preserved unchanged for the next generation, while others undergo crossover and mutation. When the termination criteria are met, the most optimized solution is obtained according to the defined fitness function, as will be discussed in Sec. 3.3. In the following,
we explain how the genome encoding, as well as the processes of crossover and mutation for the proposed algorithm, are adjusted.

Encoding. The node gene represents a neuron and consists of learnable parameters, including resistors $R_0$ and $R_0$ and learnable parameters of the candidate AF circuit and the negative weight circuit, i.e., $q$. Additionally, it also contains a learnable and discrete variable that determines which candidate activation circuit is selected to connect to the crossbar output. Each node gene is uniquely identified by a global index. On the other hand, the edge gene denotes connections between neurons and comprises a learnable resistance $R$ in the crossbar for weights, along with a learnable boolean parameter indicating the connectivity state (enabled/disabled) for circuit topology. Each edge gene is distinguished by the indices of the connected nodes, and all edges are directional, meaning $(i, j) \neq (j, i)$. Here, we denote the set of genes from all the genomes in the population as $\mathcal{F}$.

Crossover. In the proposed algorithm, unique genes (not shared between parents) of the fitter parent are directly inherited by the offspring, as illustrated by node 6 in the offspring from parent 1 in Fig. 3(b). For common genes, crossover involves the random inheritance of genes from the shared topological structure (genes with identical global index) of both parent genomes, as demonstrated in Fig. 3(b) for node 3, which is common between the two parents. During crossover, the features of each gene are exchanged randomly between the parents, with a higher probability of selection for the fitter parent. Subsequently, the mutation process is applied to the offspring.

Mutation. Mutation, as depicted in Fig. 3(c), is a two-stage process involving genome-level mutation for neural architecture and gene-level mutation for the parameters related to node and edge genes. At the genome-level, mutations can involve either the addition (Fig. 3(c-1)) or deletion (Fig. 3(c-2)) of edges between existing nodes, and similarly, nodes can be added (Fig. 3(c-3)) or deleted (Fig. 3(c-4)) at existing edges. When deleting a edge, a random edge is selected. Before deleting a node, edges associated with that node are removed to prevent disruption. Importantly, to avoid the extinction of new genomes, new structures should not influence genome fitness. Therefore, to maintain unchanged circuit output (and thus performance), the conductance of new edges should be initialized to zero when adding edges. Additionally, when adding a node to an edge, as shown in Fig. 3(c-3), the existing edge is disabled (not deleted) and the new node is introduced with two connections to replace said edge. To preserve the output, the conductance on edge $(k, j)$ should be initialized by that of the edge $(i, j)$, whereas the output of the node $k$ should be the same as that of node $i$. At the gene-level, mutation involves perturbing crossbar conductance $\theta$ and nonlinear circuit parameters $q$ by adding scaled samples from a normal distribution $p(\theta)$ and $p(q)$ respectively (Fig. 3(c-5)). The type of selected activation circuit mutates randomly among $[p\text{-sigmoid}, p\text{-tanh}, p\text{-ReLU}, p\text{-clipped_ReLU}]$ (Fig. 3(c-6)), while the state parameter of the edge is determined by a Bernoulli variable (Fig. 3(c-7)).

3.3 Variation-aware Training with NAS

Variation-aware training is critical in optimizing the reliability and performance of pNCs, which often suffer from intrinsic printing variations. We therefore integrate the proposed NAS to dynamically adjust the design of pNCs, ensuring that they not only meet desired classification accuracy, but also demonstrate resilience to printing variations. This methodology not only enhances the adaptability of pNCs but also their usefulness in real scenarios, where variations are a critical concern.

In this framework, all parameter corresponding to printed resistances, i.e., $\theta$ and transistors, i.e., $q$, are subject to process variation arising from ink dispersion on the substrate, droplet jetting oddness and satellite drops wetting [31]. Each printing/processing step of the resistances and the n-EGTs (channel, dielectric, and top-gate) introduces variations resulting in non-Gaussian distributions for both the process and electrical parameters of this technology[32]. Consequently, these parameters are modeled as random variables to account for the inherent printing variations, i.e., $\theta \sim p(\theta)$ and $q \sim p(q)$ respectively. These variables adhere to their respective probability distributions to mirror potential deviations arising from the printing process. To evaluate the robustness of different architectures against these variabilities and thus guide the evolution process, the expected loss with respect to parameter variation is used as the training objective, namely

$$\text{minimize } L = \mathbb{E}_{\theta, q} \left( L(\theta, q, D) \right),$$

$$= \int_{\theta} \int_{q} L(\theta, q, D) p(\theta) p(q) \, d\theta \, dq,$$

where $D = \{x, y\}$ refers to the target datasets, while $L(\cdot)$ refers to the cross-entropy loss [33], which is commonly used to improve classification accuracy. However, Eq. (8) poses a challenge that the optimization variable $\theta$ and $q$ will be integrated out. To facilitate the training of these parameters, we introduce a reparameterization strategy [34] to decouple the learnable parameters from the random variables expressing the variation. Consequently, $\theta = \theta_0 \otimes \epsilon_\theta$ and $q = q_0 \otimes \epsilon_q$, where $\theta_0$ and $q_0$ denote target values to be optimized, while each element in $\epsilon_\theta$ and $\epsilon_q$ follows a distribution $p(\epsilon)$ respectively. With this approach, the training objective can be re-formulated as

$$\text{minimize } L = \mathbb{E}_{\epsilon_\theta, \epsilon_q} \left\{ L(\theta_0 \otimes \epsilon_\theta, q_0 \otimes \epsilon_q, D) \right\},$$

$$= \int_{\epsilon_\theta} \int_{\epsilon_q} L(\theta_0 \otimes \epsilon_\theta, q_0 \otimes \epsilon_q, D) p(\epsilon_\theta) p(\epsilon_q) \, d\epsilon_\theta \, d\epsilon_q,$$

With Eq. (9), the parameters $\theta_0$ and $q_0$ can be trained to guarantee the optimal classification accuracy under the expectation of given variation $p(\epsilon_\theta)$ and $p(\epsilon_q)$. However, the integration in Eq. (9) still has no closed form, which poses challenge to its optimization. For this, we employ an estimation of the integration through Monte-Carlo sampling, i.e.,

$$L \approx \frac{1}{N} \sum_{n=1}^{N} L(\theta_0 \otimes \epsilon'_n, q_0 \otimes \epsilon'_n, D),$$

$$\text{where } N \text{ is the number of samples.}$$
where $e'_\theta \sim p(\epsilon_{\theta})$ and $e'_v \sim p(\epsilon_v)$ are samples drawn from their respective distribution in each calculation of $L$. Moreover, $N$ denotes the number of samples utilized to estimate the integration.

Finally, Eq. (10) is utilized within the training objective (i.e., fitness function $f(x)$) of the variation-aware training. For the training objective, we also consider the expected classification accuracy ACC, i.e.,

$$f(x) = ACC - L.$$  

(11)

With this objective, NAS aims not only to improve the classification accuracy of pNCs, but also improve the robustness against intrinsic stability with respect to variations. This process involves an assortment of optimization techniques, which may extend beyond conventional gradient-based methods, to seek out architectures that guarantee reliable performance despite the unpredictable nature of the printing process. The ultimate aim of using NAS in this context is to strike an optimal balance between performance and resilience, ensuring proper operation in real scenarios.

4 EVALUATION

To assess the effectiveness of the proposed method, we utilized PyTorch to implement the algorithm\(^1\) and carried out experiments on 13 benchmark datasets. These datasets are also used in other state-of-the-art studies on pNCs [12, 35], and match the complexity within the application domains of PE.

4.1 Experiment Setup

We conduct training on the pNC utilizing the EA methodology and test it on 13 benchmark datasets against the established gradient-based optimization techniques as a baseline of this work.

Circuit Setup. The AF circuits in Fig. 2 (top (e), (f), (g)) were designed based on the well-developed n-EGT P-PDK [20] and the ranges of learnable parameters are determined by performing sweep analysis. We used Cadence Virtuoso\(^2\) tool to simulate the transfer characteristics (as shown in Fig. 2 (bottom (e), (f), (g))) in SPICE.

Initialization. Drawing insights from other works on EA and guided by a series of preliminary trials, we have strategically initialized the network topologies for all datasets as unconnected networks, which consist solely of nodes corresponding to the number of outputs, featuring only #output nodes. The population for these experiments is robustly set at 1,000 individuals. Each node is initialized to have a random AFs circuit among the given designs.

In terms of the mutation mechanisms employed, we have defined specific probabilities for the genetic alterations within the network structures: the probability of introducing either a new node or a new connection is set at a substantial rate of 0.7, while the probability for the deletion of a node or a connection is comparatively lower, at 0.3. Moreover, there exists a 0.1 chance that any given edge within the network will toggle its state from enabled to disabled, or vice versa, as part of the mutation process. Moreover, the mutation rate of changing selected AFs circuit is 0.1.

In terms of variation, we take uniform distribution, i.e., $\epsilon_{\theta} \sim U[1-\epsilon, 1+\epsilon]$ and $\epsilon_v \sim U[1-\epsilon, 1+\epsilon]$, to reflect the printing variation, because the printing variation is primarily determined by the geometric variation of the printing shape which varies within one printing pixel. More specific, we select an $\epsilon = 5\%$ to simulate a relatively high printing precision, while another $\epsilon = 10\%$ to simulate a relatively low printing precision. This is because the typical printing resolutions range from 20 μm to 100 μm [19], whereas the component feature sizes in printed neuromorphic circuits are on the order of 1 mm [35]. Moreover, for Monte-Carlo sampling, we select $N = 20$ for numerical estimation of the integration, as it can already yield sufficiently precise estimation in our experiments.

Training. In training (evolution) process, we utilize a full-batch training, with termination upon a patience threshold of 100 generations. This specific criterion hinges on observing no significant improvement in the performance metrics on the validation dataset over the aforementioned span of generations. To ensure that our findings are statistically reliable and to mitigate the variability due to stochastic elements of the training process, we repeat the training sessions ten times for each value of $\gamma$, employing different random seeds for each session, varying from 1 to 10. This repetition ensures that we achieve sufficiently optimal and robust solutions.

Baseline. To conduct experiment with baseline approach, we perform training with topologies initialized as #input-3-#output. We use the Adam [36] optimizer with default parameterization to train parameters. We start with an initial learning rate of 0.1 and halve it after a 100-epoch patience. Additionally, the training

Table 4: Simulation Result and Runtime of gradient-based approach without variation and comparison with EA with baseline in (i) high precision printing (5% variation) and (ii) low-precision printing (10% variation) on 13 Benchmark Datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Reference accuracy (without variation)</th>
<th>High-precision printing (±5%)</th>
<th>Low-precision printing (±10%)</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reference accuracy</td>
<td>Baseline</td>
<td>EA</td>
<td>Baseline</td>
</tr>
<tr>
<td>Acute Inflammation</td>
<td>1.000 ± 0.000</td>
<td>1.000 ± 0.000</td>
<td>1.000 ± 0.000</td>
<td>0.999 ± 0.012</td>
</tr>
<tr>
<td>Balance Scale</td>
<td>0.902 ± 0.017</td>
<td>0.880 ± 0.004</td>
<td>0.896 ± 0.008</td>
<td>0.877 ± 0.008</td>
</tr>
<tr>
<td>Breast Cancer Wisconsin</td>
<td>0.971 ± 0.001</td>
<td>0.963 ± 0.008</td>
<td>0.966 ± 0.006</td>
<td>0.931 ± 0.039</td>
</tr>
<tr>
<td>Cardiotocography</td>
<td>0.879 ± 0.007</td>
<td>0.874 ± 0.004</td>
<td>0.857 ± 0.005</td>
<td>0.763 ± 0.002</td>
</tr>
<tr>
<td>Energy Efficiency (y1)</td>
<td>0.915 ± 0.019</td>
<td>0.889 ± 0.032</td>
<td>0.916 ± 0.026</td>
<td>0.847 ± 0.012</td>
</tr>
<tr>
<td>Energy Efficiency (y2)</td>
<td>0.894 ± 0.016</td>
<td>0.883 ± 0.023</td>
<td>0.891 ± 0.038</td>
<td>0.867 ± 0.026</td>
</tr>
<tr>
<td>Iris</td>
<td>0.965 ± 0.005</td>
<td>0.912 ± 0.034</td>
<td>0.923 ± 0.050</td>
<td>0.843 ± 0.045</td>
</tr>
<tr>
<td>Mammographic Mass</td>
<td>0.788 ± 0.003</td>
<td>0.782 ± 0.017</td>
<td>0.810 ± 0.018</td>
<td>0.766 ± 0.053</td>
</tr>
<tr>
<td>Pendigits</td>
<td>0.577 ± 0.054</td>
<td>0.554 ± 0.038</td>
<td>0.559 ± 0.039</td>
<td>0.548 ± 0.047</td>
</tr>
<tr>
<td>Seeds</td>
<td>0.891 ± 0.031</td>
<td>0.820 ± 0.034</td>
<td>0.851 ± 0.023</td>
<td>0.820 ± 0.041</td>
</tr>
<tr>
<td>Tic-Tac-Toe Endgame</td>
<td>1.000 ± 0.001</td>
<td>0.713 ± 0.012</td>
<td>0.765 ± 0.018</td>
<td>0.660 ± 0.017</td>
</tr>
<tr>
<td>Vertebral Column (2 cl.)</td>
<td>0.830 ± 0.007</td>
<td>0.716 ± 0.007</td>
<td>0.794 ± 0.004</td>
<td>0.661 ± 0.000</td>
</tr>
<tr>
<td>Vertebral Column (5 cl.)</td>
<td>0.811 ± 0.010</td>
<td>0.634 ± 0.086</td>
<td>0.791 ± 0.016</td>
<td>0.634 ± 0.075</td>
</tr>
<tr>
<td>Average</td>
<td>0.879 ± 0.013</td>
<td>0.809 ± 0.023</td>
<td>0.848 ± 0.019</td>
<td>0.786 ± 0.029</td>
</tr>
</tbody>
</table>

\(^1\)https://github.com/Neuromorphic/eNAS_learnable_selectable_INC.

we also trained the pNCs without any variation, i.e., after completing the training of all pNCs, we carefully select the different training configurations. Additionally, Fig. 4 illustrates the improvement in printing error rates. Also, Fig. 5 details the selection percentages of various learnable AF circuits, highlighting their preferences during the training process.

The mean, standard deviation of their accuracy and their corresponding runtime performances are presented in Tab. 4, showing a scalar average across all datasets for a clearer comparison of different training configurations. Additionally, Fig. 4. illustrates the improvement in printing error rates. Also, Fig. 5 details the selection percentages of various learnable AF circuits, highlighting their preferences during the training process.

process is stopped, when the learning rate was halved 10 times. Other setups are kept the same as its EA counterpart. To provide an upper bound classification accuracy of each dataset, we also trained the pNCs without any variation, i.e., ε = 0. Because the accuracy in this case can be seen as the theoretically highest achievable values. We denote the accuracy in this case as the reference accuracy.

4.2 Result

After completing the training of all pNCs, we carefully select the most optimal pNCs w.r.t to the random seed in each experimental setup based on their performance on the validation loss. These selected circuits are the ones designed for physical realization. Subsequently, we evaluate their performance on the test sets. In testing, all pNCs are tested and trained under the identical variations.

The mean, standard deviation of their accuracy and their corresponding runtime performances are presented in Tab. 4, showing a scalar average across all datasets for a clearer comparison of different training configurations. Additionally, Fig. 4 illustrates the improvement in printing error rates. Also, Fig. 5 details the selection percentages of various learnable AF circuits, highlighting their preferences during the training process.

5 CONCLUSION

Printed Electronics, owing to their distinctive features, are gaining significant attention for the evolution of next-generation electronics. In this realm, analog printed neuromorphic circuits are becoming increasingly popular due to their ability to provide customized computational functions at a minimal cost for target applications. However, the maskless additive manufacturing process increases variability, challenging consistent performance.

To enhance the robustness of printed neuromorphic circuits and mitigate the effects of printing variations, we propose an evolutionary algorithm to train printed neuromorphic circuits with selectable activation circuits and optimal neural architecture, including parameters like crossbar conductances and physical quantities in nonlinear circuits. This approach not only opens new optimization opportunities but also enhances robustness against printing variations. Future research may further explore parallel computing using evolutionary algorithms.

ACKNOWLEDGMENTS

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


Optimization.