PRINT-SAFE: Printed Ultra-Low-Cost Electronic X-Design with Scalable Adaptive Fault Endurance

PRIYANJANA PAL, Department of Computer Science-CDNC - Chair of Dependable Nano Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany
TARA GHESHLAGHI, Department of Computer Science-CDNC - Chair of Dependable Nano Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany
HAIBIN ZHAO, Department of Computer Science-Chair of Pervasive Computing Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany
MICHAEL HEFENBROCK, RevoAI GmbH, RevoAI GmbH, Karlsruhe, Germany
MICHAEL BEIGL, Department of Computer Science-Chair of Pervasive Computing Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany
MEHDI TAHOORI, Department of Computer Science-Chair of Dependable Nano Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany

The demand for next-generation flexible electronics in applications like smart packaging and smart bandages has driven the need for cost-effective solutions. Traditional silicon-based electronics struggle with high costs and rigidity, making them unsuitable for these emerging markets. In this regard, additive printed electronics (PE) offer a viable alternative with their flexibility and ultra-low-cost manufacturing. printed analog neuromorphic circuits (pNCs) are well-suited for these target applications, especially for classification tasks, as their low device count can efficiently meet the needs of the technology. However, low-cost additive manufacturing comes with higher defect rates, such as misprints, broken connections, and defective components, posing significant challenges to the reliability of printed circuits. This article presents a novel co-design of training algorithm and hardware for fault-tolerant pNCs using fault-aware training (FAT). The proposed method introduces a fault-tolerant version of printed nonlinear transformation circuits, combined with a bespoke training process that selects different types of printed activation functions (AFs) for different neurons to optimize both fault endurance and hardware costs. Experiments on benchmark datasets demonstrate an improvement in the accuracy of fault-tolerant (FT) pNCs from 62.1% to 79.4% under a 10% fault rate. Moreover, combining both normal and fault-tolerant versions of activation functions (AFs) using *gumble-softmax* distribution shows an acceptable accuracy drop with an average reduction in power and area of 54.5% and 6.54%, respectively, while reducing the training time significantly by 56.2%, compared to only using FT-AFs.

Priyanjana Pal and Tara Gheshlaghi contributed equally to this research.
This work has been supported by the European Research Council (ERC) (Grant No. 101052764).
Authors' Contact Information: Priyanjana Pal, Department of Computer Science - Chair of Dependable Nano Computing (CDNC), Karlsruhe Institute of Technology, Karlsruhe, BW, Germany; e-mail: priyanjana.pal@kit.edu; Tara Gheshlaghi, Department of Computer Science - Chair of Dependable Nano Computing (CDNC), Karlsruhe Institute of Technology, Karlsruhe, BW, Germany; e-mail: tara.gheshlaghi@kit.edu; Haibin Zhao, Department of Computer Science-Chair of Pervasive Computing Systems, Karlsruhe Institute of Technology, Karlsruhe, BW, Germany; e-mail: haibin.zhao@kit.edu; Michael Hefenbrock, RevoAI GmbH, RevoAI GmbH, Karlsruhe, BW, Germany; e-mail: michael.hefenbrock@revoai.de; Michael Beigl, Department of Computer Science-Chair of Pervasive Computing Systems, Karlsruhe Institute of Technology, Karlsruhe, BW, Germany; e-mail: michael.beigl@kit.edu; Mehdi Tahoori, Department of Computer Science-Chair of Dependable Nano Computing, Karlsruhe Institute of Technology, Karlsruhe, BW, Germany; e-mail: mehdi.tahoori@kit.edu.

ACM 1539-9087/2025/09-ART88
https://doi.org/10.1145/3758096

CCS Concepts: • **Hardware → Fault tolerance**; **Emerging technologies**; **Robustness**; **Flexible and printable circuits**; • **Computing methodologies → Neural networks**; **Machine learning approaches**; **Bio-inspired approaches**; • **Computer systems organization → Fault-tolerant network topologies**; **Redundancy;**

Additional Key Words and Phrases: Printed electronics, neuromorphic computing systems, electrolyte-gated transistors, redundancy, fault-aware training

ACM Reference Format:
Priyanjana Pal, Tara Gheshlaghi, Haibin Zhao, Michael Hefenbrock, Michael Beigl, and Mehdi Tahoori. 2025. PRINT-SAFE: Printed Ultra-Low-Cost Electronic X-Design with Scalable Adaptive Fault Endurance. *ACM Trans. Embedd. Comput. Syst.* 24, 5s, Article 88 (September 2025), 22 pages. https://doi.org/10.1145/3758096

1 Introduction

Despite significant advancements in power efficiency and transistor density, silicon-based electronics face challenges in being adopted for many consumer edge applications, such as smart packaging [1], smart bandages [2], wearables [3, 4], IoT [5–7], RFID tags and other disposable electronics [8, 9] (as shown in Figure 1[1]). These applications require devices to be flexible, customizable, biocompatible, and capable of being produced on demand, with manufacturing costs expected to stay below just a few cents. However, these requirements are difficult to meet with traditional silicon-based VLSI technology due to its bulky substrates and the complexity of lithography-based manufacturing processes. Printed Electronics (PE) emerges as a promising and cost-effective alternative for these applications. PE's key advantage is its ability to offer customized application-specific solutions, whether in high or low volumes. This adaptability is made possible by the low-cost, additive nature of the additive printing process, which contrasts with the more expensive and complex lithography-based fabrication used in traditional silicon electronics.

To enable basic signal processing tasks, such as classification in printed devices, various printed computing circuits are required [10, 11]. **Printed analog neuromorphic circuits** (**pNCs**), which use resistor crossbars and inverter-based AF circuitry, emulate artificial neural network (ANN) operations by processing analog sensory input directly, thereby eliminating the need for expensive analog-to-digital converters.

The primary benefit of additive printing processes is the significant cost reduction achieved through maskless manufacturing. However, these processes come with reduced process control, leading to a higher variability and defect rate during both manufacturing and runtime. Common defect mechanisms in PE include misprints, incomplete traces, broken or misaligned connections, crossovers, and material degradation [12], which can cause open-short circuits, unpredictable electrical behavior, or even complete non-functionality of the circuits as shown in Figure 2. In addition, issues such as ink smudging, delamination, or void formation can further reduce the reliability of printed circuits [13].

Given these challenges, it is very critical to ensure high manufacturing yield while maintaining reliability in PE during in-field operation. Furthermore, the bespoke architectures used in pNCs [10, 14–17] introduce additional layers of complexity, further increasing vulnerability to faults [18]. Therefore, cost-effective fault-endurant custom hardwired (bespoke) architectures are needed to address these inherent defects, ensuring that printed circuits continue to function reliably despite the presence of manufacturing defects [19, 20], allowing the system to remain operational, thereby maintaining its overall quality and robustness.

[1]Images are generated by the DALL-E AI tool.

Fig. 1. Target application domains of PE: (a) smart bandages, (b) smart food packaging, (c) smart fruit package, (d) RFID tags, and (e) smart milk carton.

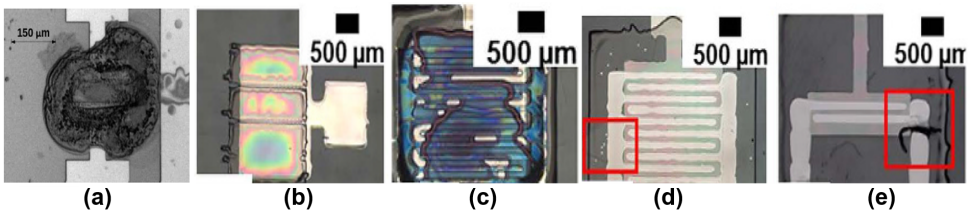

Fig. 2. (a) Transistor with exploded electrolyte, (b) inkjet-nozzle clog, (c) inhomogeneous layer formed, (d) satellite drops of ink, and (e) short circuit between S-D (sourced from [12, 19]).

Although significant efforts have been made to implement various pNCs [10, 15, 21], very few studies have focused on fault modeling and fault-tolerant FT design for printed ANN architecture [21]. In this context, our work focuses on the X-design (co-design) and modeling of FT nonlinear activation circuits, fault injection, and fault-aware training (FAT) in the neural network (NN) architecture.

In short, the contributions of this work are:

(1) This work, for the first time, proposes the X-design (co-design) of **FT printed bespoke** nonlinear transformation circuits at both the circuit and the algorithmic level.
(2) A gradient-based FAT approach is introduced to optimize pNCs in a bespoke manner. Using *Gumbel-Softmax distribution*, the most fault-tolerant AF is selected for each neuron, allowing differentiable backpropagation to dynamically adapt to printing defects.

The proposed FT-pNC demonstrates an accuracy improvement from 62.1% to 79.4% under a 10% fault injection compared to the baseline scenario (4-normal AFs). However, adopting only fault-tolerant activation functions (4-FT AFs) significantly increases hardware and training time overhead. To address these, we propose our combined FAT approach with 8-combined AFs, which integrates both normal and fault-tolerant versions and significantly reduces area, power and training time by $\approx$ 6.54%, $\approx$ 54.5%, and 56.2%, respectively, while maintaining an acceptable drop ($\approx$ 7%) in classification accuracy compared to using only 4-**fault-tolerant (FT)** versions.

The rest of this article is structured as follows: Section 2 introduces PE, pNC, fault models, and other preliminaries. Sections 3 and 4 describe the design and modeling of FT nonlinear transformation circuits and formulation of FAT. In Section 5, the proposed approach is validated with extensive simulations. Finally, Section 6 summarizes this work.

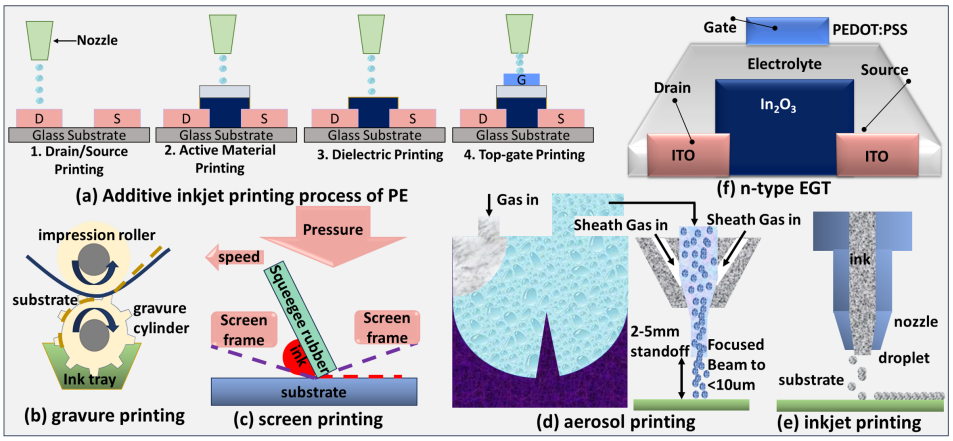

Fig. 3. Schematic of (a) additive printing process of PE, (b)–(e) types of printing processes; (f) front view of an inkjet-printed n-EGT.

2 Preliminaries

2.1 Printed Electronics (PE)

PE is a fabrication technology that utilizes printing methods such as inkjet, screen, or gravure printing [22]. This approach allows for the production of ultra-low-cost electronic circuits due to its simple additive manufacturing steps as shown in Figure 3(a) and low fabrication costs. In contrast, silicon-based processes require expensive foundries, even for older technology nodes [23]. Additionally, PE enables flexible electronics through its contactless printing techniques, such as inkjet printing, combined with advanced functional inks, conductive, semiconductive, and insulating materials. These inks are used to fabricate devices based on organic [24] or oxide-based materials [25].

Printing technologies are categorized into two primary types, each suited to specific applications and material needs: (i) contact printing and (ii) non-contact printing. Contact printing methods include: (a) replication printing, such as gravure printing as shown in Figure 3(b), which is optimized for high-volume production; and screen printing in Figure 3(c), which requires significant manufacturing time and costs while delivering relatively low resolution. On the other hand, non-contact printing includes: aerosol printing (Figure 3(d)), suitable for low-volume production due to its slower printing speeds and higher maintenance requirements; and Figure 3(e), jet printing, with inkjet printing being a key example, tailored for bespoke fabrication of small-batch electronic circuits.

As shown in Figure 3(a), the fabrication process of an N-type Electrolyte-Gated Transistor (n-EGT) begins with the printing of the drain (D) and source (S) electrodes directly onto a glass substrate using inkjet printing techniques. The active semiconductor material is then printed between these electrodes to facilitate charge transport. Following this, an electrolyte material, is deposited over the semiconductor to act as a gate dielectric, enabling voltage control. Finally, the gate electrode is printed on top of the dielectric layer. Although this streamlined additive process exhibits greater variability [21, 26], it reduces material waste, simplifies manufacturing [27], and is ideal for low-cost, flexible, and energy-efficient electronics [1, 2, 28].

Most state-of-the-art inkjet-printed **field-effect transistors** (**FETs**) often use organic semiconductors, but their low field-effect mobility and high operating voltages make them unsuitable for low-power applications in PE powered by energy harvesters or small batteries [29]. As shown in

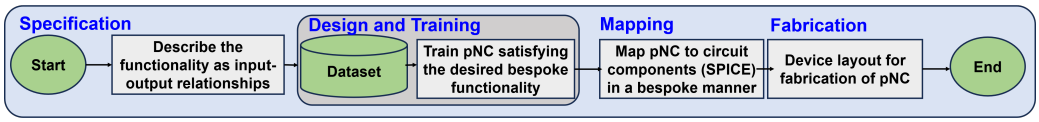

Fig. 4. On-demand design and fabrication given a specification of a desired functionality realized through training a pNC. The derived design can be readily fabricated through the on-demand fabrication capabilities of inkjet-PE [33].

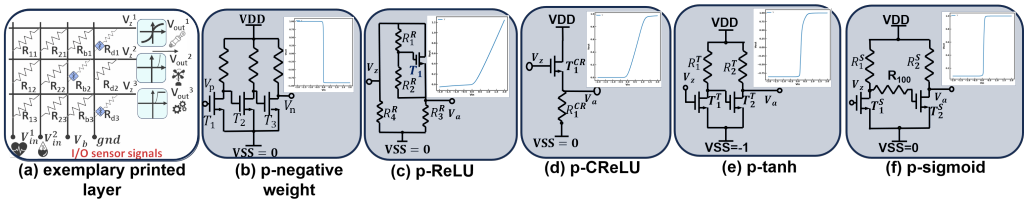

Fig. 5. Schematic of printed neuromorphic circuits. (a) example of a 3-input and 3-output printed layer based on crossbar array; (b) p-negative weight (c) p-ReLU (d) p-CReLU (e) p-tanh (f) p-sigmoid AF circuits [15].

Figure 3(f), inorganic oxide semiconductors, particularly n-EGTs are more promising, as they benefit from high electron mobility and sub-1V operation due to their high gate capacitance [30], making n-EGTs ideal for energy-efficient PE, while no reliable P-type EGTs have been reported yet [29, 31].

2.2 Printed Analog Neuromorphic Circuits (pNCs)

Neuromorphic computing,[2] driven by advancements in artificial intelligence, offers a powerful approach for solving complex tasks, with small NNs enabled by pNCs being especially cost-effective. These pNCs are ideal for real-time sensor data processing in resource-constrained applications like wearable sensors, flexible displays, and smart bandages [1, 2, 32], where they outperform conventional silicon-based hardware. By utilizing basic operations like weighted-sum and nonlinear activation, pNCs offer efficient, on-demand computing for various PE target applications. Figure 4 shows the existing flowchart of a pNC as available in the literature.

2.2.1 Hardware Primitives. Figure 5(a) illustrates the circuit schematics of a neuron in pNC. Some negative weight circuits are also incorporated in case required. Figure 5(b) and (c)–(f) shows the specific schematics of the negative weight circuit and the printed activation circuits [33]. The cross-sectional layouts of the primitives are shown in Figure 6. In the following, we will provide a detailed introduction of these circuit primitives.

Resistor crossbars. The resistor crossbar circuit, shown on the left side of Figure 5(a), follows Ohm's Law and Kirchhoff's Laws to calculate the output voltage V_z as a weighted-sum of input voltages V_i, with the weights determined by the conductance ratios as shown in Equation (1).

$$V_z = \frac{g_1}{G}V_1 + \frac{g_2}{G}V_2 + \frac{g_3}{G}V_3 + \frac{g_b}{G}V_b, \tag{1}$$

where g_i signifies the conductances of the resistors R_i and G represents the aggregate conductance $\sum_j g_j + g_b + g_d$. This design enables the customization of conductances to achieve desired weights, analogous to training in ANNs.

[2]Here, in these works, the term "neuromorphic" is used broadly to indicate analog computational circuits inspired by NN architectures, using nonlinear AFs and resistor crossbar arrays for parallel computation.

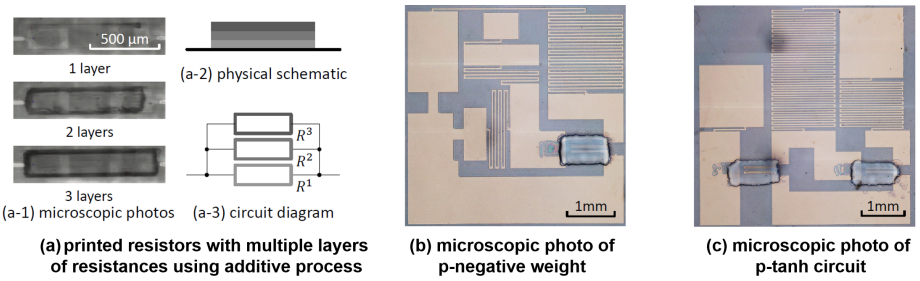

Fig. 6. Cross-sectional view of pNC primitives: (a) resistive crossbars, (b) printed negative-weight, and (c) printed tanh with nEGT, resistors in PE [33].

Printed negative weight circuits. Since the conductances in the crossbar resistor array can only represent positive weights, some resistors are paired with inverter-based circuits [28], as depicted in Figure 5(b), to enable negative weight representation. This setup allows for the emulation of multiplication with negative weights by inverting the input voltages V_i.

Printed activation circuits. After passing through the crossbar, the signals can be processed by different printed activation circuits, as shown in Figure 5 (c–f), which emulate the AFs commonly used in ANNs. These AFs are crucial in introducing nonlinearity into the system, enabling the network to model complex functions. For example, **p-tanh** maps the input to a range [-1,1], which is useful for balanced and stronger gradient flow [15]. The **p-sigmoid** outputs values between [0,1], which is ideal for binary classification tasks [34]. Similarly, **p-ReLU** passes only positive inputs, promoting network sparsity and addressing gradient saturation issues [35], while **p-CReLU** limits the maximum output value, ensuring stability by preventing overly large activations. Although the existing printed AFs are learnable and are designed to maintain ANN performance, they are more sensitive to defects, which leads to unstable behavior in pNCs [21].

2.3 Related Works

Recent studies on analog VLSI implementations of NNs have made significant progress in improving fault endurance [43, 44] and robustness to process variations through innovative design methodologies. Gowda et al. introduced redundancy and dynamic reconfiguration strategies that allow faulty components to be replaced with spare fault-free ones, ensuring high reliability in silicon-based circuits under process variations [36]. In addition, recent work has introduced **printed stochastic** NNs hardware accelerators, leveraging stochastic computing principles to improve robustness against process variations and transient errors in PE [45–47]. [42] proposed an FT on-line training method for **Resistive Random-Access Memory** (**RRAM**)-based computing systems, combining a quiescent-voltage comparison method for fault detection with a threshold-training and remapping scheme to enhance robustness in neural computing applications. [48] proposes a **device variability aware** (**DVA**) training methodology where stochastic noise is added to the parameters during training to enhance the robustness of network to the parameter's variation. FAT-RABBIT [37] introduces FAT with the M-SAM optimizer to improve robustness against bit-flip attacks in deep neural networks without additional hardware overhead. [49, 50] introduces zero-overhead solutions to improve the reliability of **deep neural networks** (**DNNs**) against transient hardware failures by redesigning and retraining models. [51] presented several network design choices and a training procedure that increases the robustness of standard DNN models. Also, Leem et al. introduced cross-layer error resilience for robust systems [41].

Table 1. Related Works and Comparison with Our Proposed Method

Ref.	Technology	Fault Model	Red.	Variation	Aging	FAT	Acc.	Fault Detection	Complexity and Hardware overhead
[36]	Analog VLSI	✓	–	–	–	–	–	Active	High (different signal levels)
[37–40]	Digital NN	✓	✓	–	–	✓	>80%	Active	High (digital logic, MUX, switching)
[41]	Digital VLSI	–	–	–	–	–	–	Cross-layer	High (digital logic, MUX, switching)
[42]	Memristive NN	✓	–	✓	–	✓	> 70%	Active	High (MUX, logic switching)
[14]	pNC	–	–	–	✓	–	>80%	–	Low (bespoke design)
[15]	pNC	–	–	✓	–	–	>78%	–	Low (bespoke design)
[33]	pNC	–	–	✓	–	–	> 66%	–	Low (bespoke design)
Proposed	pNC	✓	✓	✓	–	✓	**79.5%**	**Passive**	**Moderate (bespoke design, FER, FAT)**

Prior studies addressed fault modeling in VLSI, variation and aging awareness in PE. Our approach uniquely integrates the critical aspects through redundant (Red.) circuit design using FER, preserves the accuracy (Acc.) fault modeling and variation awareness within FAT.

Table 1 summarizes the state-of-the-art works on fault modeling and training methodologies, focusing on their capabilities (fault modeling, variation and aging awareness, FAT) and applicability to PE. Although several approaches in PE address specific aspects such as variation [15, 33], redundancy [37–40], or aging [14] individually, they do not consider to integrate fault modeling with FAT into their approaches. Moreover, conventional methods typically involve complex hardware architectures with fine-grained control and significant overhead, making them impractical for ultra-low-cost and disposable printed circuits. Also, pNCs must uniquely address challenges such as higher defect rates, significant manufacturing variability, and stringent constraints on cost and disposability.

Our proposed approach fundamentally differs from conventional memristive crossbars and conventional analog ANNs. Memristive crossbars encode synaptic weights through adjustable memristive elements, offering general-purpose programmability but requiring complex reconfiguration circuitry. In contrast, our pNCs offers precise fabrication control through additive manufacturing, embedding permanently optimized device characteristics specifically matched to trained neural models. This bespoke hardware mapping significantly reduces complexity and overhead, enabling efficient fault endurance. By integrating FAT with a Gumbel-Softmax distribution [52], our method facilitates dynamic AFs selection, utilizing lightweight hardware redundancy within additive manufacturing constraints to achieve increased robustness.

2.4 Fault Models in Printed Circuits

Circuit faults can arise from various issues, such as defective components, signal line breaks, short circuits, and delays, which hinder the proper functioning of circuits. Figure 2 illustrates the typical defects that occur in n-EGTs [53] during the inkjet printing process [19]. In general, circuit faults can be categorized into two types: *permanent faults*, which persist and are easily detectable during testing, and *temporary faults*, which appear and disappear quickly. Permanent faults are further classified into *catastrophic faults* (such as open and short circuits) and *parametric faults* (resulting from variations in process parameters) [54].

2.4.1 Catastrophic Faults. Catastrophic faults are categorized into *stuck-open* and *stuck-short* faults. In stuck-open faults, the terminals of a component lose contact, leading to high resistance. These faults can be simulated by adding a large series resistor (e.g., $R_s = 100\,\mathrm{M\Omega}$) to the faulty component. In contrast, stuck-short faults involve a short circuit between terminals, effectively bypassing the components. These can be simulated by adding a small parallel resistor (e.g., $R_p = 1\,\Omega$). Both types of faults can occur in resistors, capacitors, or transistors. Although catastrophic faults often disrupt circuit functionality, some may only affect performance specifications without impacting overall operation [36, 54].

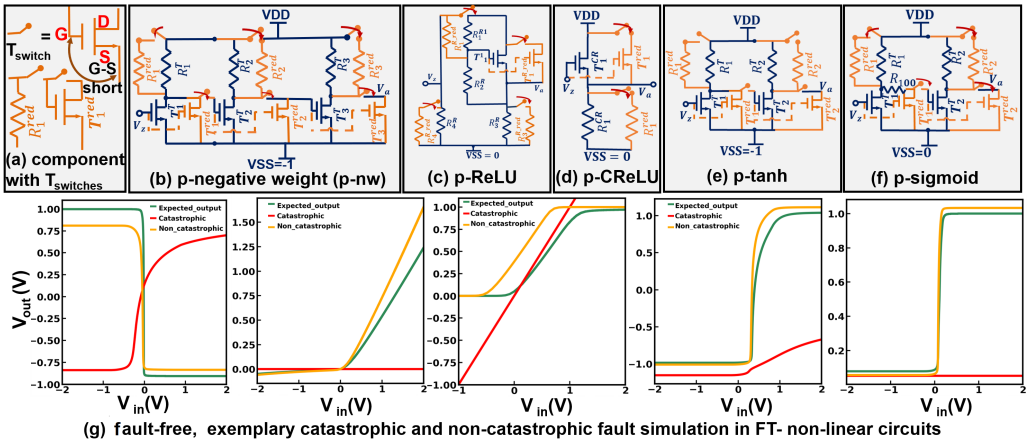

Fig. 7. Overview of (a) redundant transistor and resistor configuration for FT pNCs (b) p-negative weight circuit; printed activation circuit designs of (c) p-ReLU (d) p-CReLU (e) p-tanh (f) p-sigmoid and (g) corresponding fault-free and faulty characteristics curves.

2.4.2 Parametric Faults. Parametric faults impact the values of components such as resistors, capacitors, and transistors and can be caused by local or global defects. *Global parametric faults* arise from manufacturing imperfections that affect all components during production, while *local parametric faults* arise from specific defects, such as particles affecting the channel length of a transistor.

Our methodology addresses both catastrophic and parametric faults. Catastrophic faults, such as open or short circuits, are mitigated by adding hardware redundancy with parallel transistor-resistor paths, followed by FAT. Parametric faults, like resistor or transistor variations, are addressed through variation robustness as described in [15], and then further mitigated using FAT. This combined approach ensures robustness against both types of faults without explicit runtime fault detection or additional complexity.

3 Fault-Tolerant (FT) Printed Circuit Design

As shown in Figure 7, in this section, we describe the design of FT nonlinear circuits, aimed mainly at mitigating catastrophic faults by assuming single-component failures in either resistors or transistors. Furthermore, in Section 4, we introduce an FAT algorithm that dynamically selects the appropriate AF for each neuron. This bespoke approach aims to minimize accuracy degradation in pNCs, ensuring robustness while reducing hardware costs, even in the presence of faults.

The printed FT nonlinear circuits are designed with redundant components to ensure that it continues to function properly even if certain components fail, and thus follows a consistent design principle. The main component of these circuits includes redundant resistors, transistors, and transistors configured as switches to reroute the current path in the event of any faults.

3.1 Circuit Working Principle

The FT activation circuits use printed n-EGTs and resistor networks to implement a robust analog neuromorphic computing system. The core working principle relies on the inherent tunability of EGT threshold voltages (V_{th}) by simple adjustments during the printing process—such as varying electrolyte concentration or gate material composition—individual transistor switches can be precisely used to exhibit different conduction states. This simplifies the design of FT nonlinear transfer

functions (e.g., ReLU, sigmoid, and tanh), without requiring additional complex manufacturing steps.

Unlike CMOS fabrication, where individual threshold voltage adjustments are challenging due to specialized implants and costly processes, printed EGT allows easy per-device threshold adjustments. Additionally, printed resistors uniquely benefit from additive manufacturing, allowing precise deposition of individual resistors without added complexity. Although redundancy is common in traditional CMOS circuits, redundancy in PE has not been extensively explored due to its technology-related constraints. Thus, we introduce redundancy through multiple parallel EGT and resistor paths, each with slightly ***varied thresholds and conduction characteristics***. This provides inherent robustness against manufacturing variability, catastrophic faults (short/open circuits), and parametric deviations. Crucially, our circuits rely solely on built-in redundancy based on the concept of ***analog forward error recovery (FER)***, in which circuit components are designed with redundancy to inherently tolerate ***single-device defects (transistor or resistor)***. This approach does not require any form of reconfiguration after faults occur, significantly enhancing yield and reliability for ultra-low-cost scalable printed neuromorphic systems.

3.1.1 FT Operation. In Figure 7(a), the T_{switch} transistors act as two-terminal switches, with gate-source (G-S) shorted, operating based on the voltage difference between them and are connected with each redundant component. Under normal operation, they remain off, but if a catastrophic fault occurs in any component of the circuit, the G-S voltage of T_{switch} increases above a threshold voltage (V_{th}). The T_{switch} turns on, and allows the current to flow between the drain and the source (D-S) to either one of the redundant resistor's (R_1^{red}) or transistor's (T_1^{red}) paths. *It is worth mentioning that the threshold control in n-EGTs is easily achieved by transistor channel sizing [55], thus enabling multiple n-EGTs with varying thresholds in the same circuit.* So, even if the primary component fails, the current flow is maintained without disrupting the overall functionality. However, in any non-catastrophic fault, such as a partial short or open resistor R_2^R (as in ReLU), the circuit remains functional, without causing significant performance degradation, and does not require redundant components. Also, the redundancy ensures automatic functionality without requiring explicit switches or external logic control. Under normal conditions, each transistor-resistor path, having slightly varied electrical characteristics (threshold voltage (V_{th})), together contributes to the intended circuit functionality (as shown in Figure 5).

FT Printed Negative Weight (Inverter) Circuit. The transfer characteristic of the FT negative weight (analog inverter) circuit (Figure 7 (b)) is characterized by

$$\text{neg}(V_z) = -\left(\eta_1^N + \eta_2^N \cdot \tanh\left(\left(V_z - \eta_3^N\right) \cdot \eta_4^N\right)\right), \tag{2}$$

where $\eta^N = [\eta_1^N, \eta_2^N, \eta_3^N, \eta_4^N]$ = [0.04, 0.95, 0.01, 142.20] are fixed auxiliary parameters determined by the physical quantities $q^N = [R_i^N, W_i^N, L_i^N]$, where R_i^N, W_i^N, L_i^N are the resistances, width, and length of the transistors, respectively.

FT Printed ReLU Circuit. The printed ReLU activation circuit shown in Figure 7(c) implements a piecewise linear function that remains linear for positive inputs and nullifies negative inputs. So, a combination of a softplus function for smoothness at $V_z = 0$ and a linear function for the negative slope is used to accurately describe the circuit's behavior. Consequently, the function describing p-ReLU circuit is

$$V_a = \eta_1^R \cdot \left(x - \eta_3^R\right) + \eta_2^R \cdot \text{softplus}\left(V_z - \eta_3^R, \eta_5^R\right) + \eta_4^R, \tag{3}$$

and $\eta^R = [\eta_1^R, \eta_2^R, \eta_3^R, \eta_4^R, \eta_5^R]$=[0.01, 0.66, 0.20, 0.003, 7.74] are fixed auxiliary parameters determined by circuit components $q^R = [R_i^R, W_i^R, L_i^R]$, where R_i^R, W_i^R, L_i^R are the resistances, width and length of the transistors, respectively.

*FT Printed **Clipped ReLU (CReLU)** Circuit.* The FT CReLU activation, as shown in Figure 7(d), extends the basic ReLU by capping the output voltage so $V_a \in [0, V_{max}]$. The mathematical expression of the CReLU is

$$V_a = \begin{cases} \eta_1^{CR}, & V_z < \eta_3^{CR} \\ \eta_2^{CR}, & V_z > \eta_4^{CR} \\ \frac{\eta_2^{CR} - \eta_1^{CR}}{\eta_4^{CR} - \eta_3^{CR}} V_z + \frac{\eta_1^{CR}\eta_4^{CR} - \eta_2^{CR}\eta_3^{CR}}{\eta_4^{CR} - \eta_3^{CR}}, & \text{otherwise,} \end{cases} \tag{4}$$

with $\eta^{CR} = [\eta_1^{CR}, \eta_2^{CR}, \eta_3^{CR}, \eta_4^{CR}]$ = [0.001, 0.96, 0.02, 1.17] are fixed auxiliary parameters determined by the physical quantities $q^{CR} = [R_i^{CR}, W_i^{CR}, L_i^{CR}]$, where $R_i^{CR}, W_i^{CR}, L_i^{CR}$ are the resistances, width, and length of the transistors, respectively.

FT Printed Tanh Circuit. Figure 7 (e) shows the schematic of an FT p-tanh circuit and is realized by:

$$V_a = \text{ptanh}(V) = \eta_1^T + \eta_2^T \cdot \tanh\left(\left(V - \eta_3^T\right) \cdot \eta_4^T\right) \tag{5}$$

with the fixed auxiliary parameters $\eta^T = [\eta_1^T, \eta_2^T, \eta_3^T, \eta_4^T]$ =[0.05, -0.91, 0.23, -9.24] determined by $q^T = [R_i^T, W_i^T, L_i^T]$,, where R_i^T, W_i^T, L_i^T are the resistances, width, and length of the transistors, respectively.

Printed Sigmoid Circuit. Similar to the p-tanh AF circuit design in Figure 7(f), the p-sigmoid can also be modeled by a suitable mathematical equation:

$$V_a = \eta_1^S + \eta_2^S \cdot \text{sigmoid}\left(\left(V_z - \eta_3^S\right) \cdot \eta_4^S\right), \tag{6}$$

where the sigmoid(x) function is defined as sigmoid$(x) = \frac{1}{1+e^{-x}}$, and $\eta^S = [\eta_1^S, \eta_2^S, \eta_3^S, \eta_4^S]$ = [0.99, -0.91, 0.10, -48.71] are fixed auxiliary parameters determined by the physical quantities $q^S = [R_i^S, W_i^S, L_i^S]$, where R_i^S, W_i^S, L_i^S are the resistances, width, and length of the transistors, respectively.

The normal and faulty transfer characteristics of all the AFs are shown in Figure 7(g), assuming single-component failures in either resistors or transistors.

3.1.2 Redundancy Management and Complexity. In our FT approach, redundancy is realized through analog FER. Unlike conventional digital redundancy, which requires active fault detection and complex reconfiguration, our printed circuits leverage passive redundancy directly embedded during additive manufacturing. Specifically, each printed AFs incorporates multiple parallel transistor-resistor paths, inherently managing faults passively by automatically rerouting current in the event of single-component-level faults such as open or shorts. This design significantly minimizes additional complexity as it eliminates the need for dedicated fault detection circuits, digital logic overhead, or external runtime configuration.

Furthermore, the implementation uses bespoke pNCs, in which NN parameters are physically embedded during fabrication. While bespoke network architectures inherently risk fault vulnerability due to fabrication deviations affecting hard-coded parameters, our passive redundancy substantially mitigates this risk. Moreover, our approach utilizes FT architecture search, dynamically selecting robust combinations of normal and FT components during training. This co-optimization ensures high fault tolerance and minimal hardware overhead, leveraging the NN's inherent approximate computing nature and intrinsic redundancy.

4 Fault-Aware Training (FAT) Framework

In this section, we present our FAT framework for modeling FT-pNCs, which integrates three key aspects and is also shown in the proposed implementation flow, Figure 8:

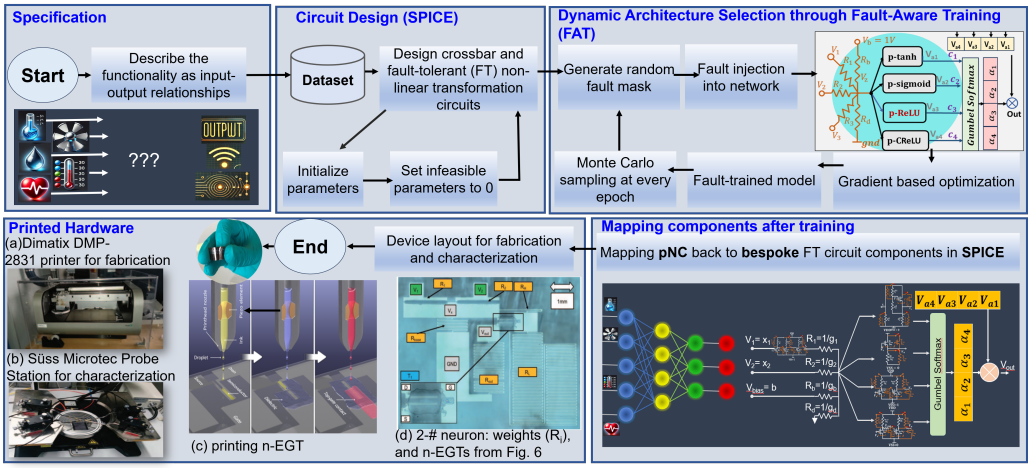

Fig. 8. Proposed implementation flow for an on-demand bespoke fault-tolerant printed neuromorphic circuit (FT-pNC) X-design given a specification of a desired functionality realized through Gumbel-Softmax distribution through FAT.

- **Initialization of Fault-Aware Printed Layer:** Our method uses a printed NN layer (`pLayer`) structure (Algorithm 1, Lines 1–5). Initially, each layer is configured with several trainable parameters:
  - **Conductances (θ):** These represent the crossbar weights and are initialized with small random values.
  - **Selection Logits (*coefficients_act, coefficients_neg*):** Trainable parameters controlling dynamic selection of activation functions and negative weight implementations via the Gumbel-Softmax mechanism.
  - **Temperature Parameters (*act_temp, neg_temp*):** Initially set to a higher value to encourage exploration in early training phases, and later gradually annealed during training iterations.
- **Fault Injection During Training:** We introduce defects (catastrophic: resistor open/short, transistor G-D/G-S/D-S shorts, non-catastrophic: resistance variation, transistor threshold voltage, width, and length variations) into crossbars and nonlinear circuits, derived from SPICE-level simulations.
- **Dynamic Architecture Search:** We use Gumbel-Softmax to *dynamically select* AFs and inverters at each neuron, enabling a differentiable architecture search that adapts to faults.
- **Fault-Aware Classification Loss:** We incorporate these faults into the forward pass and compute an expected classification loss, ensuring the model learns to remain accurate under realistic manufacturing defects.

In summary, our FAT framework integrates realistic fault scenarios directly into the training phase of pNCs. Using Monte Carlo-based fault injection, we randomly introduce realistic fault behaviors, derived from detailed SPICE simulations, into our NN training process and dynamically optimizes the architecture, selecting bespoke AFs via a Gumbel-Softmax, ensuring maximum robustness against the injected faults. Although our framework considers AFs, negative weight circuit, and crossbar conductances' faults, we place greater emphasis on faults in AFs due to their critical nonlinear behavior and inherent vulnerability to faults, which require more careful analysis

and mitigation strategies. The following subsections detail each component, with *Algorithm* 1 summarizing the core pseudo-code.

4.1 Simulated Fault Injection During Training

To evaluate pNCs fault endurance, we employ a *Monte Carlo-based* fault injection approach during training, targeting both crossbars and nonlinear circuits. As illustrated in Algorithm 1 (Lines 8, 12, 18), random masks $M_{\text{Res}}, M_{\text{INV}}, M_{\text{ACT}}$ alter the behavior of conductances, inverters, and AFs, modeling realistic printing defects. We consider resistor open/short conditions and transistor faults (gate-drain (G-D), gate-source (G-S), or drain-source (D-S) shorts/opens), each with specific behaviors derived from SPICE simulations.

4.1.1 Top-Down Sampling Strategy for Faulty Layers. As shown in *Algorithm* 1 (Lines 21–32, 35–38), we simulate faults by selecting which layers are faulty, then deciding which components in those layers (AFs, inverters, conductances) will be corrupted. Let N_l denote the total number of components in layer l. We define a probability

$$p(l) \;=\; \frac{N_l}{\sum_{l=1}^{L} N_l}, \tag{7}$$

which ensures that layers with more components have a higher probability of receiving faults. After choosing e_{fault} layers based on $p(l)$, we further distribute faults among AFs, inverters, and conductances proportionally to their device counts. Specifically,

$$\begin{aligned} p(l) &= \frac{N_l}{\sum_{l=1}^{L} N_l}, \qquad p(\text{Type} = ACT \mid l) = \frac{N_l^{ACT}}{N_l}, \\ p(\text{Type} = INV \mid l) &= \frac{N_l^{INV}}{N_l}, \qquad p(\text{Type} = Res \mid l) = \frac{N_l^{Res}}{N_l}. \end{aligned} \tag{8}$$

where N_l^{ACT}, N_l^{INV}, and N_l^{Res} denote the counts of AFs, inverters, and resistors in layer l. This approach maintains a consistent fault rate across the network, reflecting the higher probability of defects in layers with more total components.

4.1.2 Masks for Crossbars and Nonlinear Circuits. Once we determine which components are faulty, we apply specialized masks, as shown in *Algorithm* 1 (Lines 8, 12, 18):

- **Crossbar Conductances** (`Res`): A *ternary mask* M_{Res} multiplies each conductance g by $\{1, 0, \infty\}$, simulating no fault, open circuit, or short circuit, respectively. Hence, $g \to 0$ emulates an open fault, and $g \to \infty$ a short.
- **Nonlinear Circuits** (`ACT` or `INV`): We define a generic binary mask M_{NL} for any *nonlinear* subcomponent, whether it is an AF or an inverter. Concretely,

$$M_{\text{NL}} \;\in\; \{M_{\text{ACT}}, M_{\text{INV}}\},$$

meaning that M_{NL} corresponds to M_{ACT} if the subcomponent is an AF, and M_{INV} if it is an inverter. Formally, if $M_{\text{NL},k} = 0$, then subcomponent k is fault-free, while $M_{\text{NL},k} = n \neq 0$ indicates that the nth faulty transfer function is used instead of the normal function. In equation form:

$$\begin{aligned} \text{NL}_{\text{faulty}}(x) = \sum_k \big(&\mathbb{I}(M_{\text{NL},k} = 0) \cdot \text{NL}_k(x) \\ &+ \mathbb{I}(M_{\text{NL},k} = n) \cdot \mathit{FBDataset}_{\text{NL}}[k, n](x) \big), \end{aligned} \tag{9}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Thus, if M_{NL} corresponds to an AF, it acts as M_{ACT}, and if it corresponds to an inverter, it acts as M_{INV}. Either way, a nonzero mask entry means subcomponent k is replaced by the corresponding faulty behavior from the *FBDataset*.

During each forward pass, `MakeFault` updates these masks based on the sampled probabilities if $e_{\text{fault}} > 0$, while `RemoveFault` resets them otherwise, as shown in *Algorithm* 1 (Lines 22–24). The masked values directly impact the MAC output (Line 13) and subsequent layer computations.

4.1.3 Faulty Behavior Dataset (FBDataset). Although prior works [21] have analyzed fault sensitivity in certain pNCs, they did not provide any solution to improve their fault-endurance. Our approach strengthens the pNCs robustness at both the *design* and *algorithmic* levels. Specifically, we first characterize printed circuit blocks such as AFs and negative weight circuit (analog inverter) (as in Figure 6) through SPICE simulations to capture their response under multiple fault conditions, such as transistor open faults, transistor short faults, resistor open faults, and parametric deviations (e.g., variations in transistor threshold voltages and resistor values). These faulty behaviors were recorded as input-output pairs representing how each fault impacted the circuit's analog transfer characteristics. These functions are stored in a *Faulty Behavior Dataset* (*FBDataset*). When M_{ACT} or M_{INV} flags a subcomponent as faulty, the network substitutes its normal function with the corresponding entry from *FBDataset*, accurately modeling real-world printing defects.

4.1.4 Forward Pass Under Fault Masks. In each training iteration:

(1) **Faulty Layer Selection**: We choose up to e_{fault} layers based on $p(l)$. Within those layers, the probability $p(\text{Type} \mid l)$ decides how many AFs, inverters, or conductances are marked faulty.
(2) **Mask Application**: For crossbars, M_{Res} zeroes or inflates conductances to emulate open/short conditions. For nonlinear circuits, $M_{\text{ACT}}, M_{\text{INV}}$ dictate whether an AF/inverter is normal or replaced by a faulty counterpart from *FBDataset*.
(3) **Fault-Aware Computation**: The model's forward pass reflects these masked values, ensuring outputs incorporate realistic defect behaviors. Each iteration thus presents a different fault scenario, improving the model's overall robustness through repeated exposure.

By combining this mask-based approach with our dynamic architecture search (Section 4.2), the network adaptively learns circuit configurations that minimize accuracy loss, even under repeated exposures to realistic manufacturing faults.

4.2 Bespoke Architecture Optimization (Dynamic Architecture Search)

To further enhance fault endurance, we perform a *dynamic architecture search* at the circuit level. As illustrated in *Algorithm* 1 (Lines 11, 17), each neuron's choice of inverter and AF is determined by trainable logit vectors, which we sample using the Gumbel-Softmax distribution:

$$\text{GumbelSoftmax}(c_i, \tau) = \frac{\exp\big((\log(c_i) + g_i)/\tau\big)}{\sum_{j=1}^{n} \exp\big((\log(c_j) + g_j)/\tau\big)}, \tag{10}$$

where $g_i = -\log(-\log(U_i))$ with $U_i \sim \text{Uniform}(0, 1)$, and τ is a parameter controlling exploration vs. exploitation. Initially, a higher τ yields *soft* selections, allowing gradients to flow to multiple AFs/inverters. Over time, we decay τ (e.g., $\tau \leftarrow \max(0.95 \times \tau, 0.1)$) to approximate a near-hard argmax.

Masking and Gumbel-Softmax Synergy. The Gumbel-Softmax distribution serves as a differentiable approximation to categorical selection, making the choice of AFs a continuous optimization problem that can be efficiently solved using standard gradient-based backpropagation in PyTorch. Initially, each neuron considers all AF candidates simultaneously, weighted by learnable probabilities. During training, these probabilities are gradually optimized, resulting in one dominant AF per neuron at

ALGORITHM 1: Fault-Aware Printed Layer (`pLayer`) and Printed Neural Network (`pNN`)

Notation:
L: # layers; N_l: total devices in layer l; N_l^{ACT}, N_l^{INV}, N_l^{Res}: AFs, inverters, resistors in l; e_{fault}: faults per forward
$p(l) = N_l / \sum_{j=1}^{L} N_j$, $p(\text{Type} \mid l) = N_l^{\text{Type}} / N_l$, $\text{Type} \in \{\text{ACT}, \text{INV}, \text{Res}\}$
Trainable Params: θ : *conductances*; *coefficients_act, coefficients_neg* : *activation/inverter parameters*

procedure pLayer_Init($n_{\text{in}}, n_{\text{out}}$, ACT, INV)
  Init $\theta \in \mathbb{R}^{(n_{\text{in}}+2)\times n_{\text{out}}}$ ▷ $n_{\text{in}}, n_{\text{out}}$: input and output dims
  Init *coefficients_act*, *coefficients_neg*
  `act_temp`, `neg_temp` $\leftarrow 1.0$ ▷ Gumbel-Softmax temperature parameters for AF and inverter selection
  Init all-ones FaultMaskRes, zeros FaultMaskACT, FaultMaskNEG
end procedure
function MAC(**a**)
  $\theta_{\text{noisy}} \leftarrow \theta \odot (\text{noise}) \odot \text{FaultMaskRes}$ ▷ variation + fault aware training
  $\mathbf{W} \leftarrow \text{normalize}|\theta_{\text{noisy}}|$
  Split θ_{noisy} into $\mathbf{W}_+$, $\mathbf{W}_-$
  $\beta \leftarrow \text{GumbelSoftmax}(\textit{coefficients\_neg}, \texttt{neg\_temp})$
  $\mathbf{a}_{\text{neg}} \leftarrow \sum_i \beta_i \, \text{INV}_i(\mathbf{a})$ ▷ Each INV_i uses its own `FaultMaskNEG` internally
  return $\mathbf{a}\,\mathbf{W}_+ + \mathbf{a}_{\text{neg}}\,\mathbf{W}_-$
end function
function forward(**a**)
  $\mathbf{z} \leftarrow \text{MAC}(\mathbf{a})$
  $\alpha \leftarrow \text{GumbelSoftmax}(\textit{coefficients\_act}, \texttt{act\_temp})$
  $\mathbf{a}_{\text{out}} \leftarrow \sum_{i=1}^{|\text{ACT}|} \alpha_i \, \text{ACT}_i(\mathbf{z})$ ▷ Each ACT_i uses its own `FaultMaskACT` internally
  return $\mathbf{a}_{\text{out}}$
end function
procedure MakeFault(e_{fault})
  if $e_{\text{fault}} = 0$ **then**
    RemoveFault; **return** (FaultMaskRes, FaultMaskACT, FaultMaskNEG)
  end if
  for $i = 1 \ldots N_{\text{fault}}$ **do**
    Split $e_{\text{fault}} \sim (N_l^{\text{Res}} : N_l^{\text{ACT}} : N_l^{\text{INV}})$
    Pick random resistor indices; set $M_{\text{Res}} = 0$ or ∞
    Pick random AF indices; set M_{ACT} to nonzero code
    Pick random inverter indices; set M_{INV} to nonzero code
  end for
  return (FaultMaskRes, FaultMaskACT, FaultMaskNEG)
end procedure
pNN: sequence of `pLayer` modules
function pNN_forward(**x**)
  RemoveFault
  for $i = 1 \ldots e_{\text{fault}}$ **do**
    Sample layer $l \sim \text{Categorical}\{p(1), \ldots, p(L)\}$
    Count faults per layer and call MakeLayerFault(l, 1)
  end for
    for each layer ℓ **do**
      ℓ.`act_temp` $\leftarrow \max(0.95\,\ell.\texttt{act\_temp},\ 0.1)$; ℓ.`neg_temp` $\leftarrow \max(0.95\,\ell.\texttt{neg\_temp},\ 0.1)$
      $\mathbf{x} \leftarrow \ell.\text{forward}(\mathbf{x})$
    end for
    return x
  end function

convergence. This "bespoke" selection process ensures that each neuron's AF is individually tailored, maximizing fault endurance and accuracy based on the learned robustness characteristics specific to the printed neuromorphic hardware. As faults are injected randomly each iteration, the selected AF that performs better under these defects tends to achieve lower classification loss, driving up

its logit value. Over repeated exposures, each neuron converges on the circuit component most resilient to the observed faults. Thus, the network effectively learns a *bespoke* circuit architecture, picking subcomponents that best mitigate printing defects.

4.3 Loss Function for Fault Endurance

Finally, we use a fault-aware classification loss as shown in *Algorithm* 1 (Lines 33–44) that accounts for the random fault masks M. Let $f(x;\phi,M)$ be the network output under parameters ϕ and a mask M. We define:

$$\mathcal{L}_{\text{classification}} = \mathbb{E}_{p(M)}[\mathcal{L}_{\text{primary}}(y,\ f(x;\phi,M))]. \tag{11}$$

Because enumerating all faults is intractable, we sample $M^k \sim p(M)$ multiple times and average the resulting classification loss (e.g., cross-entropy). Formally,

$$\mathcal{L}_{\text{classification}} \approx \frac{1}{N}\sum_{k=1}^{N} \mathcal{L}_{\text{primary}}\big(y,\ f(x;\phi,M^k)\big). \tag{12}$$

By combining fault injection, dynamic circuit selection, and this expected classification loss, our framework learns robust pNCs that can tolerate real-world manufacturing variability without sacrificing accuracy.

4.4 Runtime (Post-fabrication) Fault Management

The FT designs inherently manage runtime faults passively. Redundant transistor-resistor pathways are integrated during additive printing. In operational scenarios, any component-level faults, such as opens or shorts, trigger automatic rerouting of current through available functional redundant paths. This automatic rerouting eliminates the necessity of active switching logic, multiplexers, controllers, or explicit fault classification circuits. Consequently, our printed circuits maintain functionality without additional runtime complexity, significantly reducing hardware overhead and simplifying operational reliability.

5 Evaluation

To evaluate the effectiveness and robustness of our proposed FAT method[3] in pNCs, we employed standard ANN models, specifically **multilayer perceptrons** (**MLPs**), trained and tested using well-established PE benchmark datasets as in [14, 28, 33]. After training, we mapped the printed ANN weights and neuron functionalities onto multiple printed crossbar arrays, with each crossbar functioning as an analog computing block. We further assessed the generalization capabilities of our method by selecting fault injection ratios at different training-to-testing scenarios: specifically 0:0, 0:10, 0:20, 0:30, 10:10, 10:20, and 10:30 using PyTorch [56]. Ratios like 0:0 represent the ideal (fault-free) baseline scenario, matched ratios like 10:10 reflect realistic assumptions of similar fault occurrence between training and operation, and mismatched ratios (e.g., 10:20, 10:30) investigate model resilience under harsher defects than those explicitly faced during training. These assumptions are guided by practical considerations relevant to PE [19], where devices experience varying degrees of defects or degradation over their operational lifetime.

5.1 Experiment Setup

Circuit Design and Training Configuration. The FT-pNCs model was trained using PyTorch [56], with datasets normalized to [0,1] and split into 60% training, 20% validation, and 20% testing. Fault injection was performed at two levels:

[3]Code is available at https://github.com/KIT-Neuromorphic-Computing/Fault_Aware_pNC

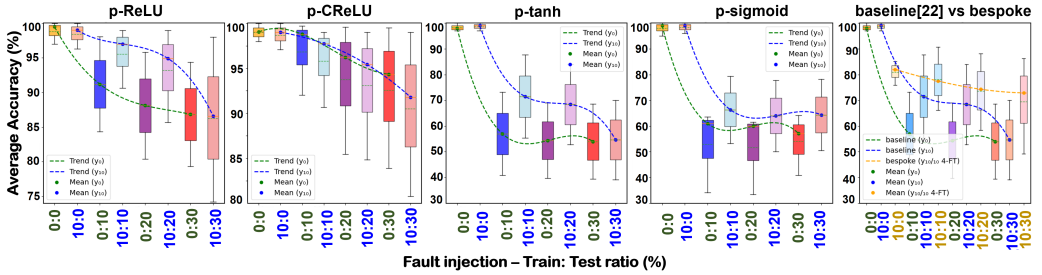

Fig. 9. (a)–(d) Evaluation of average accuracy of four single AFs using FT pNCs (e) effectiveness of bespoke FT AF over existing baseline [21] AF under 0% (no fault) and 10% (with fault) fault injection in training and up to 30% fault injection in testing averaged over 8 benchmark datasets. y_0 and y_{10} denotes 0% and 10% fault injection in training.

— **SPICE Level:** Fault injection in the non-linear AFs in Figure 7(b)–(g) was first conducted at the SPICE simulation level using the n-EGT P-PDK [26] in Cadence Virtuoso,[4] ensuring an accurate representation of physical defects and their effects on circuit transfer and fault characteristics based on ***FER***.
— **Algorithmic Level:** The fault effects, encapsulated in the SPICE-generated *FBDataset*, were abstracted into a PyTorch-compatible format for FAT. During training, fault rates ranging from 0% to 10% were introduced using Monte Carlo sampling, enhancing the model's robustness against defect-induced variations. This approach enabled the model to effectively handle both typical and extreme fault scenarios (0% to 30% fault rates) during testing. The Adam optimizer [57] was used during training.

Bespoke Architecture. In this section, we discussed the design of different AFs within the pNC, i.e., different neurons utilizing different AFs during training: (a) using 4-normal-AFs, (b) using 4-FT-AFs, and (c) using 8-combined AFs (4-normal and 4-FT). The functionality of the baseline printed neuromorphic hardware and the printed defects have been validated in [19, 33], and the contribution of this work, i.e., the fault-endurant approaches has been verified at the algorithmic level, the experiment is conducted at simulation level based on P-PDK [26].

5.2 Results and Discussion

This subsection presents the experimental results of proposed FT-pNCs, comparing its accuracy and hardware costs with the baseline and combined versions of AFs under various fault conditions on 8 benchmark datasets.

The box plots in Figure 9 show that FAT enhances the robustness of FT-AFs across various pNCs. In addition, for single FT-AFs, both p-ReLU and p-CReLU show the most notable accuracy with minimal variation under 0% till 30% fault conditions. Meanwhile, the ReLU family consistently demonstrated strong robustness when faults are injected during both training and testing, suggesting inherent characteristics that improve its fault resilience. However, they require high-value resistors ($\approx$ 1.5MΩ), which not only occupy a huge area but also consume more power.

As shown in Figure 10, the average accuracy across 8 benchmark datasets for the ideal fault-free scenario (0% fault during training and testing) is 72.3% when employing 4-Normal AFs. Under real fault conditions (10% faults during both training and testing), the accuracy of normal AFs significantly drops to 62.1%, while the 4-FT AFs improve this accuracy notably to 79.4%. Figure 10

[4] https://www.cadence.com/en_US/home.html

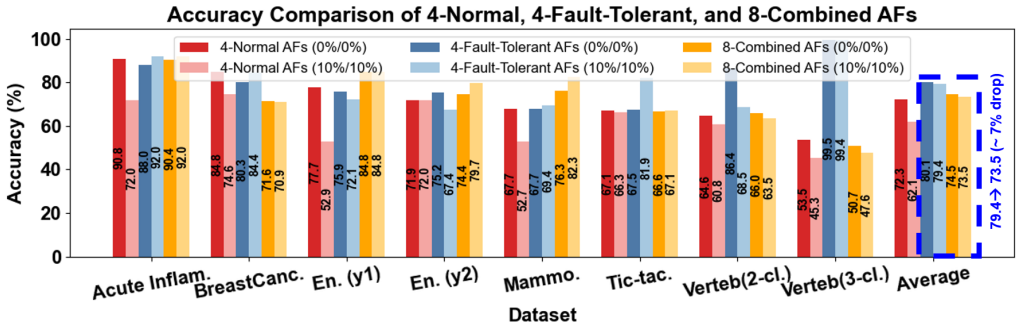

Fig. 10. (a)–(d) Evaluation of average accuracy of 4-Normal AFs, 4-FT AFs and 8-combined AFs under 0:0 and 10:10 train : test ratios over 8 benchmark datasets.

Table 2. Hardware Costs and Training Time (hours) Comparison of Normal, FT, and Combined AFs over 8 Benchmark Datasets Under Train: Test = 10% : 10% Ratios

Dataset	4-Normal AFs			4-FT AFs			8-Combined AFs		
	Area (mm^2)	Power (mW)	Time (hr)	Area (mm^2)	Power (mW)	Time (hr)	Area (mm^2)	Power (mW)	Time (hr)
Acute Infl.	34.40	0.455	10.6	32.48	4.616	45.8	33.66	1.978	17.1
BreastCanc.	21.36	0.552	11.0	36.48	0.712	46.9	40.87	1.119	18.2
En. (y_1)	32.65	5.560	4.9	43.96	10.91	54.9	40.33	85.2	24.8
En.(y_2)	33.06	6.276	7.3	36.94	40.45	51.6	30.92	1.402	24.9
Mammo.	34.26	28.24	9.6	43.76	31.26	46.7	42.29	0.620	20.5
Tic-tac	32.09	31.00	8.5	55.07	79.5	44.5	47.74	0.615	20.7
Verteb(2-cl.)	26.22	94.19	11.6	37.82	288.9	38.8	34.81	124.1	17.7
Verteb(3-cl.)	41.83	1.939	7.02	47.33	18.83	37.6	41.43	0524	16.7
Average	**31.98**	**20.57**	**8.81**	**41.73**	**59.39**	**45.9**	**39.00 ↓(6.54%)**	**26.99 ↓ (54.5%)**	**20.07 ↓(56.2%)**

(The reduction w.r.t 4-FT AFs at an acceptable accuracy drop (≈ 7%) is highlighted in blue.

and Table 2 further confirm that the 8-combined AF balances accuracy, robustness, area, and power, offering a viable solution for applications where these tradeoffs are critical. For datasets, *verte3cl.* and *tictac.*, the 4-FT-AFs show an unusually high accuracy (81.9% and 99.4%, respectively) at the 10%/10% train/test ratio, which is much higher than their accuracy in other datasets, whereas *energy2.* dataset results in an accuracy drop to 67.4% in 4-FT-AFs. These suggest that certain AFs have struggled or excelled depending on the characteristics of the datasets. For some datasets, the accuracy of the FT-AF under the 10%/10% train/test ratio fault scenario exceeds that of 0%/0% train/test fault-free scenario. Fault injection during training has acted as a form of regularization, allowing the model to generalize better, even in a non-faulty environment.

However, our evaluation clearly shows that adopting only 4-FT AFs introduces higher overhead compared to the baseline (4-normal AFs), increasing area by ≈ 30.5% (1.31 ×), power by ≈ 188% (2.89 ×), and training time by ≈ 420% (5.21 ×). To address this issue, we further adopt the combined fault-aware approach (8-Combined AFs), which effectively reduces these overheads, achieving significant reductions of 6.54% (1.23 ×) in area, 54.5% (1.31 ×) in power, and 56.2% (2.28 ×) in training time compared to the only 4-FT AFs while incurring only ≈ 7% classification accuracy drop. This ≈ 7% accuracy degradation for significant reductions in power (54.5%) and area (6.54%) is practically justified in resource-constrained applications where energy efficiency, cost, and size constraints outweigh the demand for maximum achievable accuracy. For e.g., in applications like disposable smart

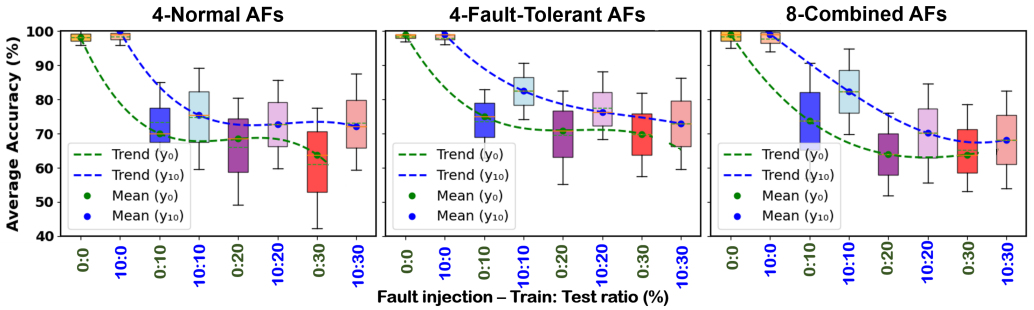

Fig. 11. (a)–(c) Evaluation of normalized accuracy of three bespoke architectures under 0% (no fault) and 10% (with fault) fault injection in training and upto 30% fault injection in testing averaged over 8 benchmark datasets. y_0 and y_{1_0} denotes 0% and 10% fault injection in training.

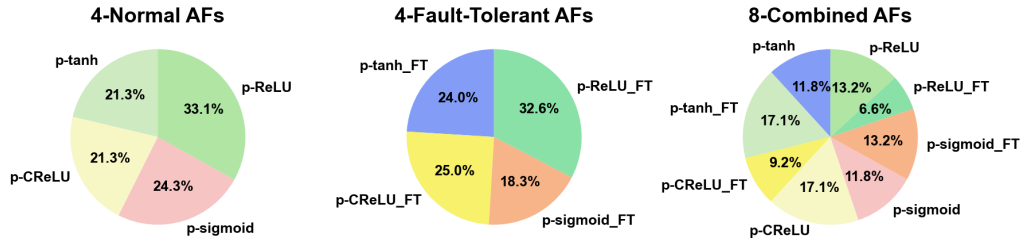

Fig. 12. Percentage of AFs used when (i) 4-Normal AFs (ii) 4-FT AFs and (iii) 8-Combined AFs are selected averaged over 8 benchmark datasets under train: test = 10% : 10% ratios.

bandages or wearable health-monitoring sensors, maintaining adequate battery life, minimizing device footprint, and ensuring affordability are prioritized and moderate accuracy losses are acceptable, provided the system maintains sufficient reliability under real faults. It is also observed that the normal AF has a slightly larger area (34.40 mm^2) than the FT-AF (32.48 mm^2) for the **"Acute Infl."** dataset which seem counterintuitive but can occur due to the bespoke (customized) selection of AFs. In some cases, this approach might choose AFs that inherently require fewer or smaller circuit components compared to the uniformly chosen normal AFs, resulting in a smaller overall area.

Also, the training time significantly increases from the normal AFs ($\approx$ 8.81*hours*) to the FT ones ($\approx$ 45.9*hours*), reflecting the additional computational complexity introduced by fault-aware optimization. However, using combined AFs moderately reduces the training overhead ($\approx$ 20.07*hours*) by $\approx$ 56.2%, showing a tradeoff between accuracy, robustness, and computational cost.

Figure 11 compares the robustness of three bespoke AF architectures—4-Normal, 4-FT, and 8-combined—across different fault injection train-test ratios, averaged over 8 benchmark datasets. At low fault levels (e.g., 0:0 and 10:10), all architectures exhibit high accuracy and low standard deviations, indicating stable performance. However, as fault severity increases (e.g., 0:30 and 10:30 scenarios), the 4-Normal AFs show a sharp decline in average accuracy accompanied by notably higher standard deviations, indicating increased variability and instability. In contrast, the 4-FT AFs demonstrate significantly better robustness, maintaining higher mean accuracy with smaller standard deviations, showing consistent and reliable operation under increased fault levels. The 8-combined AFs give an intermediate solution, balancing accuracy and robustness, with moderate mean accuracy and lower standard deviation compared to the normal AFs.

The selection of AFs in each pNC, as in Figure 12, reflects their ability to handle faults in faulty scenarios. In both 4-normal-AF and 4-FT-AF, ReLU family is preferred for its simplicity, efficiency, and better generalization in resource-constrained environments, while p-sigmoid is chosen for its robustness against severe faults. The 8-combined AF balances all the AFs leveraging their strengths to optimize area and power consumption, thus minimizing the drop in accuracy under various faulty conditions.

6 Conclusion

Printed Electronics, due to their unique features, are gaining attention for next-generation electronics. However, the inherent defects of maskless additive manufacturing reduce the manufacturing yield and lower run-time reliability of the printed circuits. This work establishes a foundation for the design of fault-tolerant printed neuromorphic circuits by leveraging fault-aware training. Also, our work currently focus on single-component failures, capturing typical fault conditions. However, the inherent redundancy and forward error recovery embedded in our approach may also provide resilience against more complex, multi-component fault scenarios. This approach not only opens new fault-aware optimization opportunities, but also improves robustness against manufacturing defects. Future research may explore fault-aware strategies in complex NAS architectures.

References

[1] A. U. Alam, P. Rathi, H. Beshai, G. K. Sarabha, and M. J. Deen. 2021. Fruit quality monitoring with smart packaging. *Sensors* 21, 4 (2021), 1509. https://doi.org/10.3390/s21041509

[2] Qizeng Sun, Li Wang, Guozhang Ren, Linrong Zhang, Huixiang Sheng, Yameng Zhu, Hongchen Wang, Gang Lu, Hai-Dong Yu, and Wei Huang. 2022. Smart band-aid: Multifunctional and wearable electronic device for self-powered motion monitoring and human-machine interaction. *Nano Energy* 92, Art. no. 106840 (2022). DOI: 10.1016/j.nanoen.2021.106840

[3] Ganapati Bhat, Yigit Tuncel, Sizhe An, Hyung Gyu Lee, and Umit Y. Ogras. 2019. An ultra-low energy human activity recognition accelerator for wearable health applications. *ACM Transactions on Embedded Computing Systems* 18, 5s, Article 49 (2019), 22 pages. DOI: http://doi.org/10.1145/3358175

[4] Ganapati Bhat, Dina Hussein, and Nuzhat Yamin. 2024. *Robust Machine Learning for Low-Power Wearable Devices: Challenges and Opportunities*. Springer Nature Switzerland, Cham. DOI: http://doi.org/10.1007/978-3-031-40677-5_3

[5] Sudeep Pasricha, Raid Ayoub, Michael Kishinevsky, Sumit K. Mandal, and Umit Y. Ogras. 2020. A survey on energy management for mobile and IoT devices. *IEEE Design and Test* 37, 5 (2020), 7–24. DOI: http://doi.org/10.1109/MDAT.2020.2976669

[6] Ganapati Bhat and Pietro Mercati. 2025. Special issue on wearable IoT devices for reliable mobile health applications. *IEEE Design and Test* 42, 2 (2025), 5–6. DOI: http://doi.org/10.1109/MDAT.2025.3528866

[7] Partha Pratim Pande. 2025. Special Issue on Wearable IoT Devices for Reliable Mobile Health Applications. *IEEE Design and Test* 42, 2 (2025), 4–4. DOI: http://doi.org/10.1109/MDAT.2025.3528867

[8] R. Martins, I. Ferreira, and EEMC Fortunato. 2011. Electronics with and on Paper. *Physica Status Solidi (RRL)–Rapid Research Letters* 5, 9 (2011), 332–335.

[9] Toygun Basaklar, Yigit Tuncel, and Umit Ogras. 2024. A comprehensive multi-objective energy management approach for wearable devices with dynamic energy demands. *ACM Transactions on Internet Things* 5, 4, Article 26 (2024), 24 pages. DOI: http://doi.org/10.1145/3699964

[10] Priyanjana Pal, Haibin Zhao, Maha Shatta, Michael Hefenbrock, Sina Bakhtavari Mamaghani, Sani Nassif, Michael Beigl, and Mehdi B. Tahoori. 2024. Analog printed spiking neuromorphic circuit. In *Proceedings of the IEEE DATE*. 6 S.

[11] Muhammad Husnain Mubarik, Dennis D. Weller, Nathaniel Bleier, Matthew Tomei, Jasmin Aghassi-Hagmann, and Mehdi B. Tahoori. 2020. Printed machine learning classifiers. In *Proceedings of the 2020 53rd Annual IEEE/ACM MICRO*. 73–87. DOI: http://doi.org/10.1109/MICRO50266.2020.00019

[12] Enrico Sowade, Eloi Ramon, Kalyan Yoti Mitra, Carme Martínez-Domingo, Marta Pedró, Jofre Pallarès, Fausta Loffredo, Fulvia Villani, Henrique L. Gomes, Lluís Terés, and Reinhard R. Baumann. 2016. All-inkjet-printed thin-film transistors: Manufacturing process reliability by root cause analysis. *Scientific Reports* 6, 1 (2016), 33490.

[13] Mohammad H. Behfar, Behnam Khorramdel, Arttu Korhonen, Elina Jansson, Aleksis Leinonen, Markus Tuomikoski, and Matti Mäntysalo. 2021. Failure mechanisms in flip-chip bonding on stretchable printed electronics. *Advanced Engineering Materials* 23, 12 (2021), 2100264. DOI: http://doi.org/10.1002/adem.202100264

[14] Haibin Zhao, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2022. Aging-aware training for printed neuromorphic circuits. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD'22)*. 9. DOI: http://doi.org/10.1145/3508352.3549411

[15] Haibin Zhao, Brojogopal Sapui, Michael Hefenbrock, Zhidong Yang, Michael Beigl, and Mehdi Baradaran Tahoori. 2023. Highly-bespoke robust printed neuromorphic circuits. In *Proceedings of the 2023 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 1–6. Retrieved from https://api.semanticscholar.org/CorpusID:259028025

[16] Giorgos Armeniakos, Georgios Zervakis, Dimitrios Soudris, Mehdi B. Tahoori, and Jörg Henkel. 2022. Cross-layer approximation for printed machine learning circuits. In *Proceedings of the 2022 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 190–195. DOI: http://doi.org/10.23919/DATE54114.2022.9774689

[17] Priyanjana Pal, Alexander Studt, Tara Gheshlaghi, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2025. SpikeSynth: Energy-efficient adaptive analog printed spiking neural networks. In *Proceedings of the 2025 IEEE ICCAD*. 1–6.

[18] Osman S. Unsal, Israel Koren, and C. Mani Krishna. 2002. Towards energy-aware software-based fault tolerance in real-time systems. In *Proceedings of the 2002 International Symposium on Low Power Electronics and Design (ISLPED'02)*. Association for Computing Machinery, New York, NY, USA, 124–129. DOI: http://doi.org/10.1145/566408.566442

[19] Ahmet Turan Erozan, Simon Bosse, and Mehdi B. Tahoori. 2021. Defect detection in transparent printed electronics using learning-based optical inspection. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems* 29, 8 (2021), 1505–1517. DOI: http://doi.org/10.1109/TVLSI.2021.3082476

[20] Anıl Bayram Göğebakan, Enrico Magliano, Alessio Carpegna, Annachiara Ruospo, Alessandro Savino, and Stefano Di Carlo. 2024. SpikingJET: Enhancing fault injection for fully and convolutional spiking neural networks. In *Proceedings of the 2024 IEEE 30th International Symposium on On-Line Testing and Robust System Design (IOLTS)*. 1–7. DOI: http://doi.org/10.1109/IOLTS60994.2024.10616060

[21] Priyanjana Pal, Florentia Afentaki, Haibin Zhao, Gurol Saglam, Michael Hefenbrock, Georgios Zervakis, Michael Beigl, and Mehdi B. Tahoori. 2024. Fault sensitivity analysis of printed bespoke multilayer perceptron classifiers. In *Proceedings of the 2024 IEEE European Test Symposium (ETS)*. IEEE, 1–6.

[22] Zheng Cui. 2016. *Printed Electronics: Materials, Technologies and Applications*. John Wiley and Sons.

[23] Jim Handy. 2014. Why Are Computer Chips so Expensive? Retrieved Apr 30, 2014 from https://www.forbes.com/sites/jimhandy/2014/04/30/why-are-chips-so-expensive/#3ee0f8f479c9

[24] Seungjun Chung, Seul Ong Kim, Soon-Ki Kwon, Changhee Lee, and Yongtaek Hong. 2011. All-inkjet-printed organic thin-film transistor inverter on flexible plastic substrate. *IEEE Electron Device Letters* 32, 8 (2011), 1134–1136.

[25] Feng Shao and Qing Wan. 2019. Recent progress on jet printing of oxide-based thin film transistors. *Journal of Physics D: Applied Physics* 52, 14 (2019), 143002.

[26] Farhan Rasheed, Michael Hefenbrock, Rajendra Bishnoi, Michael BeigI, Jasmin Aghassi-Hagmann, and Mehdi B. Tahoori. 2019. Predictive modeling and design automation of inorganic printed electronics. In *Proceedings of the 2019 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 30–35. DOI: http://doi.org/10.23919/DATE.2019.8715159

[27] Haibin Zhao, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2023. Split additive manufacturing for printed neuromorphic circuits. In *Proceedings of the 2023 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 1–6. DOI: http://doi.org/10.23919/DATE56975.2023.10136891

[28] Haibin Zhao, Priyanjana Pal, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2023. Power-aware training for energy-efficient printed neuromorphic circuits. In *Proceedings of the 42nd IEEE/ACM International Conference on Computer-Aided Design*.

[29] Suresh Kumar Garlapati, Tessy Theres Baby, Simone Dehm, Mohammed Hammad, Venkata Sai Kiran Chakravadhanula, Robert Kruk, Horst Hahn, and Subho Dasgupta. 2015. Ink-jet printed CMOS electronics from oxide semiconductors. *Small* 11, 29 (2015), 3591–3596.

[30] F. Rasheed and M. B. Tahoori. 2020. *Compact Modeling and Physical Design Automation of Inkjet-Printed Electronics Technology*. Karlsruhe Institute of Technology (KIT).

[31] Gabriel Cadilha Marques, Dennis Weller, Ahmet Turan Erozan, Xiaowei Feng, Mehdi Tahoori, and Jasmin Aghassi-Hagmann. 2019. Progress Report on "From Printed Electrolyte-Gated Metal-Oxide Devices to Circuits". *Advanced Materials* 31, 26 (2019), 1806483.

[32] Haibin Zhao, Tobias Röddiger, and Michael Beigl. 2021. Aircase: Earable charging case with air quality monitoring and soundscape sonification. In *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*. 180–184.

[33] Dennis D. Weller, Michael Hefenbrock, Michael Beigl, Jasmin Aghassi-Hagmann, and Mehdi B. Tahoori. 2021. Realization and Training of an Inverter-based Printed Neuromorphic Computing System. *Scientific reports* 11, 1 (2021), 1–13. DOI: http://doi.org/10.1038/s41598-021-88396-0

[34] Xiaonan Zou, Yong Hu, Zhewen Tian, and Kaiyuan Shen. 2019. Logistic regression model optimization and case analysis. In *Proceedings of the 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*. IEEE, 135–139.

[35] Johannes Schmidt-Hieber. 2020. Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics* 48, 4 (August 2020). DOI: https://doi.org/10.1214/19-aos1875

[36] S. M. Gowda, B. J. Sheu, J. Choi, C. -G. Hwang, and J. S. Cable. 1993. Design and characterization of analog VLSI neural network modules. *IEEE Journal of Solid-State Circuits* 28, 3 (1993), 301–313. DOI: http://doi.org/10.1109/4.209997

[37] Hossein Pourmehrani, Javad Bahrami, Parsa Nooralinejad, Hamed Pirsiavash, and Naghmeh Karimi. 2024. FAT-RABBIT: Fault-aware training towards robustness againstbit-flip based attacks in deep neural networks. In *Proceedings of the 2024 IEEE International Test Conference (ITC)*. 106–110. DOI: http://doi.org/10.1109/ITC51657.2024.00029

[38] Siva Satyendra Sahoo, Anup Das, and Akash Kumar. 2025. *Fault Tolerant Architectures*. Springer Nature Singapore, Singapore. DOI: http://doi.org/10.1007/978-981-97-9314-3_11

[39] Lance Harvie. 2025. How to Implement Fault-Tolerant Designs in Critical Electronics Applications. (2025). Retrieved May 26, 2025 from https://medium.com/%40lanceharvieruntime/how-to-implement-fault-tolerant-designs-in-critical-electronics-applications-567c2e9bc2c4

[40] Shashikiran Venkatesha and Ranjani Parthasarathi. 2024. Survey on redundancy based-fault tolerance methods for processors and hardware accelerators - trends in quantum computing, heterogeneous systems and reliability. *ACM Computing Surveys* 56, 11, Article 275 (2024), 76 pages. DOI: http://doi.org/10.1145/3663672

[41] Larkhoon Leem, Hyungmin Cho, Hsiao-Heng Lee, Young Moon Kim, Yanjing Li, and Subhasish Mitra. 2010. Cross-layer error resilience for robust systems. In *Proceedings of the 2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 177–180. DOI: http://doi.org/10.1109/ICCAD.2010.5654129

[42] Lixue Xia, Mengyun Liu, Xuefei Ning, Krishnendu Chakrabarty, and Yu Wang. 2019. Fault-tolerant training enabled by on-line fault detection for RRAM-based neural computing systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38, 9 (2019), 1611–1624. DOI: http://doi.org/10.1109/TCAD.2018.2855145

[43] Siva Satyendra Sahoo, Anup Das, and Akash Kumar. 2023. Fault tolerant architectures. In *Proceedings of the Handbook of Computer Architecture*. Springer, 1–44.

[44] Timo Schonwald, Jochen Zimmermann, Oliver Bringmann, and Wolfgang Rosenstiel. 2007. Fully adaptive fault-tolerant routing algorithm for network-on-chip architectures. In *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*. 527–534. DOI: http://doi.org/10.1109/DSD.2007.4341518

[45] Dennis D. Weller, Nathaniel Bleier, Michael Hefenbrock, Jasmin Aghassi-Hagmann, Michael Beigl, Rakesh Kumar, and Mehdi B. Tahoori. 2021. Printed stochastic computing neural networks. In *Proceedings of the 2021 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 914–919. DOI: http://doi.org/10.23919/DATE51398.2021.9474254

[46] Priyanjana Pal, Brojogopal Sapui, Dennis D. Weller, and Mehdi B. Tahoori. 2025. Efficient analog error correction for printed unary-encoded computing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2025), 1–1. DOI: http://doi.org/10.1109/TCAD.2025.3570162

[47] Priyanjana Pal, Haibin Zhao, Tara Gheshlaghi, Michael Hefenbrock, Michael Beigl, and Mehdi B. Tahoori. 2025. *Neural Architecture Search for Highly Bespoke Robust Printed Neuromorphic Circuits*. Association for Computing Machinery, New York, NY, USA. DOI: https://doi.org/10.1145/3676536.3676812

[48] Zihao Deng and Michael Orshansky. 2021. Variability-aware training and self-tuning of highly quantized DNNs for analog PIM. In *Proceedings of the 2022 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 712–717. Retrieved from https://api.semanticscholar.org/CorpusID:244102975

[49] Muhammad Abdullah Hanif and Muhammad Shafique. 2023. eFAT: Improving the effectiveness of fault-aware training for mitigating permanent faults in DNN hardware accelerators. Retrieved from https://arxiv.org/abs/2304.12949

[50] Soyed Tuhin Ahmed, Kamal Danouchi, Christopher Münch, Guillaume Prenat, Lorena Anghel, and Mehdi B. Tahoori. 2023. SpinDrop: Dropout-based Bayesian binary neural networks with spintronic implementation. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 13, 1 (2023), 150–164. DOI: http://doi.org/10.1109/JETCAS.2023.3242146

[51] Fernando Fernandes dos Santos, Niccolo Cavagnero, Marco Ciccone, Giuseppe Averta, Angeliki Kritikakou, Olivier Sentieys, Paolo Rech, and Tatiana Tommasi. 2025. Improving deep neural network reliability via transient-fault-aware design and training. *IEEE Transactions on Emerging Topics in Computing* (2025), 1–12. DOI: http://doi.org/10.1109/TETC.2024.3520672

[52] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. Retrieved from https://arxiv.org/abs/1611.01144

[53] Haifeng Ling, Dimitrios A. Koutsouras, Setareh Kazemzadeh, Yoeri Van De Burgt, Feng Yan, and Paschalis Gkoupidenis. 2020. Electrolyte-gated transistors for synaptic electronics, neuromorphic computing, and adaptable biointerfacing. *Applied Physics Reviews* 7, 1 (2020), 011307.

[54] L. Milor and V. Visvanathan. 1989. Detection of catastrophic faults in analog integrated circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* 8, 2 (1989), 114–130. DOI: http://doi.org/10.1109/43.21830

[55] Farhan Rasheed, Manuel Rommel, Gabriel Cadilha Marques, Wolfgang Wenzel, Mehdi B. Tahoori, and Jasmin Aghassi-Hagmann. 2021. Channel geometry scaling effect in printed inorganic electrolyte-gated transistors. *IEEE Transactions on Electron Devices* 68, 4 (2021), 1866–1871.

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035.

[57] Diederik P. Kingma and Jimmy Ba. 2017. *Adam: A Method for Stochastic Optimization*. Retrieved from https://arxiv.org/abs/1412.6980

Received 28 July 2025; revised 28 July 2025; accepted 29 July 2025