Int. J. Electron. Commun. (AEÜ) 186 (2024) 155467

Contents lists available at ScienceDirect

Int. J. Electron. Commun. (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

An ultra-low power adjustable current-mode analog integrated general purpose artificial neural network classifier

Vassilis Alimisis [*], Andreas Papathanasiou, Evangelos Georgakilas, Nikolaos P. Eleftheriou, Paul P. Sotiriadis

Department of Electrical and Computer Engineering, National Technical University of Athens, 15780 Athens, Greece

ARTICLE INFO

Keywords:
Analog integrated circuits and systems
Artificial neural network
Adjustable system
Low-power design
Current-mode design

ABSTRACT

This study introduces a methodology tailored to analog hardware architecture for implementing an artificial neural network. The fundamental components of the architecture include current-mode circuits, representing the class, and a voltage-mode comparator. Specifically, the current mode circuits comprise the Mahalanobis distance circuit, Sigmoid function circuit, analog multiplier, and current mirrors. Regarding the voltage comparator, which receives the final decision, a folded-cascode operational amplifier is employed. The operational principles of the architecture are extensively explained and applied in a power-efficient configuration (operating under 976nW) with low power supply rails (0.6 V). The proposed implementation is tested on real-world biomedical classification tasks, achieving classification accuracy exceeding 91.6%. The designs are implemented using a 90 nm CMOS process and developed using the Cadence IC Suite for both schematic and layout design. Monte-Carlo analysis, encompassing both process and mismatch, as well as corner analysis, are provided to confirm the robust characteristics of the proposed classifier. Through comparative analysis of post-layout simulation results with an equivalent software-based classifier and related literature, the proper operation of the proposed architecture is confirmed.

1. Introduction

Driven by the increasing integration of Machine Learning (ML) and Artificial Intelligence (AI) in bioengineering [1–3], this study explores the synergy between innovative hardware solutions and biomedical applications. As ML and AI revolutionize research, diagnostics, and treatment approaches in bioengineering [1], the need for sophisticated hardware architectures becomes paramount [4,5]. These architectures, ranging from high-performance computing systems [6,7] to specialized hardware accelerators [8,9], complement the cognitive capabilities of ML and AI, enabling the processing of extensive datasets and real-time analyses inherent in bioengineering tasks. By seamlessly integrating data acquisition, processing, and feedback mechanisms, these hardware advancements expedite the execution of ML and AI algorithms and pave the way for novel methodologies [10].

Analog computing emerges as a promising avenue to augment ML methodologies in biomedicine, addressing the computational demands of complex tasks with precision, energy efficiency, and low latency [11,12]. Leveraging the capacity of analog computing to process continuous signals in real-time, researchers can achieve faster and more energy-efficient ML inference for applications such as real-time diagnostics and wearable health monitoring [11,12]. The fusion of analog computing with ML holds promise for unlocking novel insights, facilitating faster decision-making, and enhancing the overall efficiency of data-driven medical interventions.

In addition to analog computing [11,12], another major trend is soft computing [13,14], which introduces a new methodology for enhancing computational efficiency and flexibility in solving various tasks. Furthermore, through parallelization and real-time data management, soft computing gains additional capabilities through analog computing. Having already methodologies to handle uncertainty, fuzziness, and non-linearity, it now adds new concepts to integrate them into real-world classification tasks [13,14]. Thus, we can design new hybrid architectures that combine the advantages of both, capable of being widely used in dynamic environments. This leads to new possibilities for developing novel machine learning methods.

Motivated by the efficiency requirements of biomedical smart sensor systems [15,16], this study proposes an alternative, low-power, and analog integrated architecture based on a artificial neural network (ANN). Demonstrating considerable promise as a classifier suitable for battery-dependent biomedical smart sensor classification systems, the implemented design attains high accuracy. Implemented design

[*] Corresponding author.
E-mail address: alimisisv@gmail.com (V. Alimisis).

https://doi.org/10.1016/j.aeue.2024.155467
Received 31 May 2024; Accepted 1 August 2024
Available online 3 August 2024

demonstrates proper operation, validated using real-world biomedical datasets. Post-layout simulations in a TSMC 90 nm CMOS process via Cadence IC Suite validate the accuracy of the devised implementation. A comprehensive comparative analysis with related analog classifiers is incorporated to ensure thoroughness. Our approach deviates from existing ones in the sense that it can handle a large number of features without the need of Principal Component Analysis (PCA) [17], more than 20, the related currents of the classifier can be adjusted as low as necessary just to ensure proper circuit operation (high accuracy and low noise), the weights of each feature can be tuned independently and pure current-mode analog circuits are combined to implement the activation functions (more than one).

In the literature there is a variety of analog hardware classifiers including: a Manhattan distance network [18], a Fuzzy [19], a Gaussian mixture model (GMM) [20], a Radial Basis Function (RBF) [21], a RBF-Neural Network (NN) [22–24], a Artificial Neural Network (ANN) [25], Bayes [26], Support Vector Machine (SVM) [27,28], a K-means [29], a Support Vector Regression (SVR) [30], a Support Vector Domain Description (SVDD) [31], a Self-Organized Map (SOM) [32], a Long Short-Term Memory (LSTM) [33], a Multilayer Perceptron (MLP) [34], a Threshold [35], a cascaded-connected Centroid [36], a Spiking Neural Network (SNN) [37,38] and a Pattern-Matching (PM) classifier [39].

Compared to this work, related studies [19–21,26–28,32,35,36] lack the ability to control weights for each separate feature; instead, they can only adjust the overall probability for the entire class. Also, in comparison with RBF-NN (Gaussian function as activation function) [22–24], this work employs a variety of activation functions and hidden layers. The ANN designed in [25] is a simple implementation which is based on current mirrors and common source amplifiers, which approximates the behavior of linear activation functions. Last but not least, this work differs from [18] in terms of architecture, mathematical model, training methods, and design procedure. We will further analyze the novelty in Sections 4–6.

The remainder of this study is structured as follows: Section 2 delves into the essential background of the ANN. A literature review based on both traditional approaches and next generations AI hardware processors is provided in Section 3. Section 4 presents an analysis of the proposed classifier's architecture and transistor-level implementations. Section 5 describes the training and tuning capabilities of the proposed ANN. Section 6 confirms the proper operation of the proposed ANN by employing real-world biomedical datasets and compare with a software-based equivalent. Section 7 provides a comparison study with related analog classifiers, and Section 8 concludes with final remarks.

2. Artificial neural networks

Artificial Neural Networks (ANNs) have emerged as a fundamental of modern artificial intelligence, drawing inspiration from the neural structures of the human brain to process and interpret complex data [40]. ANNs consist of chains of interconnected nodes or neurons, with each layer transforming incoming data through weighted connections and activation functions to produce output [41]. This architecture allows ANNs to learn from data, making them highly adaptable for a variety of tasks, including image and speech recognition, natural language processing, and autonomous systems [42]. As both the numbers and complexity of data increase, we have reached a point where the use of ANNs is necessary to obtain significant results, something that has become evident across various sectors.

The fundamental methodology of training an ANN involves a series of steps designed to optimize the network's performance on specific tasks [43]. Initially, data is fed into the input layer, where it undergoes a series of transformations through one or more hidden layers, each equipped with numerous neurons. Each connection between neurons is associated with a weight that is adjusted during the training process. Training typically employs the backpropagation algorithm, which calculates the gradient of the loss function – measuring the difference between predicted and actual outputs – and updates the weights to minimize this loss. This iterative process, guided by optimization techniques such as gradient descent and its variations, allows the network to gradually enhance its accuracy and predictive power.

ANNs have achieved remarkable results in a variety of applications and fields, ranging from medical diagnostics to economic studies and forecasts [42]. The ability to detect anomalies, tumors, and pathological conditions has improved the accuracy of many new techniques that utilize ANNs. Consequently, it has led to the advancement of medical imaging, for instance. In economics, from the other side, they aid in predicting market trends and managing risks by analyzing historical data and recognizing patterns that may not be immediately apparent to human analysts. Despite their advantages, ANNs also face significant challenges, including the need for large amounts of labeled data, substantial computational resources, and the risk of overfitting, where the network performs well on training data but poorly on unseen data. Ongoing research focuses on addressing these challenges through innovations in network architectures, regularization techniques, and efficient training algorithms, aiming to make ANNs more resilient, efficient, and widely applicable [44].

For the implementation of any algorithm, machine learning model, or ANN using analog integrated circuits, there must first be a proper mathematical modeling of the respective classifier [45]. A feedforward ANN can be mathematically represented based on the following methodology: Let L be the number of layers in the ANN, including obviously the input and output layers. We assume there is an index l for the various layers, where $l = 1$ for the input layer and $l = L$ for the output layer. Let n_l be the number of neurons in a layer l. As we will analyze below for our implementation, we will have $n_l = N_d$, which relates to the number of features. To connect layer l to the next layer $l + 1$, the weight matrix $\mathbf{W}^{(l)}$ is used. Also, $b^{(l)}$ is defined as the bias vector for layer l. Similarly, the activation vector for layer l is denoted as $\alpha^{(l)}$.

Based on the above description, let there be an input vector x, then the computations are carried out as follows for each layer [45]. For the input layer, we have $l = 1$ and activation vector $\alpha^{(1)} = x$. Similarly, the hidden layers for $l = (2, 3, L - 1)$ are given by:

$$z^{(l)} = \mathbf{W}^{(l-1)}\alpha^{(l-1)} + b^{(l-1)} \tag{1}$$

$$\alpha^{(l)} = g(z^{(l)}) \tag{2}$$

where: $z^{(l)}$ represents the weighted sum of inputs to each neuron in layer, $g()$ is the activation function applied element-wise to the elements $z^{(l)}$. Similarly, for the output layer $l = L$, the corresponding description equations are:

$$z^{(L)} = \mathbf{W}^{(L-1)}\alpha^{(L-1)} + b^{(L-1)} \tag{3}$$

$$y = \alpha^{(L)} = g(z^{(L)}) \tag{4}$$

where y is the prediction of the ANN. During the training process, by minimizing the cost function $J(\mathbf{W}, b)$ through techniques like gradient descent, the ANN learns the optimal values for the weights W and parameters b. Thus, it achieves the desired classification accuracy without overfitting.

3. ANNs from traditional approach to next generations hardware

ANNs have gained significant attention in recent years, as they are now not only implemented as pure software solutions but also exist in software-hardware co-design architectures and fully hardware implementations [46,47]. This has been facilitated by their utilization across a plethora of applications, ranging from digit classification to autonomous vehicles [42]. Through ANNs, we have been led to the complete automation of decision-making, even in complex tasks.

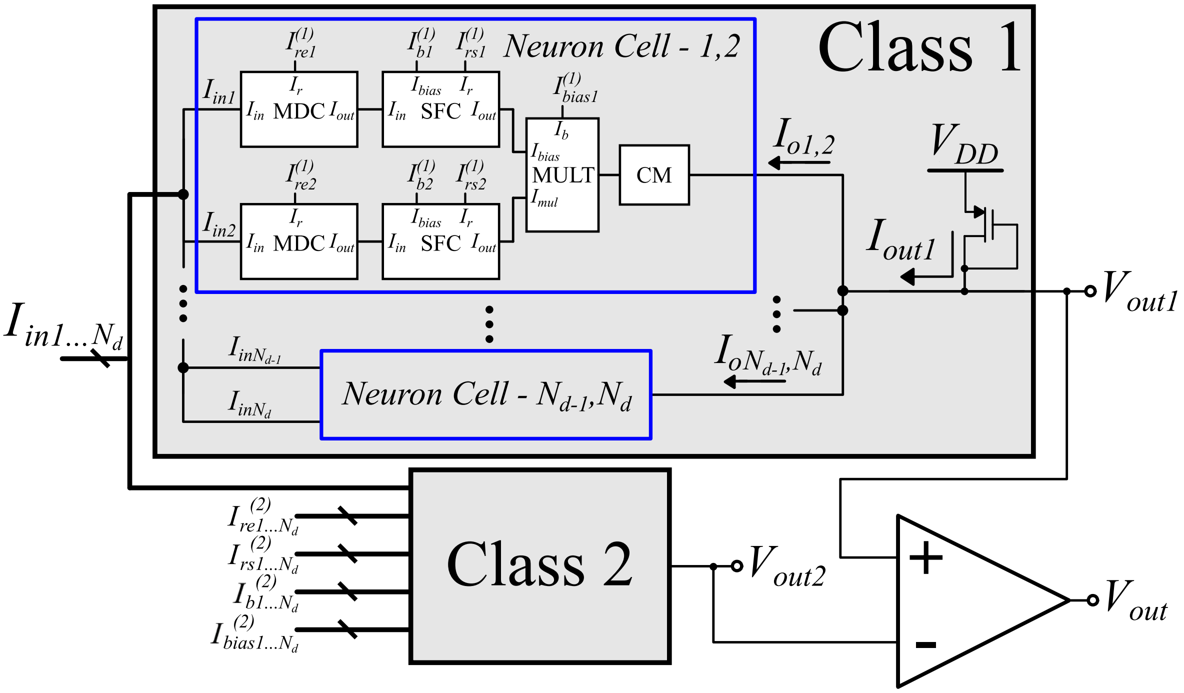

Fig. 1. The high level architecture of the analog integrated ANN classification model implemented with N_d MDCs, N_d SFs, $\frac{N_d}{2}$ analog multipliers and $\frac{N_d}{2}$ CMs for each class and a voltage comparator.

Starting from the software domain, there are several papers in recent years that examine ANNs [42,48,49]. The application and complexity of each implementation vary and are related to the domain of interest. Based on a related survey, the majority of ANNs are employed in management, education and science [42]. Furthermore, in software, we can implement various types of ANNs, including; feed-forward Neural Networks (NNs), recurrent NNs, convolutional NNs, Long Short-Term Memory Networks (LSTMs), Generative Adversarial Networks (GANs) and Auto-encoder Neural Networks (AENN) [50–52]. Feedforwared NNs is the simplest type because the information flows in one direction, from the input to the output. It consists of input, hidden and output layers. Recurrent NNs exhibit temporal dynamics behavior because their connections form a cycle. They typically use sequential data. Convolutional NNs consist of fully connected, pooling and convolutional layers. They apply convolutional operations to identify patterns and as a result they are employed computer vision tasks. LSTMs are an expansion of Recurrent NNs which are suitable for learning from sequences of data. GANs are complex models, which consists of two NNs, a discriminator and a generator. They are suitable for data augmentation and image generation. AENNs combines two types of networks an encoder and a decoder which first maps and then reconstruct the data. They are typically used for dimensionality reduction (e.g feature learning).

Regarding hardware implementations, there is a variety of implementations based on different approaches. Graphics Processing Units (GPUs) are widely used for accelerating neural network training and inference due to their highly parallel architecture [53–55]. They excel at performing matrix multiplications and other compute-intensive operations commonly found in deep learning algorithms. Also, Field-Programmable Gate Arrays (FPGAs) offer flexibility and reconfigurability, making them suitable for implementing custom neural network architectures and accelerating specific tasks [56–58]. They can be programmed to efficiently execute neural network operations in parallel, providing low-latency and energy-efficient solutions. An interesting solution but power-hungry in comparison with analog one, is digital Application-Specific Integrated Circuits (ASICs) which are custom-designed integrated circuits optimized for specific tasks, including neural network computation [59–61]. Digital ASICs can offer high performance and energy efficiency by implementing dedicated hardware modules for neural network operations. Moreover, memristors are emerging non-volatile memory devices that can be used to implement neural network synapses and weights [62–64]. They offer advantages such as low power consumption, high density, and analog behavior, making them suitable for neuromorphic computing and analog neural network implementations. An alternative hardware approach consists of neuromorphic chips, which mimic the structure and functionality of biological neural networks, typically consisting of spiking neurons and synaptic connections [65–67]. These chips can efficiently process temporal data and are well-suited for tasks such as pattern recognition and sensory processing. Last but not least, analog ICs leverage continuous voltage signals to perform neural network computations, offering potential advantages in terms of energy efficiency and scalability [68–70]. They can implement neural network models with high analog precision, enabling efficient hardware implementations of certain algorithms.

4. Proposed artificial neural network's architecture

In this section, we will present and analyze the proposed architecture along with its fundamental structural elements. Aimed at a more design-oriented approach, we will elucidate the reasons behind selecting specific power supply, structural components, and dimensions to achieve the desired specifications. With the objective of attaining the lowest possible power consumption coupled with high accuracy and processing speed, we have introduced this particular architecture in conjunction with the design specifications and fundamental structural elements for implementing the classifier.

4.1. High-level architecture

The proposed architecture is based on a hardware-friendly approximation of the mathematical modeling of ANN. As depicted in Fig. 1, the proposed architecture consists of 2 classes and N_d features. This means it can classify binary classification tasks with any desired number of features. The activation function used is the Euclidean distance, which for correlation reasons with our previous work [18], we consider to be approximated by the Manhattan distance. The Manhattan distance circuit (MDC) approximates this activation function. Through this specific function, the ANN can learn the desired relationship between input data and various outputs [45]. To achieve higher accuracy, the implemented model also utilizes a second layer where the activation function is the sigmoid function [45]. The sigmoid function is suitable for binary classification, yielding better results compared to other activation functions when outputs need to have a probabilistic nature. Additionally, in this implementation, the weight change multiplicative factor is used as a parameter that regulates how the input of a neuron is combined with the inputs of previous neurons.

The output of each sigmoid function circuit (SFC) is used as input to each of the analog multipliers (Mult). Each pair of features has a common analog multiplier that is fed with the two output currents of the corresponding SFC. Based on Fig. 1, the output of the first SFC enters the I_b terminal, and the output of the second enters the I_{mul} terminal. The primary use of the analog multiplier is to correlate the outputs of individual features [71]. The first step was to find the correlation coefficient and covariance for each pair of features. If both features have high output, then their combined output (multiplier's output) will be high. Conversely, it will be low if both have low output. In the case where one has low output and the other has high output, then the multiplier's output will be low. This characteristic is exploited by this architecture to reduce the current introduced into the cascode current mirrors (CMs). Additionally, this block has the capability to adjust the weight if the desired value has not been achieved by the previous layer.

The summation on the classes' output node is carried out through the cascode CMs. These are utilized in order to minimize potential distortions in the calculations that might arise from undesirable effects on the output currents of the multiplier. The output currents from each feature are collected together at a shared node and subsequently directed into a PMOS diode. This process transforms the cumulative output current of class $k = 1, 2$ into the corresponding output voltage, V_{out_k}. The classifier's prediction is determined by an operational amplifier operating in an open-loop configuration, which compares the output voltages of the two classes. The resulting output voltage saturates either to V_{DD} or V_{SS}, depending on the victorious class. All building blocks operate in the sub-threshold region with power-supply rails $V_{DD} = -V_{SS} = 0.3$ V.

4.2. Main building blocks

The first circuit to be analyzed is the MDC which approximates the euclidean distance [18]. The circuit implementation of the MDC is depicted in Fig. 2 and consists of both NMOS and PMOS CMs as well as the translinear loop. Based on techniques from both sub-threshold region and the use of the translinear loop, it emerges that the output current is given by: $I_{out} = \|I_{in} - I_r\|$. As it is a current-mode implementation, both current subtraction and the final output can be achieved simply by connecting wires. To ensure robust mirroring even for very small currents, cascode CMs were employed. By using cascode CMs the channel-length modulation effect (Early effect) is reduced and the quality of the mirroring is increased. Also, the cascode configuration provides some level of immunity to noise and interference, improving the signal-to-noise ratio and overall performance of the circuit. Additionally, the selection of the range for the values of the currents I_r and I_{in} was made with the aim of minimizing power consumption while optimizing circuit operation. The transistor dimensions are equal to $(W/L) = \frac{400 \text{ nm}}{1600 \text{ nm}}$ (for NMOS) and $(W/L) = \frac{1600 \text{ nm}}{1600 \text{ nm}}$ (for PMOS). The behavior of the output current I_{out} as a function of the input current I_{in} for different values of the current I_r is illustrated in Fig. 3.

In this subsection, we present an alternative approach to constructing a SFC [72]. The introduced SFC comprises three primary sub-modules: an NMOS cascode current mirror, a PMOS current mirror, and an NMOS Winner-takes-all (WTA) circuit [73]. In a standard SFC, the differential pair is commonly utilized. However, in this implementation, it is replaced by the NMOS WTA to achieve a more pronounced Sigmoid function curve. The WTA circuit is preferred due to its superior linearity when compared to a conventional differential pair. The depicted illustration of the proposed SFC can be found in Fig. 4. The NMOS WTA is constructed using four NMOS transistors with $(W/L) = \frac{400 \text{ nm}}{1600 \text{ nm}}$.

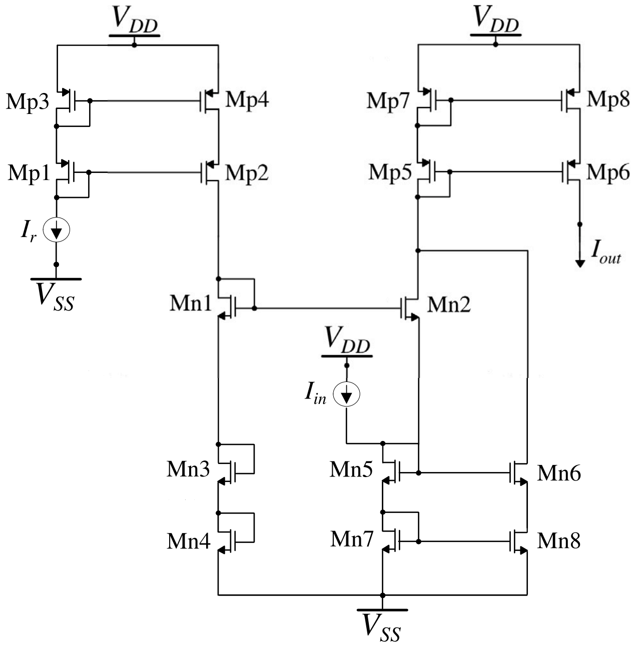

Fig. 2. The implementation of the Manhattan distance with analog integrated circuits. It is a low-power implementation which operate in sub-threshold region. The lowest output value is achieved when input current I_{in} is equal to I_r.

Initially, the WTA operates in the sub-threshold region for inputs I_{in} and I_r and bias current I_{bias} [73]. In an ideal operating scenario, if $I_{in} > I_r$, then the outputs would be $I_{on2} = I_{bias}$ and $I_{on1} = 0$, and conversely, $I_{on1} = I_{bias}$ and $I_{on2} = 0$ for $I_{in} < I_r$. However, due to the circuit operating in the linear region for close values of I_{in} and I_r, their output behaves akin to an exponential function with I_{in} as the variable. The SFC's transistor dimensions are equal to $(W/L) = \frac{400 \text{ nm}}{1600 \text{ nm}}$ (for NMOS) and $(W/L) = \frac{1600 \text{ nm}}{1600 \text{ nm}}$ (for PMOS).

Manipulating two circuit parameters, I_{bias} and I_r, achieves the electronic adjustment of the Sigmoid function's height and center. An additional parameter, V_c, linked with bulk-controlled transistors, may be introduced to finely adjust the Sigmoid function's width, though it does not affect classification accuracy for this circuit. These parameters are determined during the classifier's training process, which is executed through software-based implementation. Fig. 5 demonstrates how the bias current I_{bias} controls the resulting Sigmoid output current's height while maintaining a constant $I_r = 5$ nA. On the other hand, Fig. 6 depicts how the mean value of the derived Sigmoid function is altered by the current I_r, with $I_{bias} = 5$ nA held constant. An interesting point is that the output curve of the sigmoid function circuit is a little too steep. This provides an advantage during training procedure because it approximates a step response behavior which provided higher classification accuracy for the specific tasks.

For precise linear scaling, an analog multiplier circuit [72,74], depicted in Fig. 7, is utilized. This multiplier functions based on the translinear principle [72], which dictates that the product of clockwise translinear elements' currents within a translinear loop equals the product of counterclockwise translinear elements' currents derived

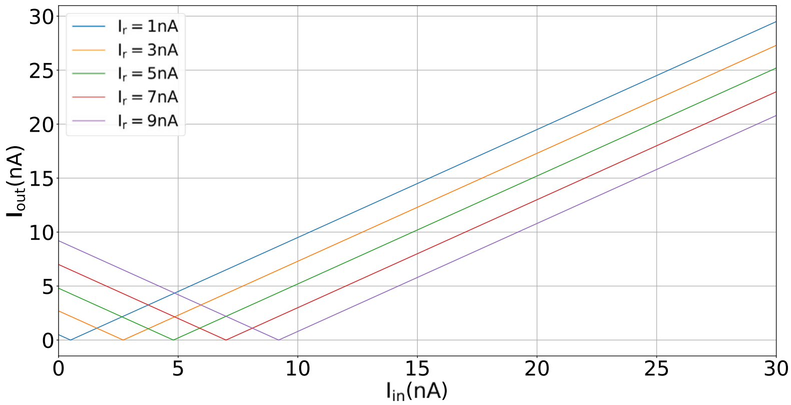

Fig. 3. The output current of the MDC as function of the input current I_{in} and parameterized on I_r.

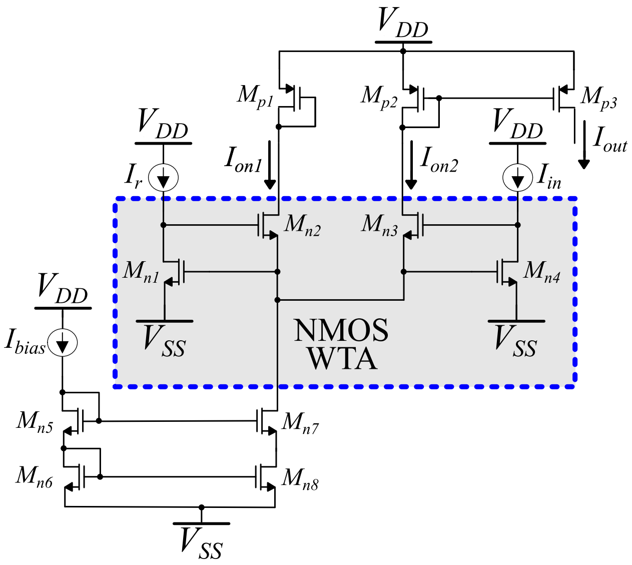

Fig. 4. The high level schematic of the proposed SFC. It consists of a NMOS cascode CM, a NMOS WTA circuit and a PMOS CM. The I_r parameter current tunes the mean value and I_{bias} alters the height of the Sigmoid function curve.

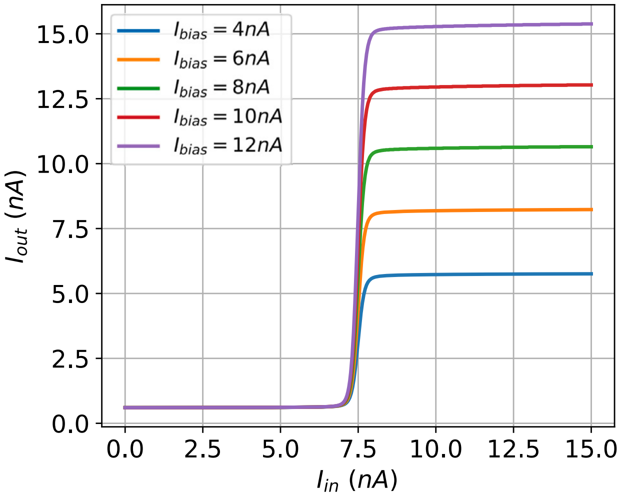

Fig. 5. The output current of the SFC as a function of I_{in} and parameterized on I_{bias}, for $I_r = 5$ nA.

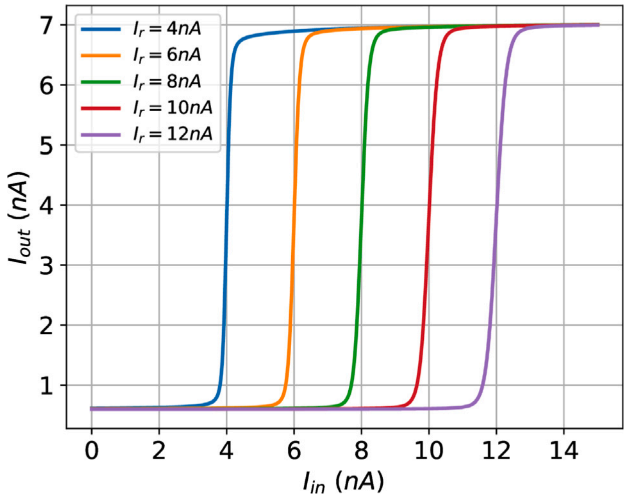

Fig. 6. The output current of the SFC as a function of I_{in} and parameterized on I_r, for $I_{bias} = 5$ nA.

within the same loop. Essentially, in the sub-threshold region of MOS operation, the translinear principle converts the sum of gate-to-source voltages around the loop into a current product. This conversion is facilitated by the exponential characteristics of MOS operation in the sub-threshold region [75], relative to its gate-to-source voltage, which arises from the application of Kirchhoff's voltage law within the loop.

Given that all four transistors (M_{n1}, M_{n2}, M_{n3}, and M_{n4}) are operating within the sub-threshold region and following the translinear principle, the output current of the MP can be represented as:

$$I_{out} = \frac{I_b I_{bias}}{I_{mul}}. \tag{5}$$

In this scenario, I_b and I_{bias} serve as inputs to the analog multiplier circuit, while I_{mul} acts as a constant normalizing current. The presence of transistor M_{n5} is essential for appropriately biasing the translinear loop. Comprehensive dimensions of the transistors within the multiplier circuit are equal to $(W/L) = \frac{2800\ \text{nm}}{2400\ \text{nm}}$ (for NMOS) and $(W/L) = \frac{1800\ \text{nm}}{3200\ \text{nm}}$ (for PMOS).

According to Eq. (5), the output current I_{out} exhibits a linear increase with the rise in currents I_b and I_{bias}, and uniformly decreases with the increment of current I_{mul}. This behavior is further validated by the simulation outcomes. Fig. 8 illustrates the output current of the analog multiplier circuit as a function of I_{bias}, parameterized by I_{mul}, with $I_b = 5$ nA. Fig. 9 demonstrates the output current of the multiplier circuit as a function of I_{bias}, with $I_{mul} = 10$ nA, and parameterized by I_b. Similarly, Fig. 10 presents the output current of the MP circuit as a function of I_{mul}, with $I_{bias} = 5$ nA, and parameterized by I_b.

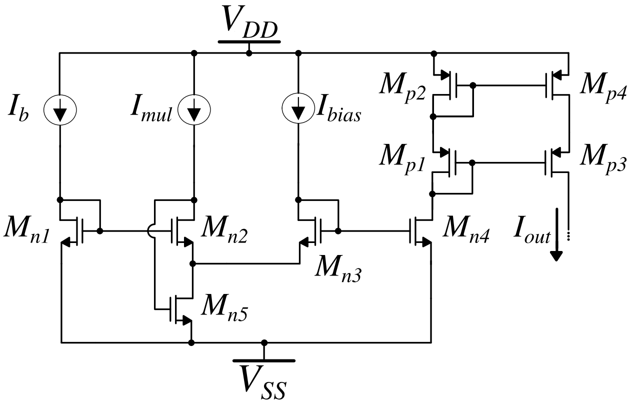

Fig. 7. The implementation of the analog multiplier. It consists of a translinear-loop.

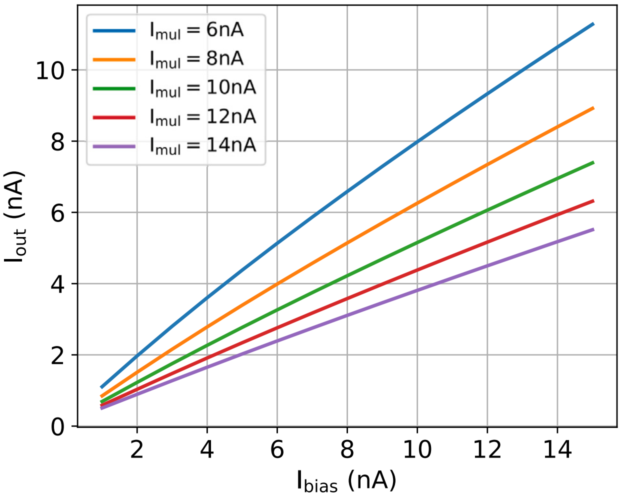

Fig. 8. The output current of the analog multiplier as a function of I_{bias} and parameterized on I_{mul}, for $I_b = 5$ nA.

Apart from parametric sweeps, a transient analysis is conducted to further assess the analog multiplier's performance. Figs. 11 and 12 depict the transient inputs and output of the analog multiplier respectively. In Fig. 12 the ideal value of I_{out} - calculated as shown in Eq. (5) - is also plotted. It can be observed that I_{out} follows its theoretical value with satisfactory accuracy, with the error between them not exceeding a few nA at most.

An intriguing observation is that when I_b is set to a high value (greater than 2 nA) and I_{bias} is set to a low value (e.g., 500 pA), the output current equals the low value. To elaborate, if $I_b = 5$ nA (high) and $I_{bias} = 500$ pA (low), the circuit approximates an "AND Logic Gate", with the output equaling 500 pA (low), rather than 2.5 nA (midpoint).

The folded-cascode operational amplifier was used as the output stage of the classifier. This amplifier is utilized as a voltage comparator, providing either V_{DD} or V_{SS} output depending on the voltages received at the input. Specifically, if the voltage at the positive input is higher than the negative input, the output will have a voltage close to V_{DD}; otherwise, it will have a voltage close to V_{SS}. The topology of the amplifier is illustrated in Fig. 13. The voltages $V_{bias,i}$, where $i = 1, 2, 3, 4$, originate from an additional biasing stage with diode-connected transistors, similar to the implementation in [76]. This topology is chosen because it provides a relatively high gain of approximately 50 dB, its biasing is straightforward, and compensation can be easily achieved using a capacitor at the output. The transistors' dimensions are summarized in Table 1.

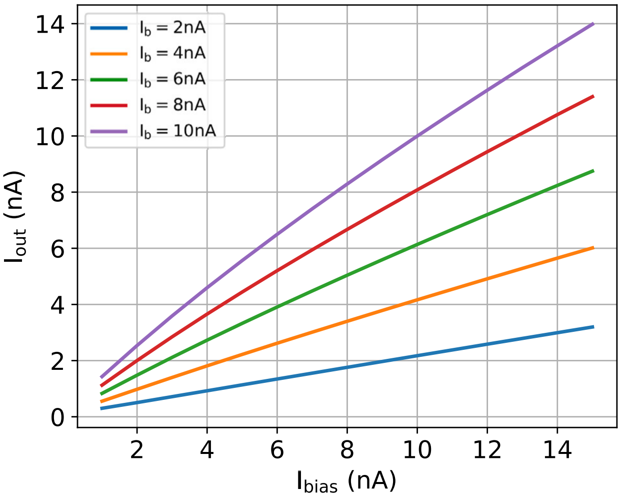

Fig. 9. The output current of the analog multiplier as a function of I_{bias} and parameterized on I_b, for $I_{mul} = 8$ nA.

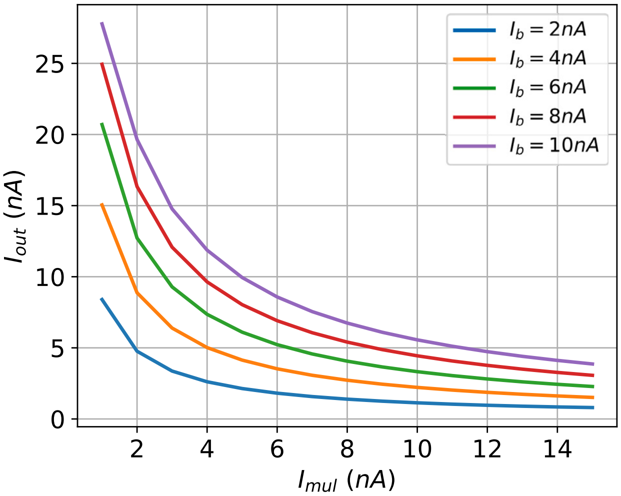

Fig. 10. The output current of the analog multiplier as a function of I_{mul} and parameterized on I_b, for $I_{bias} = 20$ nA.

Table 1
Folded Cascode operational amplifier sizing (Fig. 13).

NMOS	W/L (μm/μm)	PMOS	W/L (μm/μm)
M_{n1}, M_{n3}	1.8/2.0	M_{p1}, M_{p2}	3.2/3.0
M_{n2}, M_{n3}	2.4/2.0	M_{p4}, M_{p6}	4.8/3.6
M_{n2}, M_{n3}	2.4/2.0	M_{p3}, M_{p5}, M_{p7}	3.2/3.6

Another interesting point is the comparison between this work and our previous one [18]. Firstly, [18] involves a set of centroids representing different classes. The network computes the Manhattan distance between input data points and these centroids. Also, it uses centroids and Manhattan distance for straightforward classification. It involves finding the optimal positions of the centroids, often using techniques like k-means clustering with the Manhattan distance metric. Moreover, it is simpler, since it has more interpretable models. It is better for simple classification tasks with clear centroid-based separation.

In contrast, this work introduces an alternative ANN architecture but incorporates Manhattan distance in its computations. More specifically, it is similar to a traditional ANN, but incorporates the Manhattan distance within its computation, either as part of the hidden layers or in the distance computation. Also, the Manhattan distance can be used in various ways, such as in hidden layer transformations or as a

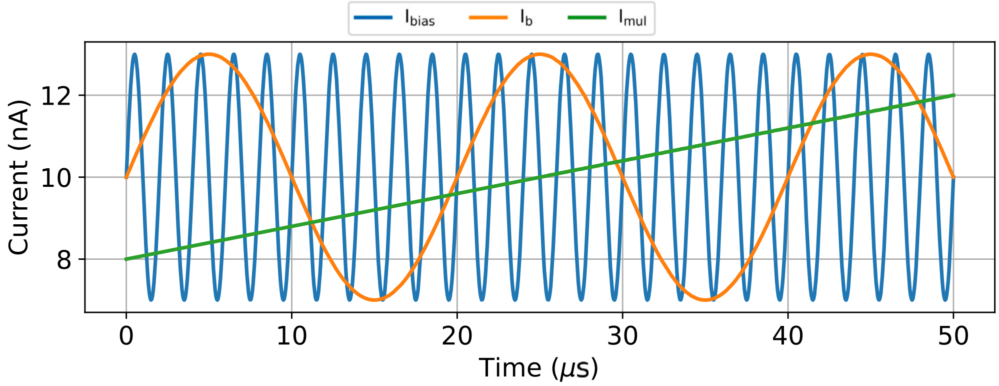

Fig. 11. The input currents applied to the analog multiplier for a transient test.

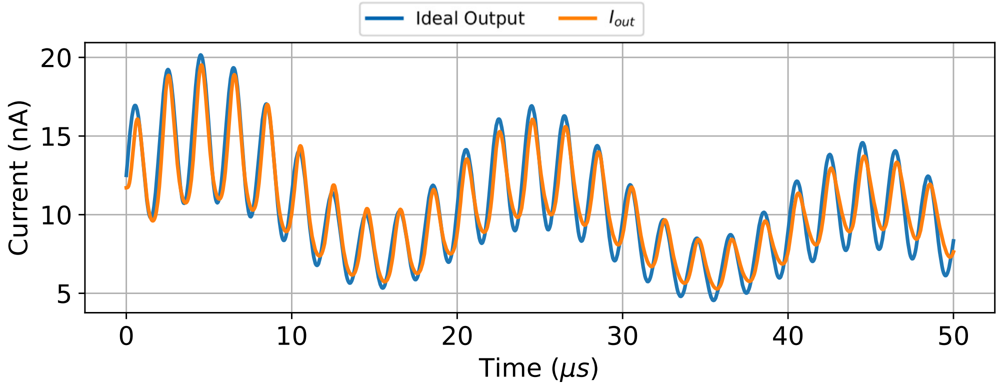

Fig. 12. A comparison between the simulated and ideal transient response of the analog multiplier circuit.

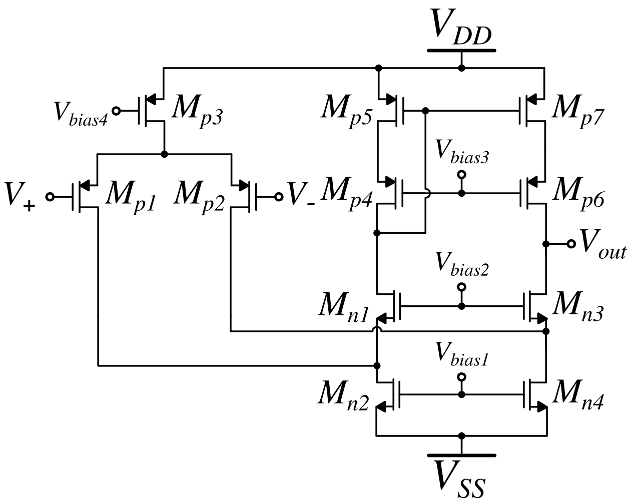

Fig. 13. The folded cascode operational amplifier realizing the voltage comparator of the analog ANN.

criterion in the loss function. Regarding training, it involves backpropagation and gradient descent with Manhattan distance influencing the network's learning process. It is capable of learning complex, non-linear relationships. Last but not least, it is suitable for more complex tasks where the properties of Manhattan distance can be leveraged within a flexible, deep learning framework.

4.3. Design procedure

In this section, we will delve into the process of determining the specifications and design parameters for the proposed architecture. Beginning with the power supply, the selection was guided by considerations of minimizing power consumption and ensuring proper circuit functionality within the sub-threshold region across Process-Voltage–Temperature (PVT) variations [75,77–79]. In particular, for applications demanding low power, the sub-threshold region is the preferred operating range. Here, devices should be biased with V_{GS} voltages nearly equivalent to V_{th} (which rises as temperature decreases due to increased carrier mobility), and $V_{DS} \geq 4V_T$, where $V_T = kT/q$ (temperature-dependent) [75,77–79]. Except from the diode connected transistor which are properly biased when applying the proper dimensions. The implemented blocks are designed with branches comprising a maximum of 3 or 4 transistors (except from the diode connected cases). Considering a maximum temperature of 125° Celsius, the V_T value amounts to 34.322 mV. Consequently, under the worst-case scenario for transistor operation, a difference of $V_{DD}-V_{SS} = 549.152$ mV is necessitated. To provide a margin for mitigating over Voltage variation (e.g., $V_{DD}-V_{SS} = 0.5$ V), we opt for a supply equivalent to $V_{DD}-V_{SS} = 0.6$ V. The decision to employ the same supply for all blocks was driven by the need for one specific power supply.

The process of selecting dimensions for each block is intricate and involves multiple parameters. Primarily, as the width (W) and length (L) of the devices are augmented, there is a corresponding increase in the overall occupied area. The aim, as per previous implementations and literature Refs. [80], is to design the architecture within a combined area smaller than $0.3\ \mathrm{mm}^2$. Both literature and simulations, particularly in the sub-threshold region, advocate for opting for a small W value while maximizing L. Augmenting W (thus increasing conducting channels) results in heightened leakage current, whereas enlarging L tends to mitigate this effect by reducing drain-induced barrier lowering (DIBL) phenomena. Moreover, bias currents were deliberately chosen to significantly surpass corresponding leakage currents [75,77–79]. They also exert an influence on the current flow across each

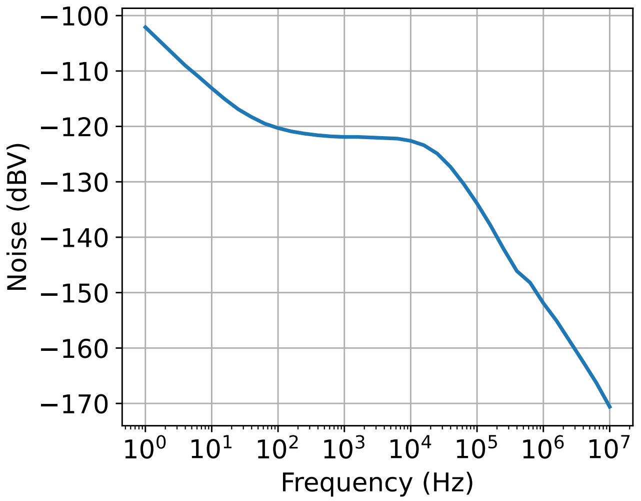

Fig. 14. The noise level for the ANN implementation.

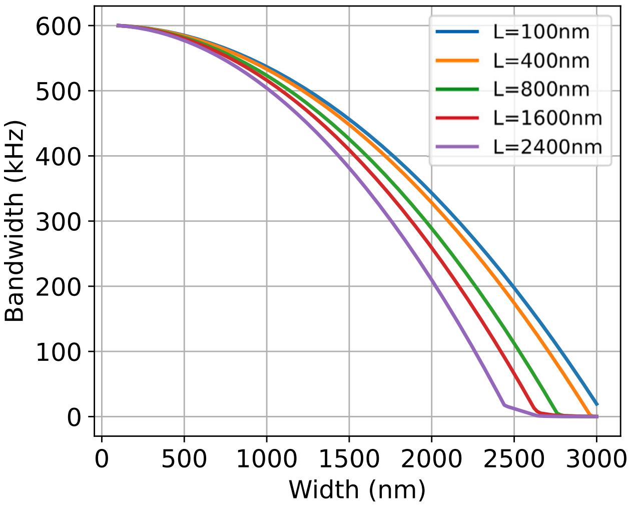

Fig. 15. The effect of the choice of dimensions on the input–output transistors, which create the dominant pole through the parasitic capacitors. In this graph we alter both W and L.

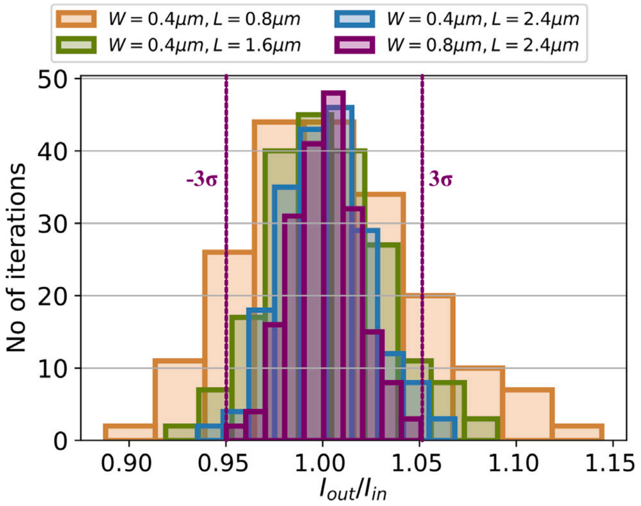

Fig. 16. The effect of mismatch for different values of W and L. For lower sizes, the output current has a large variance in comparison with the input current.

node. Additionally, the small V_{GS} voltage at low current levels, coupled with the relatively diminutive V_{th} in current technology, renders sub-threshold biasing feasible with a large L and comparatively small W. The value of V_{th} escalates with both increasing L and decreasing W (thus diminishing gate-channel capacitance). Furthermore, flicker noise diminishes with enlarging L. The selection of a specific value also hinges on the desired noise level, which, based on transient simulations, seems minimally impacted by data inaccuracies [75,77–79]. More specifically, the RMS current for the signal (square pulse) is equal to $I_{sigRMS} = 5.5$ nA and the maximum RMS current for the noise is equal to $I_{noiseRMS} = 325$ fA (based on simulation results). The simulation results regarding the noise level for the ANN implementation (in the output node) is summarized in Fig. 14. Additionally, as W and L values increase, parasitic capacitances also increase, consequently reducing the desired bandwidth (BW) of the classifier. The effect of the choice of dimensions on the input–output transistors, which create the dominant pole through the parasitic capacitors, is shown in Fig. 15, both with the alteration of W (values) and with the L (values). This results in a reduction in its processing speed, with our target being above a few KHz based on literature [81]. Furthermore, the low supply voltage of the system contributes to a diminished BW. Also, in order to reduce the mismatch between transistors, as described by Pelgrom model [82], we should increase W or L (sub-threshold region effect). Based on the previous steps and mismatch effect, the selection of dimensions was aimed at achieving a minimal variation in current mirroring, specifically targeting less than 5% deviation across PVT variations.

The impact of mismatch for varying values of W and L is illustrated in Fig. 16. Larger device dimensions (such as in a differential pair or current mirror) result in smaller variations between the input and output currents, though this increases the total area. This observation is also supported by the related mathematical model. To achieve a mismatch below 5%, the product $W \cdot L$ must exceed 1.92 μm^2 (for example, $W = 0.8$ μm and $L = 2.4$ μm).

Lastly, it is interesting to mention the design choices made in implementing the output stage of each class in Fig. 1. The argument can be made that for a class with many Neuron Cells a large current can accumulate at the MOS diode depicted in Fig. 1, causing the voltage drop across it to become abnormally high. This could impose an upper limit on the number of synapses that can be connected to a neuron since it can cause problems like degrading the performance of the Neuron Cells' output stage. An abnormally high voltage drop across the MOS diode can also cause the output voltage to saturate to the power rails.

To account for this problem, we drive the output of each Neuron Cell into a NMOS Current Mirror (CM) to buffer the output current of the Cell's analog multiplier. The CM's devices are made wider than typically used in the rest of the design's current mirrors to accommodate for low values of the voltage at their output node. We further compensate for this by using a wide device for the MOS diode to limit the effect a very large drain current can have on its voltage drop, thus increasing the amount of current that can flow from it without causing problems.

By testing the topology with an increasingly high number of features we found that the upper limit where the output stage performed well was reached at a few hundreds of Neuron Cells, which is where the PMOS diode had transitioned from the sub-threshold to the saturation region of operation due to the accumulation of current flowing through it. Since each Neuron Cell is associated with two features it is clear that the architecture can accommodate hundreds of features before degradation of performance is observed.

5. Training and tuning capabilities

The analog integrated ANN described above relies on integrating the main components to function as a distance measure for prototypes within each class (as explained in the previous Section). This configuration permits the electronically adjustable parameters, denoted as

I_r for each feature, to be utilized in fabricating a post-layout classification chip. Furthermore, the associated currents can be adjusted across all components to enhance tolerance at the expense of power consumption (especially for more intricate tasks). This flexibility facilitates easy adaptation to meet the specific demands of the intended application. Moreover, the system's adjustability can effectively tackle a wide range of classification challenges, regardless of factors such as input dimensions (N_d).

To initiate the procedure, we developed a software-based approach to implement the ANN, aiming to collect crucial parameters for the circuit. For consistency, all datasets utilized to validate the classifier were normalized to align with the operational range of the implemented EDC, specifically within the interval of $[3, 9]$ nA, as elaborated in the preceding section. Additionally, the range of I_r for SFC was defined as $[4, 12]$ nA. This introduces an additional degree of freedom in the implementation, as the parameter current of the SFC, I_r, can act as a multiplicative factor affecting weight changes. Consequently, a mapping between the values of I_r and the corresponding weights was established. Subsequently, the software-based classifier underwent a customized training process utilizing a specific methodology. This approach facilitated the extraction of input dimensions for each class, directly correlating with the associated current parameters of the hardware counterpart.

In obtaining the values of I_{mul}, a deliberate decision was made due to the lack of a direct method within the ANN to determine them during the training process. It was opted to assign it an arbitrary value that remained consistent across both classes. The choice of I_{mul} is tied to a balance between accuracy and power consumption. This intentional choice aims to highlight any significant decrease in accuracy in the hardware implementation attributable to the extraction of software-based I_r values for both blocks, simplifying the development process and reducing unnecessary complexity. This step is performed once for each unique application, and the resulting parameters are subsequently exported and stored in analog memory [83].

Training an ANN encompasses various stages [45]. Below, a top-level examination of the process is provided step by step. (1) Initially, the collection of a dataset is required, consisting of inputs (features) and their corresponding labels (correct answers) for each example. (2) Next, the dataset is divided into two subsets: a training set and a validation set. The training set is used to train the model, while the validation set is used to evaluate its performance on independent data. (3) Then, the appropriate neural network architecture needs to be selected, which includes the number of layers and neurons in each layer. (4) Training of the neural network using the training set. During training, the network weights are adjusted to minimize the deviation between predictions and actual values. (5) Evaluation of the trained model's performance on the validation set to assess its ability to generalize to new data. (6) Adjustment of the model's hyperparameters (such as learning rate and number of training epochs) to optimize performance. (7) Iteration of the previous steps if necessary to achieve the desired performance. Pruning techniques are applied to reduce its size and complexity. This can help prevent overfitting. Finally, due to the abundance of data, a 70–30% training-test split was utilized.

The previous steps are different in comparison with [18] in which we have the following procedure. (1) Normalize and scale the digital dataset to fit within the circuit's operational current range of [4,9] nA. This ensures compatibility with the analog circuit's limits and prevents operational issues. (2) Use the preprocessed datasets to train a software-based centroid classifier. This classifier serves as a reference for the hardware implementation and computes the necessary parameters, such as the centroid vectors for each class. (3) Once the software classifier is trained, extract the centroids and the processed dataset. Translate these centroids into the current settings for the hardware implementation, setting the centroids as current parameters I_r and the processed data as current inputs I_{in} in the analog circuit. (4) Test the analog classifier using a subset of the dataset to verify its accuracy and alignment with the software-based model. Analyze any discrepancies identified during testing and make necessary adjustments to the parameter mappings to enhance the classifier's performance. (5) Optimize the analog centroid classifier to achieve high classification accuracy while maintaining low power consumption.

In this study, each feature operates independently of the others, except for those that are correlated. Consequently, if a generalized implementation is supplied, it can readily adjust the number of input dimensions. For an implementation featuring N_d features, it becomes feasible to deactivate $N_d - 1$ input features either by biasing each block with zero currents or by assigning I_{in} values significantly different from I_r (for instance, setting $I_{in} = 3$ nA and $I_r = 9$ nA). Furthermore, deactivating the entire classifier is straightforward by employing the aforementioned technique for all classes and input dimensions.

6. Application examples and simulation results

In this section, we will present the applications in which the classifier was tested along with the simulation results of both the software-based implementation and the proposed hardware-friendly ANN. Our previous implementation [18] was tested on a dataset related to Chronic Kidney disease. Similarly, the proposed architecture will be tested on biomedical datasets, aiming to create a generalized range of applications that will contribute to medical diagnosis and personalized medicine (understanding the needs of each patient for a variety of diseases based on their characteristics). The first dataset, Echocardiogram dataset, originates from a medical research group based in Long Island, New York, and readers can access it through the University of California, Irvine (UCI), Machine Learning Repository [84]. The second dataset, Primary Tumor dataset, contains information related to patients with primary tumors, focusing primarily on the colon and rectal cancer domain. Readers can access it through the University of California, Irvine (UCI), Machine Learning Repository [85].

Regarding the first dataset, the Echocardiogram dataset comprises 132 instances with 12 attributes, encompassing both numerical and categorical data [84]. More specifically, this dataset consists of two classes and $N_d = 9$ features. These attributes include clinical measurements such as age, sex, and echocardiographic parameters like fractional shortening and E-point septal separation. Additionally, the dataset contains information on left ventricular dimensions, wall motion scores, and the number of major vessels affected by narrowing or blockage. One crucial attribute, "Alive-at −1", serves as the target variable, indicating whether patients survived for at least one year following their echocardiogram examination. With its inclusion of diverse patient characteristics and cardiac measurements, this dataset serves as a valuable resource for developing predictive models to assess patient survival outcomes based on echocardiographic findings and clinical parameters.

The second dataset, called Primary Tumor dataset, offers valuable insights into the domain of colon and rectal cancer [85]. Originating from the Department of Surgery at the University of Ankara and the Department of Information and Computer Science at Ege University in Turkey. This dataset comprises attributes (17 features) detailing diverse characteristics of primary tumors. These attributes encompass clinical and pathological features such as tumor size, location, histological type, and lymph node involvement. Each instance in the dataset represents a patient diagnosed with a primary tumor, providing clinicians and researchers with a comprehensive view of tumor-related information. Commonly employed for classification tasks, the dataset serves as a foundational resource for developing predictive models to assess tumor malignancy based on its distinctive attributes. More specifically, this dataset consists of two classes and $N_d = 17$ features. It remains an invaluable asset for advancing research in oncology and enhancing understanding of primary tumor behavior and prognosis.

The concept here is that some of the data (e.g. in the echocardiogram dataset) are long-term measurements or medical exam results

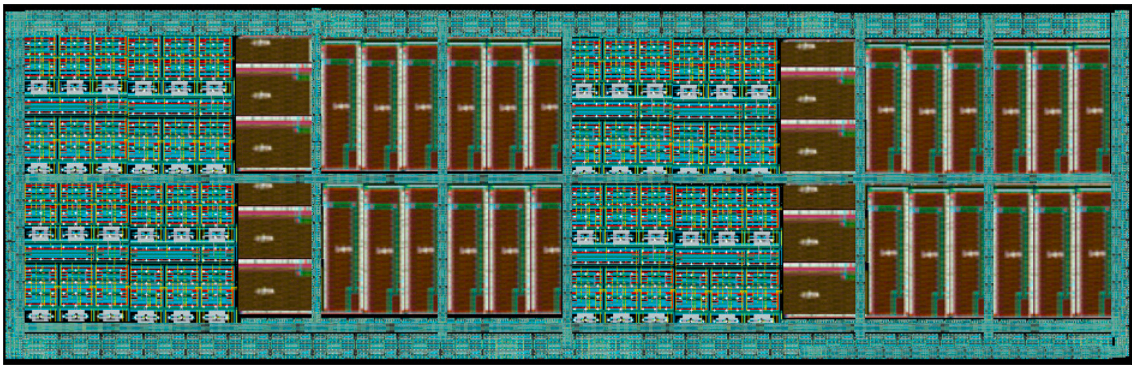

Fig. 17. The implemented layout of the proposed ANN classifier. It consists of extra dummy transistors.

Table 2
Classification results on both datasets over 100 iterations. The first two lines are related to the first dataset and the last two lines are related to the second dataset respectively.

Method	Best (%)	Worst (%)	Mean (%)	Std. (%)
Software	100.0	97.2	98.7	0.93
Proposed	99.7	96.8	98.4	1.01
Software	97.7	93.2	95.4	1.13
Proposed	97.2	92.1	94.9	1.51

which can be considered as short-time constants. At the same time, real-time (critical) data like ECG can be provided on-the-fly by a smart watch or similar wearable device. As a result, this low-power analog classifier can be used as a basic candidate for low-power wearable biomedical applications. This is critical as it frees the system from the need for additional power-hungry circuits such as converters (no need for ADC and DAC).

The two datasets do not contain the same number of features; therefore, the implemented layout will consist of 2 classes and 17 features with the aim of being able to incorporate both classification tasks. For the implementation of the first dataset, which contains 9 features, we will use the techniques mentioned in the previous section to disable the extra blocks. The design, simulation, and layout procedures were carried out using the Cadence IC suite within a TSMC 90 nm CMOS process. The implementation of the layout, as depicted in Fig. 17, utilizes the common-centroid technique. Moreover, extra dummy transistors have been integrated to minimize discrepancies and cater to manufacturing concerns. The total area measures 0.127 mm^2, encompassing the classifier and the biasing circuits within the layout. We should note that the software's simulation results are compared with the post-layout simulation result.

To minimize the potential occurrence of over-fitting, the ANN classifier underwent the training–testing procedure 100 times, with the summarized results presented in Table 2 and Figs. 18 and 19 for both datasets. Specifically, to address random effects introduced by the training algorithm, 100 separate software-based training iterations were conducted to extract the requisite parameters of the ANN. As demonstrated, the proposed implementation achieves nearly perfect accuracy results on the best train–test split for both datasets. Moreover, the comparison between software and hardware validates the precision of the topology, with a sub-1% decrease observed between mean software and hardware performance for both datasets. It is worth noting a slight decrease in hardware accuracy compared to software, primarily due to the circuits generating an approximation of the requested functions rather than an ideal representation. However, the training of the parameters lays the groundwork for their ideal model.

In addition to the 100-iteration test, the designed analog circuits need to undergo testing for their sensitivity to PVT variations. Therefore, a Monte-Carlo analysis, considering process and mismatch variations, was conducted with N = 500 distinct points (equivalent to 6σ). The comprehensive results are outlined in Figs. 20 and 21, with their statistical attributes presented in Table 3 for both datasets. The proposed architecture demonstrates robustness, maintaining a worst-case accuracy above 91.6% for both datasets. Simultaneously, the calculated variance in both cases remains below the 1% threshold,

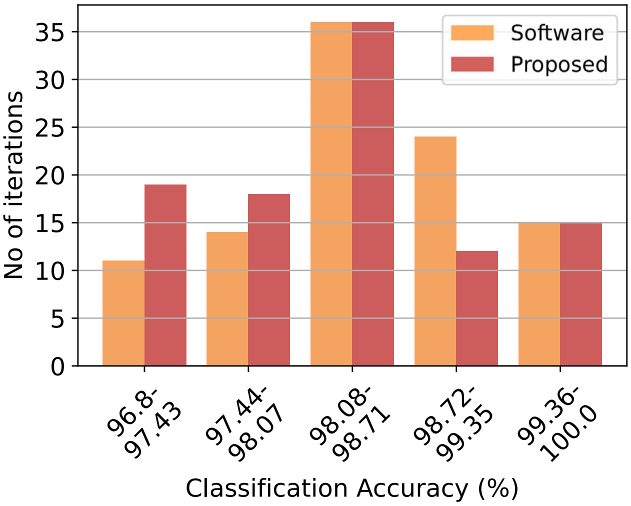

Fig. 18. Classification results of the introduced architecture and its software counterpart on the first dataset across 20 iterations.

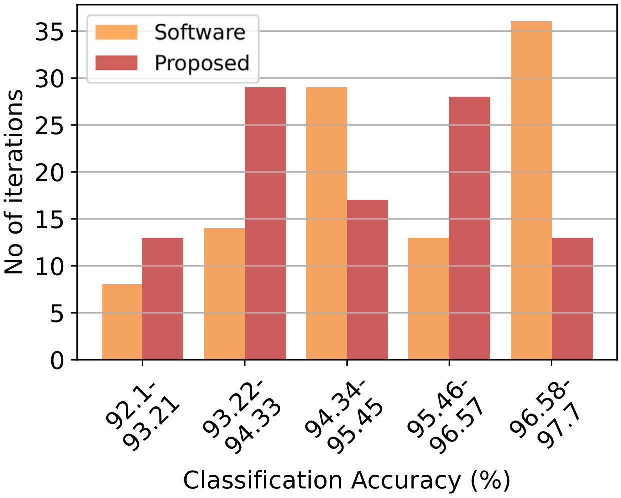

Fig. 19. Classification results of the introduced architecture and its software counterpart on the second dataset across 20 iterations.

Table 3
Monte Carlo analysis simulation results for both datasets.

Method	Best (%)	Worst (%)	Mean (%)	Std. (%)
First dataset	99.5	95.3	97.81	0.95
Second dataset	97.2	91.6	94.03	1.32

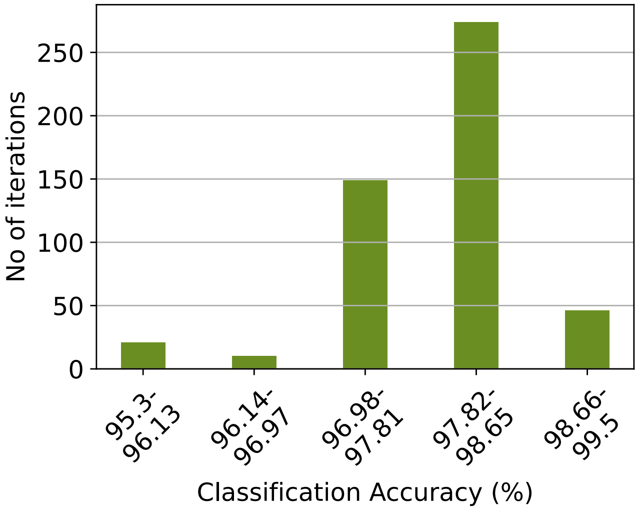

Fig. 20. Post-layout Monte-Carlo simulation results of the proposed architecture on the first dataset.

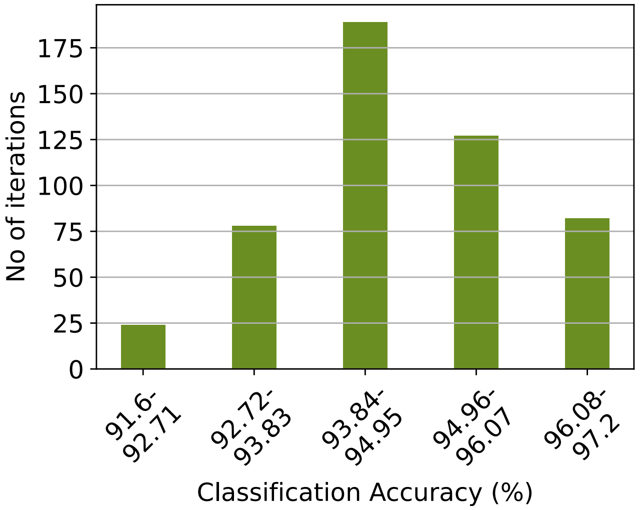

Fig. 21. Post-layout Monte-Carlo simulation results of the proposed architecture on the second dataset.

thereby confirming the acceptable sensitivity characteristics of the proposed classifiers.

Except from the Monte-Carlo analysis, the proposed classifiers is tested over PVT variations. The selected corners encompass TT, SS, FF, SF, FS (where T stands for Typical, S for Slow, and F for Fast). Additionally, the power supply rails fluctuate within the range of $V_{DD} = -V_{SS} = 0.25$ V to $V_{DD} = -V_{SS} = 0.35$ V. Regarding temperature, the assessed spectrum ranges from −25 °C to 125 °C. The proposed architecture exhibits resilience across corners, maintaining a minimum classification accuracy of 90.8%, under the worst-case scenario for the difficult dataset among the two. The most challenging corner scenario emerges with SS, −25 °C, $V_{DD} = -V_{SS} = 0.25$ V, coupled with reduced software-based accuracy (worst case). Also, a combination between the worst case corner and Monte Carlo analysis (both process and mismatch) is provided in order to test the sensitivity behavior of the classifier (see Fig. 22). This combination provides PVT simulation models related to systematic variations and Monte Carlo simulation models regarding random variations. The proposed architecture demonstrates robustness, maintaining a worst-case accuracy above 84.6% for both datasets. Simultaneously, the calculated variance in both cases remains below the 1% threshold, thereby confirming the acceptable sensitivity characteristics of the proposed classifiers.

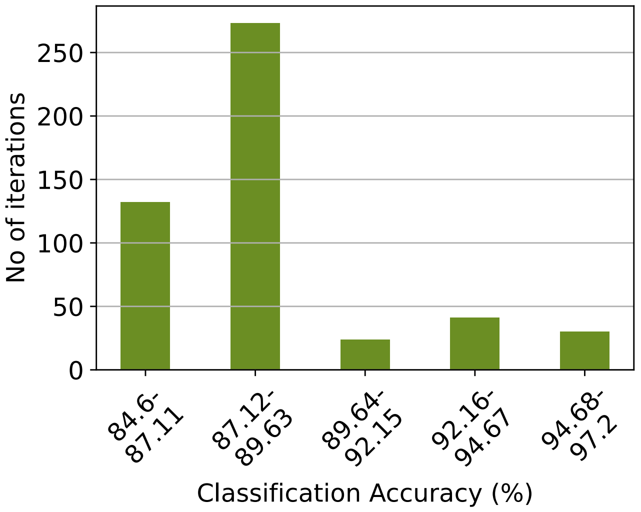

Fig. 22. Post-layout Monte-Carlo simulation results of the proposed architecture on the second dataset for the worst case corner.

Based on the corner analysis and Monte-Carlo simulation results, we can model the related effects. These variations will be added to the modeling errors. After a carefully examination, the more correct way of doing the training is "chip in the loop" training, where the weights will be adjusted individually for each chip. This training procedure needs further investigation.

7. Comparison study and discussion

The prevailing literature indicates that analog classifiers are commonly crafted as specialized engines for specific applications. This specialization poses a challenge when endeavoring to fairly compare different implementations. Thus, there exists an opportunity to customize the design of these classifiers to suit the same application, enabling a thorough evaluation of performance across various machine learning models and methodologies. Specifically, Table 4 provides a performance overview of this research alongside analog integrated classifiers, all tailored for the primary tumor classification task. All the outlined classifiers are implemented using TSMC's 90 nm CMOS process technology, with power supply rails chosen based on the operational region and a balance between heightened accuracy and reduced power consumption.

All classifiers underwent training utilizing essential software, relying on the mathematical models delineated in each implementation. Subsequently, they were all implemented and simulated using the TSMC 90 nm CMOS process. For comparison purpose, all underwent schematic-level verification, with the exception of our work, which also underwent layout-level verification. Necessary enhancements were then incorporated to optimize both classification accuracy and speed, with a primary focus on minimizing power consumption. We adhered to the identical design process as elucidated in the design procedure subsection. In instances where the architecture operates in saturation, we applied specific techniques tailored to that operational region. The aforementioned process aimed to ensure equitable comparison, given the disparate technologies and distinct classification tasks utilized in

Table 4
Analog classifiers' comparison on the primary tumor dataset.

	Classifier	Min accuracy (%)	Mean accuracy (%)	Max accuracy (%)	Power consumption (μW)	Processing speed $\left(\frac{\text{classifications}}{\text{s}}\right)$	Energy (pJ) per classification	Architecture complexity
This work	ANN	92.1	94.9	97.2	0.976	$500K$	1.95	Medium
[18]	Manhattan	89.2	91.3	94.1	0.873	$220K$	3.97	Medium
[19]	Fuzzy	91.4	93.8	96.5	1.055	$4.55K$	231.87	Medium
[20]	GMM	89.3	92.3	95.1	1.93	$120K$	16.08	Medium
[21]	RBF	86.4	90.1	92.1	20.34	$170K$	119.65	Medium
[22]	RBF-NN	92.0	94.6	97.1	1.43W	$270K$	5.29	Medium
[26]	Bayes	87.3	91.5	94.2	0.921	$120K$	7.68	Low
[27]	SVM	90.7	92.7	94.5	930.2	$870K$	1069	High
[28]	SVM	91.1	93.7	96.3	58.93	$140K$	420.93	Medium
[29]	K-means	94.2	96.2	98.6	295.45	$5M$	59.09	High
[30]	SVR	95.1	96.1	97.8	96.9	$870K$	111.37	High
[31]	SVDD	95.5	96.4	96.8	71.34	$530K$	134.61	High
[32]	SOM	95.3	96.8	99.3	812.51	$180K$	4513	Medium
[33]	LSTM	98.1	99.3	100.0	59 000	$870M$	67.81	Very high
[34]	MLP	96.3	97.4	99.1	1035	$930K$	1112	High
[35]	Threshold	91.5	92.4	94.7	0.528	$100K$	5.28	Low
[36]	Centroid	91.3	94.6	97.2	4.05	$170K$	23.82	Medium
[23]	RBF NN	91.7	94.5	96.8	8.74	$250K$	34.96	Medium
[24]	RBF NN	90.8	93.9	94.7	10.43	$310K$	33.65	Medium
[25]	ANN	88.7	91.5	94.6	26.31	$3M$	8.77	Medium
[37]	SNN	96.7	97.4	98.1	31.51	$350k$	90.03	High
[38]	SNN	95.7	96.1	97.4	1.055	$410k$	3.1	Very high
[39]	PM	92.8	94.2	95.7	93.78	$180k$	521.0	Medium

the implementations. The related comparison includes a variety of analog classifiers which are referred in Introduction.

The configurations detailed in Table 4 rely on approximations of mathematical models. Moreover, the implementations cited in [19–21,26–28,32,35,36] incorporate Gaussian function (Bump) circuits as their foundational structural elements. In these architectures, the power supply rails are established at $V_{DD} = -V_{SS} = 0.3$ V. For the remaining implementations, power supply rails were selected ranging between $V_{DD} = -V_{SS} = 0.6$ V and $V_{DD} = -V_{SS} = 0.75$ V. These architectures operate within the saturation region, necessitating a higher supply voltage. The core design principle of these endeavors revolves around the utilization of multivariate Gaussian functions, leading to the incorporation of cascaded circuits. At the circuit level, the bias current of each Bump circuit serves as the output current of the preceding one. The primary limitation arises from the attenuation of current from the input to the output of the multivariate Gaussian function circuit. In contrast to alternative studies, this work distinguishes itself by offering the capability to control weights for each individual feature, rather than adjusting the overall probability for the entire class. Additionally, existing methodologies exhibit a limited operating range for classifiers. If the chosen parameter during training approaches the power supply edges, the output current decreases in comparison to a parameter situated at the center of the power supply. Consequently, the output current of the Gaussian function circuit may diminish to a level below the permissible operating current for subsequent circuits.

Regarding architectural complexity, a spectrum of approaches exists, ranging from low to high complexity, with the specific ML model and the nature of the approximation influencing the level of complexity. The proposed ANN classifier emerges as the most effective in achieving high classification accuracy and performance. This superiority is attributed to the quality of the proposed architecture's approximation compared to other approaches. The proposed implementations outperform the area efficient classifiers in Table 4 regarding mean accuracy, except from high complexity algorithms. Notably, this heightened performance is achieved with the least energy consumption per classification compared to alternative approaches. While the Threshold classifier achieves the lowest power consumption, it does so at the expense of accuracy and processing speed due to the simplicity of its model. Moreover, this work provides a trade-off between power consumption, energy per classification, and classification accuracy, emphasizing the flexibility to sacrifice speed for power consumption in biomedical applications [86].

8. Conclusion

This study introduced a design approach specifically tailored for analog integrated ANN architecture, with a focus on low-power applications while achieving high accuracy (exceeding for both datasets 95.3% and 91.6% respectively). The architecture at a high level incorporates MDCs, SFCs, analog multipliers, CMs, and operational amplifiers. A current-mode approach was established, characterized by the circuits designated for each class. The implementation demonstrated power efficiency (below 976 nW for the challenging dataset) and operated at a low supply voltage (0.6V). Moreover, they exhibited resilience to PVT variations, validated through both Corners and Monte-Carlo simulations. The proposed architecture underwent testing in biomedical classification tasks, compared with software-based implementations and other analog IC classifiers. These designs were crafted and simulated utilizing a 90 nm CMOS process within the Cadence IC Suite.

CRediT authorship contribution statement

Vassilis Alimisis: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization. **Andreas Papathanasiou:** Writing – original draft, Visualization, Investigation. **Evangelos Georgakilas:** Writing – original draft, Visualization, Validation, Software. **Nikolaos P. Eleftheriou:** Writing – original draft, Validation, Investigation. **Paul P. Sotiriadis:** Writing – review & editing, Supervision, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

A preliminary version of this work has been published in PACET 2024 [18].

References

[1] Dzobo K, Adotey S, Thomford NE, Dzobo W. Integrating artificial and human intelligence: a partnership for responsible innovation in biomedical engineering and medicine. Omics 2020;24(5):247–63.
[2] Helmy M, Smith D, Selvarajoo K. Systems biology approaches integrated with artificial intelligence for optimized metabolic engineering. Metab Eng Commun 2020;11:e00149.
[3] Athanasopoulou K, Daneva GN, Adamopoulos PG, Scorilas A. Artificial intelligence: the milestone in modern biomedical research. BioMedInformatics 2022;2(4):727–44.
[4] Diaz-Flores E, Meyer T, Giorkallos A. Evolution of artificial intelligence-powered technologies in biomedical research and healthcare. Smart Biolabs Future 2022;23–60.
[5] Manickam P, Mariappan SA, Murugesan SM, Hansda S, Kaushik A, Shinde R, Thipperudraswamy S. Artificial intelligence (AI) and internet of medical things (IoMT) assisted biomedical systems for intelligent healthcare. Biosensors 2022;12(8):562.
[6] Peréz-Sánchez H, Fassihi A, Cecilia JM, Ali HH, Cannataro M. Applications of high performance computing in bioinformatics, computational biology and computational chemistry. In: Bioinformatics and biomedical engineering: third international conference, IWBBIO 2015, granada, Spain, April 15-17, 2015. proceedings, part II 3. Springer; 2015, p. 527–41.
[7] Li J, Wang S, Rudinac S, Osseyran A. High-performance computing in healthcare: an automatic literature analysis perspective. J Big Data 2024;11(1):61.
[8] Azghadi MR, Lammie C, Eshraghian JK, Payvand M, Donati E, Linares-Barranco B, Indiveri G. Hardware implementation of deep network accelerators towards healthcare and biomedical applications. IEEE Trans Biomed Circuits Syst 2020;14(6):1138–59.
[9] Tripathi JN, Kumar B, Junjariya D. Hardware accelerator design for healthcare applications: Review and perspectives. In: 2022 IEEE international symposium on circuits and systems. ISCAS, IEEE; 2022, p. 1367–71.
[10] Vijayan V, Connolly JP, Condell J, McKelvey N, Gardiner P. Review of wearable devices and data collection considerations for connected health. Sensors 2021;21(16):5589.
[11] MacLennan BJ. A review of analog computing. Technical report UT-CS-07-601, Department of Electrical Engineering & Computer Science, University of Tennessee; 2007, p. 19798–807, (September).
[12] Garg S, Lou J, Jain A, Guo Z, Shastri BJ, Nahmias M. Dynamic precision analog computing for neural networks. IEEE J Sel Top Quantum Electron 2022;29(2: Optical Computing):1–12.
[13] Chaturvedi DK. Soft computing. Stud Comput Intell 2008;103:509–612.
[14] Ibrahim D. An overview of soft computing. Procedia Comput Sci 2016;102:34–8.
[15] An BW, Shin JH, Kim S-Y, Kim J, Ji S, Park J, Lee Y, Jang J, Park Y-G, Cho E, et al. Smart sensor systems for wearable electronic devices. Polymers 2017;9(8):303.
[16] Pramanik PKD, Upadhyaya BK, Pal S, Pal T. Internet of things, smart sensors, and pervasive systems: Enabling connected and pervasive healthcare. In: Healthcare data analytics and management. Elsevier; 2019, p. 1–58.
[17] Greenacre M, Groenen PJ, Hastie T, d'Enza AI, Markos A, Tuzhilina E. Principal component analysis. Nat Rev Methods Primers 2022;2(1):100.
[18] Alimisis V, Eleftheriou NP, Sotiriadis PP. An analog integrated, low-power manhattan distance network with application to chronic kidney disease classification. In: 2024 panhellenic conference on electronics & telecommunications. PACET, IEEE; 2024, p. 1–4.
[19] Georgakilas E, Alimisis V, Gennis G, Aletraris C, Dimas C, Sotiriadis PP. An ultra-low power fully-programmable analog general purpose type-2 fuzzy inference system. AEU-Int J Electron Commun 2023;170:154824.
[20] Alimisis V, Gennis G, Touloupas K, Dimas C, Gourdouparis M, Sotiriadis PP. Gaussian Mixture Model classifier analog integrated low-power implementation with applications in fault management detection. Microelectron J 2022;126:105510.
[21] Peng S-Y, Hasler PE, Anderson DV. An analog programmable multidimensional radial basis function based classifier. IEEE Trans Circuits Syst I Regul Pap 2007;54(10):2148–58.
[22] Reda A, Qi L, Li Y, Wang G. A generic nano-watt power fully tunable 1-D Gaussian kernel circuit for artificial neural network. IEEE Trans Circuits Syst II Express Briefs 2020;67:3008679.
[23] Dorzhigulov A, James AP. Generalized bell-shaped membership function generation circuit for memristive neural networks. In: 2019 IEEE international symposium on circuits and systems. ISCAS, IEEE; 2019, p. 1–5.
[24] Mohamed AR, Qi L, Li Y, Wang G. A generic nano-watt power fully tunable 1-d gaussian kernel circuit for artificial neural network. IEEE Trans Circuits Syst II 2020;67(9):1529–33.
[25] T. Chandrasekaran S, Hua R, Banerjee I, Sanyal A. A fully-integrated analog machine learning classifier for breast cancer classification. Electronics 2020;9(3):515.
[26] Alimisis V, Gennis G, Dimas C, Sotiriadis PP. An analog Bayesian classifier implementation, for thyroid disease detection, based on a low-power, current-mode gaussian function circuit. In: 2021 international conference on microelectronics. ICM, IEEE; 2021, p. 153–6.
[27] Kang K, Shibata T. An on-chip-trainable Gaussian-kernel analog support vector machine. IEEE Trans Circuits Syst I Regul Pap 2009;57(7):1513–24.
[28] Alimisis V, Gennis G, Gourdouparis M, Dimas C, Sotiriadis PP. A low-power analog integrated implementation of the support vector machine algorithm with on-chip learning tested on a bearing fault application. Sensors 2023;23(8):3978.
[29] Zhang R, Shibata T. An analog on-line-learning K-means processor employing fully parallel self-converging circuitry. Analog Integr Circuits Signal Process 2013;75:267–77.
[30] Zhang R, Uetake N, Nakada T, Nakashima Y. Design of programmable analog calculation unit by implementing support vector regression for approximate computing. IEEE Micro 2018;38(6):73–82.
[31] Zhang R, Shibata T. A vlsi hardware implementation study of svdd algorithm using analog gaussian-cell array for on-chip learning. In: 2012 13th international workshop on cellular nanoscale networks and their applications. IEEE; 2012, p. 1–6.
[32] Li F, Chang C-H, Siek L. A compact current mode neuron circuit with Gaussian taper learning capability. In: 2009 IEEE international symposium on circuits and systems. IEEE; 2009, p. 2129–32.
[33] Zhao Z, Srivastava A, Peng L, Chen Q. Long short-term memory network design for analog computing. ACM J Emerg Technol Comput Syst (JETC) 2019;15(1):1–27.
[34] Lee K, Park J, Yoo H-J. A low-power, mixed-mode neural network classifier for robust scene classification. J Semicond Technol Sci 2019;19(1):129–36.
[35] Alimisis V, Gennis G, Tsouvalas E, Dimas C, Sotiriadis PP. An analog, low-power threshold classifier tested on a bank note authentication dataset. In: 2022 international conference on microelectronics. ICM, IEEE; 2022, p. 66–9.
[36] Alimisis V, Mouzakis V, Gennis G, Tsouvalas E, Sotiriadis PP. An analog nearest class with multiple centroids classifier implementation, for depth of anesthesia monitoring. In: 2022 international conference on smart systems and power management. IC2SPM, IEEE; 2022, p. 176–81.
[37] Donati E, Payvand M, Risi N, Krause R, Indiveri G. Discrimination of EMG signals using a neuromorphic implementation of a spiking neural network. IEEE Trans Biomed Circuits Syst 2019;13(5):795–803.
[38] Wang D, Chundi PK, Kim SJ, Yang M, Cerqueira JP, Kang J, Jung S, Kim S, Seok M. Always-on, sub-300-nw, event-driven spiking neural network based on spike-driven clock-generation and clock-and power-gating for an ultra-low-power intelligent device. In: 2020 IEEE Asian solid-state circuits conference. A-SSCC, IEEE; 2020, p. 1–4.
[39] Yamasaki T, Shibata T. Analog soft-pattern-matching classifier using floating-gate MOS technology. IEEE Trans Neural Netw 2003;14(5):1257–65.
[40] Suzuki K. Artificial neural networks: Architectures and applications. BoD–Books on Demand; 2013.
[41] Basheer IA, Hajmeer M. Artificial neural networks: fundamentals, computing, design, and application. J Microbiol Meth 2000;43(1):3–31.
[42] Abiodun OI, Jantan A, Omolara AE, Dada KV, Mohamed NA, Arshad H. State-of-the-art in artificial neural network applications: A survey. Heliyon 2018;4(11).
[43] Günther F, Fritsch S. Neuralnet: training of neural networks. R J 2010;2(1):30.
[44] Zou J, Han Y, So S-S. Overview of artificial neural networks. In: Artificial neural networks: Methods and applications. Springer; 2009, p. 14–22.
[45] Bishop CM. Neural networks for pattern recognition. Oxford University Press; 1995.
[46] Saravanan K, Kouzani AZ. Advancements in on-device deep neural networks. Information 2023;14(8):470.
[47] Silvano C, Ielmini D, Ferrandi F, Fiorin L, Curzel S, Benini L, Conti F, Garofalo A, Zambelli C, Calore E, et al. A survey on deep learning hardware accelerators for heterogeneous hpc platforms. 2023, arXiv preprint arXiv:2306.15552.
[48] Kumar K, Thakur GSM. Advanced applications of neural networks and artificial intelligence: A review. Int J Inf Technol Comput Sci 2012;4(6):57.
[49] Abdolrasol MG, Hussain SS, Ustun TS, Sarker MR, Hannan MA, Mohamed R, Ali JA, Mekhilef S, Milad A. Artificial neural networks based optimization techniques: A review. Electronics 2021;10(21):2689.
[50] Abiodun OI, Jantan A, Omolara AE, Dada KV, Umar AM, Linus OU, Arshad H, Kazaure AA, Gana U, Kiru MU. Comprehensive review of artificial neural network applications to pattern recognition. IEEE Access 2019;7:158820–46.
[51] Alom MZ, Taha TM, Yakopcic C, Westberg S, Sidike P, Nasrin MS, Hasan M, Van Essen BC, Awwal AA, Asari VK. A state-of-the-art survey on deep learning theory and architectures. Electronics 2019;8(3):292.
[52] Forootan MM, Larki I, Zahedi R, Ahmadi A. Machine learning and deep learning in energy systems: A review. Sustainability 2022;14(8):4832.
[53] Dally WJ, Keckler SW, Kirk DB. Evolution of the graphics processing unit (GPU). IEEE Micro 2021;41(6):42–51.
[54] Dhilleswararao P, Boppu S, Manikandan MS, Cenkeramaddi LR. Efficient hardware architectures for accelerating deep neural networks: Survey. IEEE Access 2022;10:131788–828.
[55] Jeon W, Ko G, Lee J, Lee H, Ha D, Ro WW. Deep learning with GPUs. In: Advances in computers. vol. 122, Elsevier; 2021, p. 167–215.
[56] Boutros A, Arora A, Betz V. Field-programmable gate array architecture for deep learning: Survey & future directions. 2024, arXiv preprint arXiv:2404.10076.
[57] Kim J-Y. FPGA based neural network accelerators. In: Advances in computers. vol. 122, Elsevier; 2021, p. 135–65.

[58] Ayachi R, Said Y, Ben Abdelali A. Optimizing neural networks for efficient FPGA implementation: A survey. Arch Comput Methods Eng 2021;28(7):4537–47.

[59] Beiu V. Digital integrated circuit implementations. In: Handbook of neural computation. CRC Press; 2020, p. E1–4.

[60] Jawandhiya P. Hardware design for machine learning. Int J Artif Intell Appl 2018;9(1):63–84.

[61] Kim H. Hardware and algorithm co-optimization for energy-efficient machine learning integrated circuits. 2022.

[62] Sun K, Chen J, Yan X. The future of memristors: Materials engineering and neural networks. Adv Funct Mater 2021;31(8):2006773.

[63] Aguirre F, Sebastian A, Le Gallo M, Song W, Wang T, Yang JJ, Lu W, Chang M-F, Ielmini D, Yang Y, et al. Hardware implementation of memristor-based artificial neural networks. Nat Commun 2024;15(1):1974.

[64] Liu X, Zeng Z. Memristor crossbar architectures for implementing deep neural networks. Complex Intell Syst 2022;8(2):787–802.

[65] Zhang W, Gao B, Tang J, Yao P, Yu S, Chang M-F, Yoo H-J, Qian H, Wu H. Neuro-inspired computing chips. Nat Electron 2020;3(7):371–82.

[66] Davies M, Srinivasa N, Lin T-H, Chinya G, Cao Y, Choday SH, Dimou G, Joshi P, Imam N, Jain S, et al. Loihi: A neuromorphic manycore processor with on-chip learning. Ieee Micro 2018;38(1):82–99.

[67] Du Z, Ben-Dayan Rubin DD, Chen Y, He L, Chen T, Zhang L, Wu C, Temam O. Neuromorphic accelerators: A comparison between neuroscience and machine-learning approaches. In: Proceedings of the 48th international symposium on microarchitecture. 2015, p. 494–507.

[68] Vatalaro M, Moposita T, Strangio S, Trojman L, Vladimirescu A, Lanuzza M, Crupi F. A low-voltage, low-power reconfigurable current-mode softmax circuit for analog neural networks. Electronics 2021;10(9):1004.

[69] Dlugosz R, Talaska T, Pedrycz W. Current-mode analog adaptive mechanism for ultra-low-power neural networks. IEEE Trans Circuits Syst II 2011;58(1):31–5.

[70] Ghomi A, Dolatshahi M. Design of a new CMOS low-power analogue neuron. IETE J Res 2018;64(1):67–75.

[71] Toloşi L, Lengauer T. Classification with correlated features: unreliability of feature ranking and solutions. Bioinformatics 2011;27(14):1986–94.

[72] Gilbert B. Translinear circuits: An historical overview. Analog Integr Circuits Signal Process 1996;9:95–118.

[73] Lazzaro J, Ryckebusch S, Mahowald MA, Mead CA. Winner-take-all networks of O (n) complexity. Adv Neural Inf Process Syst 1988;1.

[74] Alimisis V, Arnaoutoglou D, Serlis E, Kamperi A, Metaxas K, Kyriacou G, Sotiriadis P. A radar-based system for detection of human fall utilizing analog hardware architectures of decision tree model. IEEE Open J. Circuits Syst. 2024.

[75] Wang A, Calhoun BH, Chandrakasan AP. Sub-threshold design for ultra low-power systems, vol. 95, Springer; 2006.

[76] Eleftheriou NP, Ntasiou O, Alimisis V, Sotiriadis PP. A low-power temperature and process insensitive CMOS power management unit. In: 2024 panhellenic conference on electronics & telecommunications. PACET, IEEE; 2024, p. 1–4.

[77] Liu S-C. Analog VLSI: circuits and principles. MIT Press; 2002.

[78] Tajalli A, Leblebici Y. Extreme low-power mixed signal IC design: subthreshold source-coupled circuits. Springer Science & Business Media; 2010.

[79] Mead C. Analog VLSI and neutral systems, vol. 90. NASA STI/recon technical report A, 1989, p. 16574.

[80] Alimisis V, Eleftheriou NP, Kamperi A, Gennis G, Dimas C, Sotiriadis PP. General methodology for the design of bell-shaped analog-hardware classifiers. Electronics 2023;12(20):4211.

[81] Kapoor L, Thakur S. A survey on brain tumor detection using image processing techniques. In: 2017 7th international conference on cloud computing, data science & engineering-confluence. IEEE; 2017, p. 582–5.

[82] Pelgrom MJ, Duinmaijer AC, Welbers AP. Matching properties of MOS transistors. IEEE J Solid-State Circuits 1989;24(5):1433–9.

[83] Hock M, Hartel A, Schemmel J, Meier K. An analog dynamic memory array for neuromorphic hardware. In: 2013 European conference on circuit theory and design. ECCTD, IEEE; 2013, p. 1–4.

[84] Echocardiogram. In: UCI machine learning repository. 1989, http://dx.doi.org/10.24432/C5QW24.

[85] Zwitter M, Soklic M. Primary tumor. In: UCI machine learning repository. 1988, http://dx.doi.org/10.24432/C5WK5Q.

[86] Wu H-T. Current state of nonlinear-type time–frequency analysis and applications to high-frequency biomedical signals. Curr Opin Syst Biol 2020;23:8–21.