Microelectronics Journal 126 (2022) 105510

Contents lists available at ScienceDirect

Microelectronics Journal

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

Gaussian Mixture Model classifier analog integrated low-power implementation with applications in fault management detection

Vassilis Alimisis *, Georgios Gennis, Konstantinos Touloupas, Christos Dimas, Marios Gourdouparis, Paul P. Sotiriadis

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

ARTICLE INFO

MSC:
00-01
99-00

Keywords:
Gaussian Mixture Model
Bulk-controlled circuits
Low-power design
Bearing fault application
Analog integrated implementation

ABSTRACT

An Integrated Analog Gaussian Mixture Model classifier architecture is introduced consisting of multiple Gaussian function circuits and a Winner-Take-All circuit. It is modular and scalable to the number of classes and clusters, and, to the input dimensionality. The operating principles of the classifier are illustrated in detail and are used in a low-power, low-voltage and fully-tunable implementation targeting bearing fault management applications. The implementation was done in a 90 nm CMOS process using the Cadence IC Suite for the electrical and physical design. Post-layout simulation results were compared with a software implementation of the classifier confirming the proper operation of the design.

1. Introduction

In these days, the unprecedented availability of data and the progress in computing hardware have resulted in significant advancements in the fields of Machine Learning (ML) and Deep Learning (DL). [1] By leveraging large sets of open-access data, ML techniques provide automated decision-making, targeting a vast range of applications such as medical prognosis [2], financial predictions [3], industrial fault management [1], etc. The deployment of ML techniques in production involves data gathering and a computationally demanding process of algorithmic inference. In most cases, this process takes place in expensive hardware systems, such as data-centers.

Many of the ML application described above require real-time computation, which raises the need for impractical data transferring between the data acquisition systems and the data-centers. The solution to this problem is edge computing, with acquisition and computation systems integrated in the same device, eliminating the communication overhead [4]. This leads to a new domain of the smart industry, where Internet of Things (IoT) applications can benefit from the use of ML models [5]. An important aspect of the IoT systems is power consumption [6]; devices must perform high computation tasks autonomously by relying on batteries. This in turn results in the need for unprecedented low power dissipation and low area utilization. Therefore, in the last decades, there is a new trend in which low area and low power hardware accelerators are used for IoT and ML applications, directly connected to smart sensors or systems [7].

The research topic of hardware accelerators, involves the design of digital and/or analog circuits to perform ML operations in hardware. In the case of digital circuitry, Field-Programmable Gate Arrays (FPGAs) [8], GPUs [9] and digital ASICs [10] are the main building blocks for hardware accelerators, providing better computation speeds compared to conventional computers. FPGAs and digital ASICs, in particular, provide lower power consumption and more compact chip areas in comparison with GPUs. This reduction in power consumption is achieved without affecting the required accuracy. Though digital accelerators have gained popularity, mainly due to their ease of implementation, analog ones provide a promising alternative [11–13]. By leveraging the low-power properties of analog circuits, fueled by their capability to operate in sub-threshold [14], analog integrated classifiers could prove beneficial in comparison to their digital counterparts. Such classifiers can be used in a conceptual fully analog system level architecture for on sensor classification, as depicted in Fig. 1.

Motivated by recent works highlighting the modeling capabilities of the Gaussian Mixture Models (GMMs) [1] we propose a new analog integrated GMM-based classifier. The proposed architecture is fully electronically tunable, which allows for the realization of classifiers that can handle different classification problems. The overall design utilizes two analog building blocks, namely a Gaussian function circuit (Bump circuit) [15] and a Winner-Take-all (WTA) circuit [16]. The proposed architecture is used to design one classifier targeting damage

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

https://doi.org/10.1016/j.mejo.2022.105510
Received 13 March 2022; Received in revised form 2 June 2022; Accepted 3 July 2022
Available online 5 July 2022

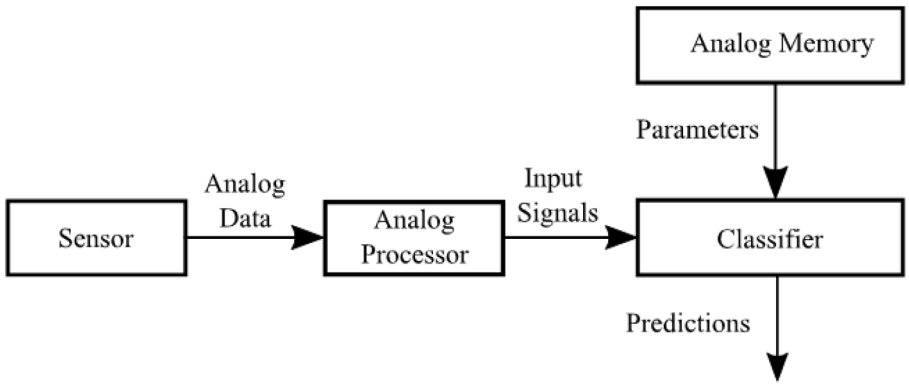

Fig. 1. Fully analog integrated implementation of a classification system.

assessment in motor bearings using two different open-source datasets. Post-layout simulation results, conducted in a TSMC 90 nm CMOS process and simulated using Cadence IC Suite, confirm the accuracy of the example implementation by comparing it with a software-based one.

The remainder of this article is organized as follows. The necessary background regarding the GMM is discussed in Section 2. Section 3 analyses the high level architecture of the proposed classifier as well as the transistor level implementations of the basic building blocks. Section 4 presents the training and tuning capabilities of the proposed architecture. The accuracy of the presented classifier is evaluated using two real-life datasets and one toy dataset in Section 5. Since there are not any analog integrated GMM implementations in the literature, works with similar scope are summarized and discussed in Section 6. Finally, Section 7 concludes the article.

2. Gaussian mixture model

Mixture Models (MM) are probabilistic models that can easily and efficiently describe complex data, making them suitable for applications in various areas of science and engineering [17]. In practice, MM-based classifiers can outperform complex models like Support Vector Machines (SVMs) and Neural Networks (NNs), that typically require excessive computational resources for training and prediction [1]. The most widespread MM is the GMM which is based on the Normal distribution and benefits from its properties [17]. GMMs constitute a highly researched topic in the literature and therefore, we consider that their implementation in analog hardware has merit.

A GMM represents the density of an N-dimensional random variable as a weighted sum of K Gaussian densities, thereby offering more expressiveness than a single Gaussian (Normal distribution) [17,18]. A GMM λ_c is uniquely defined by the number of components K, the weight factors $[w_i^c]_{i=1}^K$, the mean value vectors $[\mathbf{M}_i^c]_{i=1}^K$, $\mathbf{M}_i^c \in \mathbb{R}^N$ and the covariance matrices $[\mathbf{\Sigma}_i^c]_{i=1}^K$, $\mathbf{\Sigma}_i^c \in \mathbb{R}^{N\times N}$ of each Gaussian component. Let us consider an N-dimensional input vector $\mathbf{X} \in \mathbb{R}^N$. The probability density function (PDF) of $\mathbf{X}$, as approximated by λ_c, is given by [18]:

$$p(\mathbf{X}|\lambda_c) = \sum_{i=1}^{K} w_i^c \cdot \mathcal{N}(\mathbf{X}|\mathbf{M}_i^c, \mathbf{\Sigma}_i^c). \tag{1}$$

Here it holds that $\sum_{i=1}^K w_i^c = 1$ and $0 \le w_i^c \le 1$ for $i = 1, 2, \dots, K$. The ith N-D Gaussian component of the λ_c is denoted by $\mathcal{N}(\mathbf{X}|\mathbf{M}_i^c, \mathbf{\Sigma}_i^c)$ and its value is given by

$$\mathcal{N}(\mathbf{X}|\mathbf{M}_i^c, \mathbf{\Sigma}_i^c) = \frac{e^{-\frac{1}{2}\cdot(\mathbf{X}-\mathbf{M}_i^c)^T\cdot(\mathbf{\Sigma}_i^c)^{-1}\cdot(\mathbf{X}-\mathbf{M}_i^c)}}{\sqrt{(2\pi)^N\,|\mathbf{\Sigma}_i^c|}}, \tag{2}$$

where $|.|$ denotes the Euclidean norm. For a diagonal matrix $\mathbf{\Sigma}_i^c$, the above expression is simplified to

$$\mathcal{N}(\mathbf{X}|\mathbf{M}_i^c, \mathbf{\Sigma}_i^c) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu_n^c, (\sigma_n^c)^2), \tag{3}$$

where x_n, μ_n^c and $(\sigma_n^c)^2$ are scalars taken as the nth entry of vectors $\mathbf{X}$, $\mathbf{M}_i^c$ and the (n,n)-th entry of the matrix $\mathbf{\Sigma}_i^c$, respectively. The univariate Gaussian distribution for scalar inputs x_n is:

$$\mathcal{N}(x_n|\mu_n^c, (\sigma_n^c)^2) = \frac{1}{\sqrt{(2\pi)\cdot(\sigma_n^c)^2}}\, e^{-\frac{1}{2}\cdot\frac{(x_n-\mu_n^c)^2}{(\sigma_n^c)^2}}. \tag{4}$$

When GMMs are used in an unsupervised manner, each component captures a specific cluster of the examined dataset. This makes them suitable for clustering problems. In classification problems, such as the ones targeted by the proposed architecture, multiple GMMs are used. In this case, for each class, a single GMM is used for data clustering irrespectively to the other classes. The number of components (clusters) is chosen based on the complexity of the dataset's distribution. For an input vector $\mathbf{X}$ and C classes, the posterior probabilities $p(\lambda_c|\mathbf{X})$ are computed for each GMM $[\lambda_c]_{c=1}^C$ using the Bayes theorem:

$$p(\lambda_c|\mathbf{X}) = \frac{p(\lambda_c)p(\mathbf{X}|\lambda_c)}{p(\mathbf{X})}. \tag{5}$$

Here, $p(\lambda_c)$ is the prior and $p(\mathbf{X})$ is the evidence probability. When comparing the posterior probabilities of two classes, the evidence is ignored, since it is independent of the chosen class and serves only as a normalization constant. Therefore the overall classifier determines the winning class via

$$y = \underset{c\in[1,C]}{\operatorname{argmax}}\{p(\lambda_c)p(\mathbf{X}|\lambda_c)\}. \tag{6}$$

3. Proposed classifier's architecture

In this Section, the high level architecture of the proposed GMM-based classifier is discussed. To elucidate the reasoning behind this architecture, we assume a classification problem including N_{cla} classes and N_d inputs. The number of clusters, N_{clu}, is a hyperparameter of the overall classifier and it is chosen by exploratory data analysis of the specific application. This assumption highlights the generality of the proposed architecture, since it can be implemented accounting for various input dimensions, classes or clusters.

The structure of the proposed analog GMM-based classifier is shown in Fig. 2. Based on the classification problem formulation described previously, the classifier requires a single N_{cla}-input WTA block and N_{cla} GMM cells, each one composed of N_{clu} cluster cells. The cluster cells are in fact multidimensional Gaussian function circuits with N_d inputs. Each cell derives the probability of an input vector $\mathbf{X} = [x_1, \dots, x_{N_d}]$ belonging to a specific cluster, calculated using the Gaussian PDF of the cluster, according to (3).

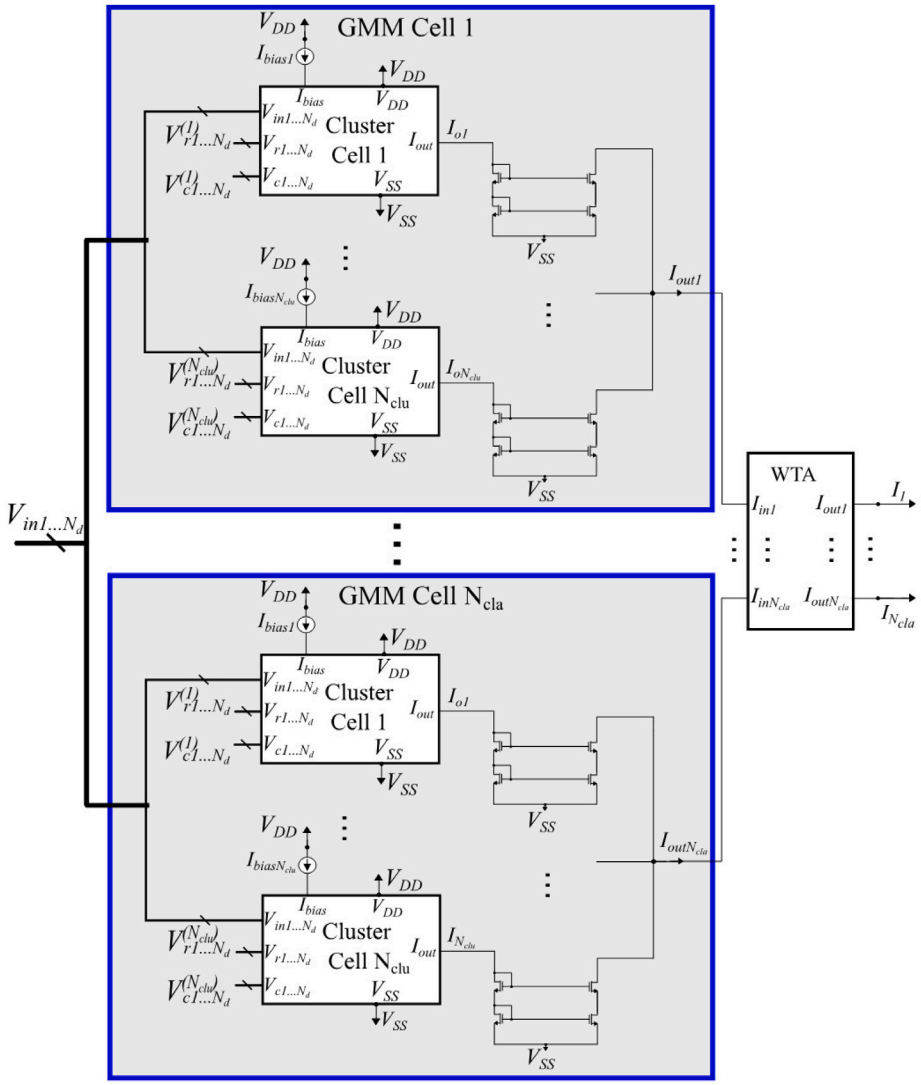

Fig. 2. Analog GMM-based classifier with N_{cla} GMM cells (classes), N_{clu} cluster cells (clusters) per class and N_d-D inputs. The WTA circuit determines the output of the classifier via the currents $[I_i]_{i=1}^{N_{cla}}$.

Based on (1), the probability of $\mathbf{X}$ belonging to a specific class is the sum of the probabilities of the clusters composing the class. This summation is performed within a GMM cell using current mirrors in order to reduce possible distortions. The WTA block implements the argmax operator and based on (6) compares the class probabilities to indicate the largest one (winning class). Additionally, by utilizing a typical WTA circuit, the winning class is indicated via a digital one-hot-vector $[I_1, \ldots, I_{N_{cla}}]$ (the currents $[I_i]_{i=1}^{N_{cla}}$ are in a binary format) [16]. Therefore, the entire classifier's output is digital.

The utilized building blocks impose a number of constraints on the maximum number of classes, clusters and input dimensions. Specifically, the number of classes is bounded by the WTA circuit's ability to accurately compare a large number of inputs. Similarly, by increasing the number of individual currents summed on a node, unwanted distortion is also increased. Therefore, the maximum number of clusters is limited by the quality of this summation. The number of input dimensions depends on the realized multidimensional Gaussian function circuit. There are multiple circuits that produce Gaussian PDFs, but in the literature they are restricted to low dimensional inputs, usually less than 5 [15].

3.1. Basic building blocks

In this subsection, the building blocks that will be used to validate the proposed classifier are thoroughly explained. Specifically, two basic analog blocks are required; a circuit generating a Gaussian PDF and a circuit for the argmax operator. Regarding the first, a typical Bump circuit [19] produces a univariate Gaussian curve and can be easily expanded for multivariate ones [20]. For the argmax, the standard Lazzaro WTA circuit [16] is used. Although these circuits provide the necessary functionalities needed for the classifier, a number of modifications were made to further increase the classifier's accuracy. Additionally, to minimize the system's power consumption all transistors operate in the sub-threshold region and the power supply rails are set to $V_{DD} = -V_{SS} = 0.3$ V for the entire classifier.

3.1.1. Modified bump circuit

Typical Bump circuits [19] output a univariate Gaussian function curve. In this work, to increase the quality of the Gaussian curve and enhance robustness, a modified Bump circuit, shown in Fig. 3, is utilized instead. In particular, instead of the non-symmetric current correlator used in [19], a symmetric one (transistors $M_{p1} - M_{p6}$ of

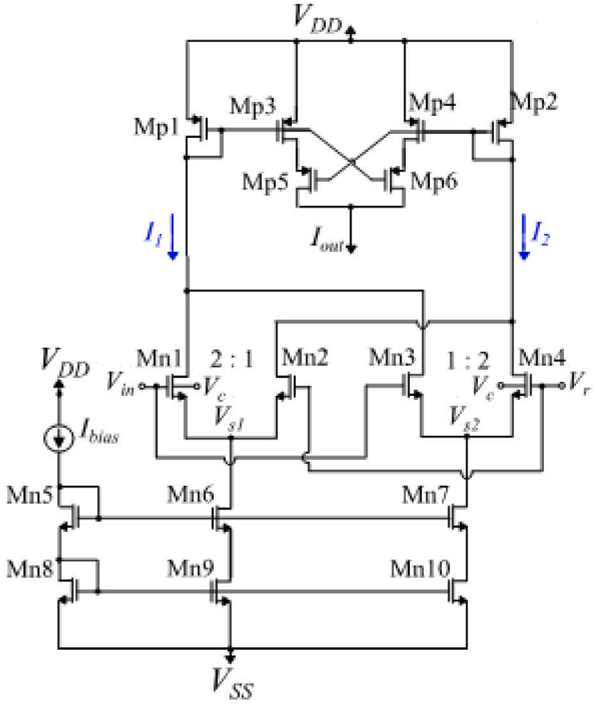

Fig. 3. Modified Gaussian Function circuit. The voltage V_{in} corresponds to the system's input. The parameter voltages V_r, V_c and the bias current I_{bias} control the mean value, the variance and the height of the Gaussian function.

Table 1
MOS Transistors' Dimensions (Fig. 3).

Block	W/L (μm/μm)	Current Correlator	W/L (μm/μm)
M_{n1}, M_{n4}	1.6/0.4	M_{p1}, M_{p2}	1.6/1.6
M_{n2}, M_{n3}	0.8/0.4	$M_{p3} - M_{p6}$	0.4/1.6
$M_{n5} - M_{n8}$	0.4/1.6	–	–
M_{n9}, M_{n10}	1.6/1.6	–	–

Fig. 3) is preferred. The motivation for this modification stems from the need for symmetric Gaussian curves, when comparing two PDFs in the case of GMMs. In practice, using a symmetric current correlator, the symmetry for inputs around the mean value is preserved even for small currents, as shown in Fig. 4. The cascode current mirror composing of transistors $M_{n5} - M_{n10}$ (Fig. 3) is used to enhance mirroring even for small bias currents. All transistors' dimensions are summarized in Table 1.

A multivariate Gaussian function curve, and hence a multivariate PDF, is produced based on (3). In practice, the connection of two or more Bump circuits in a cascaded format is equivalent to their multiplication [20]. For each Bump circuit the mean value and the variance are controlled by its voltage parameters V_r and V_c, respectively [19]. In this topology, the first Bump circuit has a bias current I_{bias}, setting the height of the Gaussian PDF, while the rest are biased with the output current of the previous Bump cell. An illustration of the Bump cascade, implementing a multivariate PDF, is shown in 5. It is worth mentioning that in a typical Gaussian function, the height is set by the variance through the normalization term α:

$$\alpha = \frac{1}{\sqrt{(2\pi)^{N_d}|\boldsymbol{\Sigma}|}}. \tag{7}$$

In our case, the Gaussian curve's height is set directly by the bias current I_{bias}.

3.1.2. Modified winner-take-all circuit

The following block to be discussed is the WTA circuit. In order to properly explain the utilized modified WTA circuit, the typical Lazzaro WTA circuit is briefly discussed. In a N_{cla} classification problem, this circuit is composed of N_{cla} neurons with a common bias current, shown in Fig. 6. Each neuron is responsible for the input and output of a single class. In particular, the output current of the neuron with the largest input current has a non-zero value, while the rest are zero. In the case where more than one input currents have similar values, this circuit operates in the linear region and more than one winners may occur. This is not desirable in most classification applications.

To address this issue, a modified WTA circuit is shown in Fig. 7. In particular, 3 WTA circuits are connected in a cascaded format, similarly to [21]. By alternating the NMOS and PMOS designs, there is no need for connecting circuitry between two consecutive WTA circuits. To highlight the benefits of this modification, Fig. 8 demonstrates the decision boundaries of a Lazzaro WTA circuit and the proposed Cascaded WTA circuit for the same 1-D dummy problem. The Cascaded WTA circuit provides much steeper linear region, in comparison with the Lazzaro WTA circuit. All transistors' dimensions for the NMOS neuron (in Fig. 6) are set to $\frac{W}{L} = \frac{0.4\ \mu m}{1.6\ \mu m}$.

4. Training and tuning capabilities

The modified Bump circuit (Section 3.1) has electronically tunable parameters. This, in turn, allows for tuning the entire classifier electronically, even after the final design of the circuit. Therefore, one can apply the same analog classifier's topology to different problems, by using an offline training procedure to determine the parameters tailored to each problem and providing the appropriate electronic references to the circuit.

4.1. Offline training

To provide with the circuit's necessary parameters, a software implementation is required. The datasets are available in digital format and their features are pre-processed to account for the circuit's operational range (in this work, $[-100, 200]$mV). Then, a software-based classifier with the same number of classes, clusters and input dimensions as the one developed in hardware is trained on these datasets. Through this classifier, the mean values, the variances and the weights of each cluster are processed to derive the voltage parameters V_r, V_c and the bias currents I_{bias} of the hardware implementation. This procedure is performed only once and the resulting parameters are exported and saved in an analog memory [22].

For each cluster cell, parameter voltages $[V_{ri}]_{i=1}^{N_d}$, where N_d is the number of the input dimensions, correspond to the entries of the modeled Gaussian PDF's mean vector. These values can be written directly to an analog memory. On the other hand, the parameter voltages $[V_{ci}]_{i=1}^{N_d}$ control the variance of each cluster through a non-linear, monotonically increasing, bounded function. To derive this function a single Bump circuit was simulated using different values over V_c and the resulting Gaussian curves were used to fit a polynomial model that maps the acquired variances to the excitation voltages V_c. Each cluster cell is biased with a current I_{bias}, which is the product of three separate parameters; the *prior* probability of each class, the weights of each cluster and the normalization term α in Eq. (7). It is worth noting that the bias currents are normalized in the $[14, 18]$ nA range, which ensures the proper operation of the circuit while maintaining a low power consumption.

4.2. Architecture tunability

The proposed classifier is capable of post-layout tuning in the number of dimensions, clusters and classes, which effectively alters the way the architecture's building blocks operate. It is underlined that this is an additional capability of the proposed classifier and it should not be confused with the Gaussian curve's tunability described previously. A particular case where this capability proves helpful is the following:

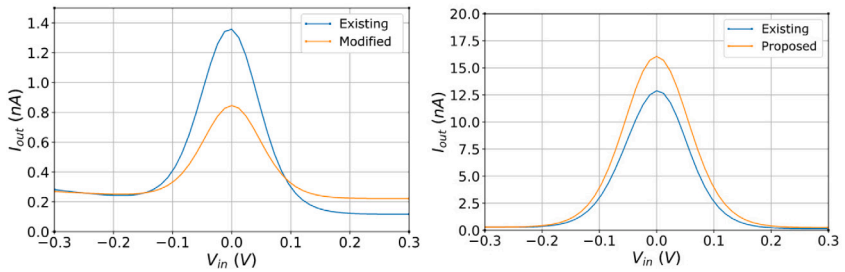

Fig. 4. Comparison between the output current of the existing Bump circuit and the modified one. The circuit's parameters are $V_r = 0$ V, $V_c = 0$ V and (left) $I_{bias} = 1$ nA (right) $I_{bias} = 16$ nA.

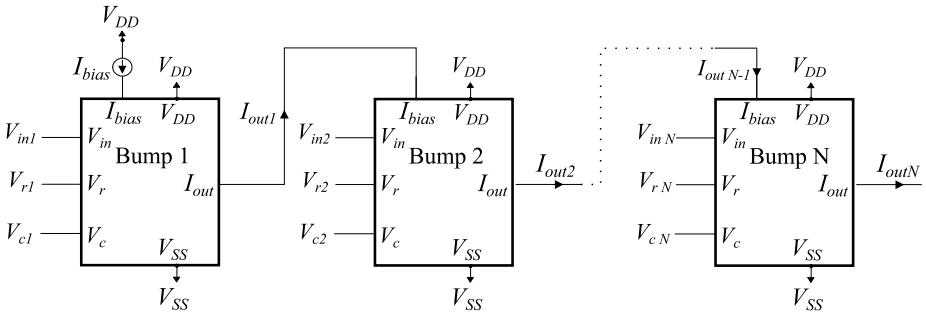

Fig. 5. A N_d-D Bump circuit implementation built by connecting N_d Bump circuits in a cascaded format. Each Bump circuit has its own voltage inputs and parameters V_{in}, V_r and V_c.

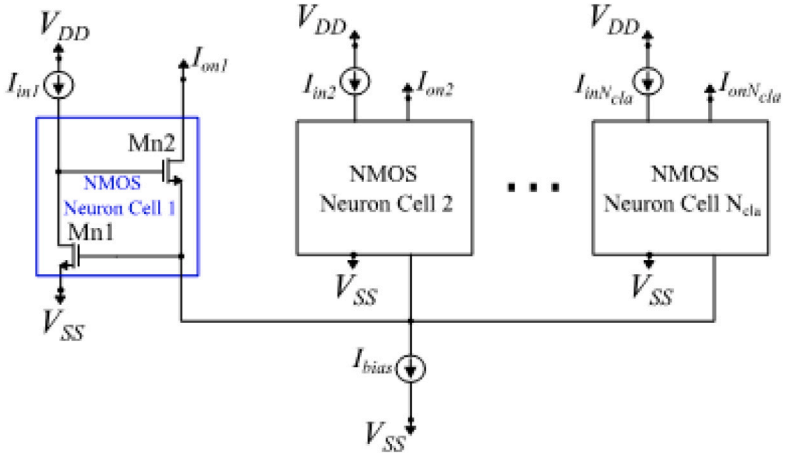

Fig. 6. Standard Lazzaro NMOS WTA with N_{cla} neurons. The complementary PMOS WTA can be built accordingly.

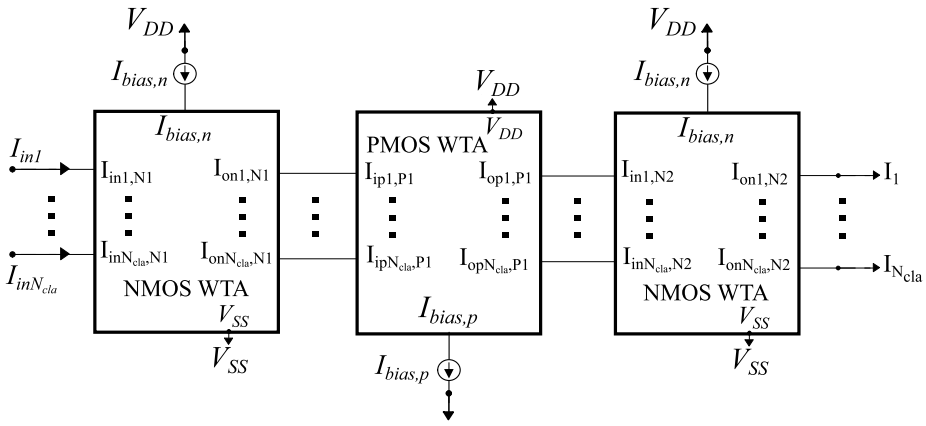

Fig. 7. The proposed Cascaded WTA circuit built by alternating the simple NMOS and PMOS WTA designs.

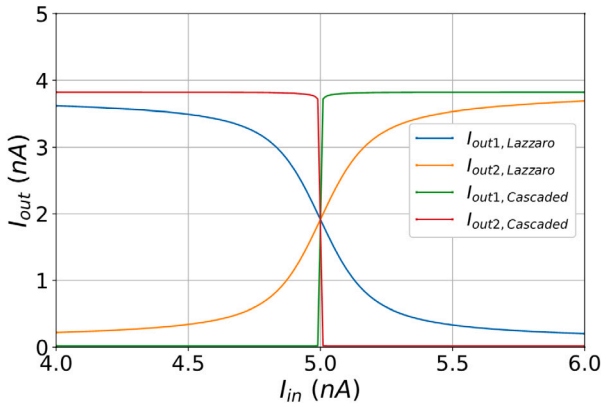

Fig. 8. Comparison between the Lazzaro WTA and the Cascaded WTA. The input current I_{in_1} is constant and equal to 5 nA, while the input current $I_{in_2} = I_{in}$ ranges linearly between 4 nA and 6 nA.

by designing a relatively large system (N_{cla} classes, N_{clu} clusters, N_d sequentially connected bump circuits), any classification problem of $m \leq N_{cla}$ classes, $k \leq N_{clu}$ clusters and $n \leq N_d$ dimensional inputs can be addressed by the same classifier's topology.

The selection of the input dimensionality is controlled via the voltage parameters V_r and V_c, as well as the input voltage V_{in}. Specifically, for the first $(N_d - n)$ out of the N_d sequentially connected Bumps, which form a cluster, all three voltages V_r, V_c, V_{in} should be set to the highest possible voltage value. By doing so these Bump circuits simply operate as current buffers for the bias current. Therefore, the system's n-D input is directed to the rest of the n Bump circuits. An important factor that needs to be considered is that this reduction in the number of dimensions does not reduce the total power consumption.

The class and cluster tunability is achieved mainly through the bias current I_{bias}. In particular, by setting the bias current of a cluster cell to zero, it remains *inactive* since its output current is typically much less than 1 nA. However, in the case of multivariate Bump circuits, the aforementioned deviation is drastically increased when passing through the chain of the univariate Bump circuits composing the multivariate one. To tackle this issue voltages V_{in} and V_r must be set to values that are far away from each other (for example to set V_r to the low supply voltage and V_{in} to the high supply voltage). Consequently, a class is inactive when all its clusters are inactive. Unlike the reduction in the input dimension, reducing the number of clusters and/or classes greatly reduces the total power consumption.

Due to the architecture's ability to deactivate clusters there are two additional design capabilities that need to be considered. First, the architecture can be set in an idle state which greatly reduces the total power consumption. This is achieved either by deactivating all of its cluster cells or, less preferably, by setting the input voltages V_{in} to the lowest possible voltage. Second, one can design a system in which cluster cells, which are implemented by different Bump circuits and produce different Gaussian curves, are activated or deactivated based on the application. This alleviates the limitations of the Gaussian curve's tunability range and effectively allows for designing multiple classifiers using the same circuit.

5. Application examples and simulation results

In this Section, to demonstrate the proposed architecture's proper operation, the classifier is tested on a custom toy and two real-life bearing fault management datasets, also used in [1]. The toy dataset has $N_{cla} = 2$ classes and $N_d = 2$-D input data, while the other two datasets have $N_{cla} = 4$ and $N_{cla} = 3$ classes and $N_d = 13$-D input data. The number of clusters for the GMM-based classifier targeting the Toy dataset is set to $N_{clu} = 2$, whereas for the other two datasets it is $N_{clu} = 4$ and $N_{clu} = 3$. For demonstration, the above dataset-related information are also provided in Table 2. By employing the techniques explained in Section 4.2, a system layout with 4 classes, 4 clusters per class and 16 sequentially connected Bump circuits can address all three previously mentioned datasets. Therefore, a single system layout, shown in Fig. 9, is designed and tuned to account for different classes, clusters and dimensionalities. The implementation of the layout is based on the common-centroid technique and extra dummy transistors are used in order to avoid mismatches and manufacturing considerations [23].

Table 2
Dataset properties.

	No. of classes	No. of clusters	No. of dimensions	No. of instances
2-D Toy	2	2	2	10000
CWRU	4	4	13	580
VSBD	3	3	13	354

Table 3
Decision Boundaries' Accuracy.

Design	Accuracy
Proposed	0.932
Baseline	0.873

To highlight the benefits of the proposed architecture, we consider a baseline GMM-based classifier, for comparison. The baseline architecture employs the typical Bump [19] and WTA [16] circuits, instead of the modified ones proposed in Section 3.1, which are utilized in the proposed one. We also compare both the proposed and the baseline classifiers with a software-based one. The proposed classifier was tested using post-layout simulation, whereas the baseline one using schematic simulations. All of the circuits and layouts discussed are developed in the Cadence IC design suite using a TSMC 90 nm CMOS process. For the software-based training and parameter extraction Python's Sklearn package [24] is utilized. The experiments were executed on a Linux workstation with 8 cores.

5.1. 2-D Toy dataset

The first classification problem considers a 2-D toy dataset. The decision boundaries of this simple problem can be illustrated in 2-D, therefore allowing for visual comparison between the results of the considered circuits. To construct this toy dataset, we consider four 2-D Gaussian distributions with means and variances equal to $\mu_1 = [-0.05, 0.06]$, $\sigma_1 = [0.02, 0.04]$, $\mu_2 = [0.15, 0.04]$, $\sigma_2 = [0.03, 0.08]$, $\mu_3 = [0.04, 0.17]$, $\sigma_3 = [0.05, 0.03]$, $\mu_4 = [0.04, -0.05]$, $\sigma_4 = [0.07, 0.02]$, where the covariance matrices are diagonal. The decision-space is $[-0.1, 0.2]^2$ and the regions where class A is the desired outcome are those where the sum of the likelihoods of the first and second previously mentioned distributions exceeds the sum of the likelihoods of the other two. The rest of the region are associated with class B. Fig. 10 (left) depicts the Ground Truth decision boundary of this problem.

By using a set of uniformly distributed points of the decision-space as inputs of the proposed classifier, its predictions were computed and saved. The same procedure holds for the baseline architecture. The resulting decision boundaries for the proposed and the baseline architectures are shown in Fig. 10 (middle) and (right), respectively. Since the decision boundaries represent the system's prediction for any given input, it is reasonable to assume that a decision boundary which resembles the Ground Truth one has a good accuracy. In this case, the proposed architecture has more limited white (ambiguous) regions and resembles the Ground Truth more accurately, compared to the baseline one. This is verified in Table 3, where the accuracies of the proposed and the baseline designs are presented.

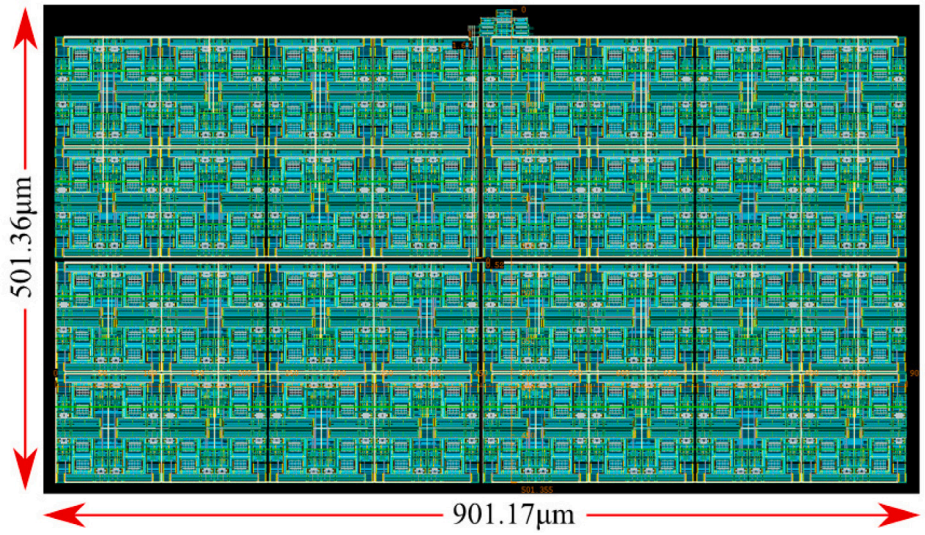

Fig. 9. Layout of a proposed GMM architecture (Proposed I) based on the design methodology (extra dummy transistors are used).

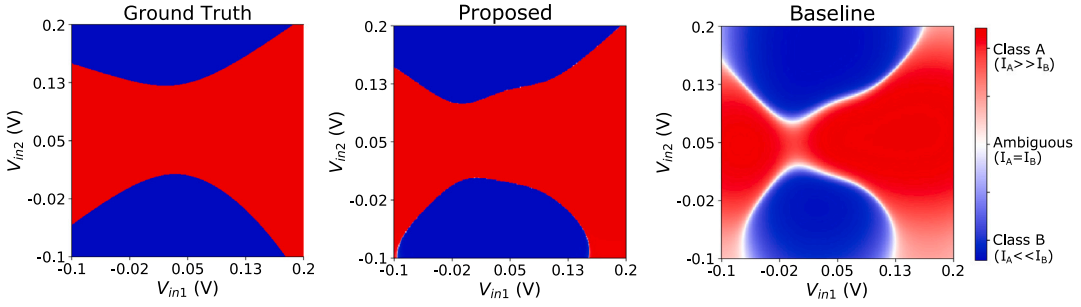

Fig. 10. 2-D decision boundaries: (left) Ground Truth (middle) the results of the proposed classifier (right) the results of the baseline classifier. The colors indicate the normalized output currents of the WTA circuit. In the white regions the value of the output current corresponding to class A is close to the value of the output current corresponding to class B. This results to low quality digital output signals for these regions. The proposed architecture has less white regions then the baseline one.

5.2. Case western reverse university dataset

The Case Western Reverse University (CWRU) dataset [25] is a bearing fault management dataset that contains accelerometer data of motors that are either operating correctly or are damaged on the inner raceway, the rolling element or the outer raceway of the bearing. The data also contain information regarding the load of the motors. In this work, we use this dataset to classify motors into four different operating conditions, namely operating correctly, faulty with inner raceway defect, faulty with outer raceway defect and faulty with rolling element defect. This classification takes place irrespectively of the motor's load conditions.

In each class, only the drive-end accelerometer data are used. These include 20-seconds time series entries, each sampled at $12 \cdot 10^3$ samples per second, which are split into 20 segments of equal duration (1 second each). Each segment is processed to produce the 13 features shown in Table 4. This results in 580 13-D input data forming the training and testing sets by using a $70\% - 30\%$ train-test split. Both training and test sets are balanced, since the four classes have 98, 112, 112, 84 training and 42, 48, 48, 36 test vectors respectively.

To highlight the gains of the proposed GMM-based classifier, two separate tests are conducted. The first one, compares, in terms of classification accuracy, the proposed, the baseline and the software-based implementation. To account for random effects induced by the training algorithm, 20 separate software-based training iterations are conducted to extract the necessary parameters of the GMM. In each iteration, all three implementations use the same parameters to ensure a fair comparison. The second test evaluates the proposed architecture's sensitivity behavior using the Monte-Carlo analysis tool for $N = 100$ points. In this case, the GMM parameters are chosen to be one of the 20 candidates of the previous test.

Table 4
Extracted features [1].

Statistic	Equation	Statistic	Equation
Root mean square	$\text{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$	Crest factor	$\text{CF} = \frac{\max(x_i)}{\text{RMS}}$
Square root of amplitude	$\text{SRA} = \left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{\lvert x_i\rvert}\right)^2$	Impulse factor	$\text{IF} = \frac{N\cdot\max(x_i)}{\sum_{i=1}^{N}\lvert x_i\rvert}$
Kurtosis value	$\text{KV} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\mu_x}{\sigma_x}\right)^4$	Margin factor	$\text{MF} = \frac{\max(x_i)}{\text{SRA}}$
Skewness value	$\text{SV} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\mu_x}{\sigma_x}\right)^3$	Frequency center	$\text{FC} = \frac{1}{N}\sum_{i=1}^{N} f_i$
Peak-to-peak value	$\text{PPV} = \max(x_i) - \min(x_i)$	Root-mean-square frequency	$\text{RMSF} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} f_i^2}$
Shape factor	$\text{SF} = \frac{\max(x_i)}{\text{SV}}$	Root variance frequency	$\text{RVF} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i - FC)^2}$
Kurtosis factor	$\text{KF} = \frac{KV}{\text{RMS}^4}$	–	–

The results in terms of classification accuracy are given in Table 5 for all three discussed implementations. It is observed that the proposed classifier circuit outperforms the baseline one and its accuracy is very close to the software-based implementation. In particular, the results of the proposed architecture have 8% increased mean accuracy and have

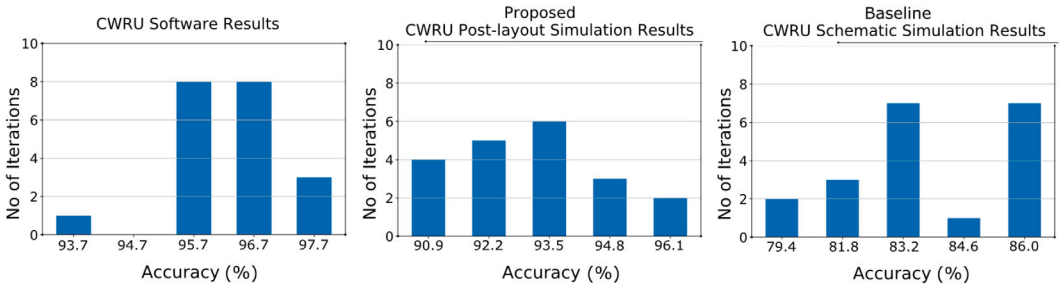

Fig. 11. Classification results, on the CWRU dataset for 20 iterations, of the (left) software implementation (middle) proposed (right) Baseline.

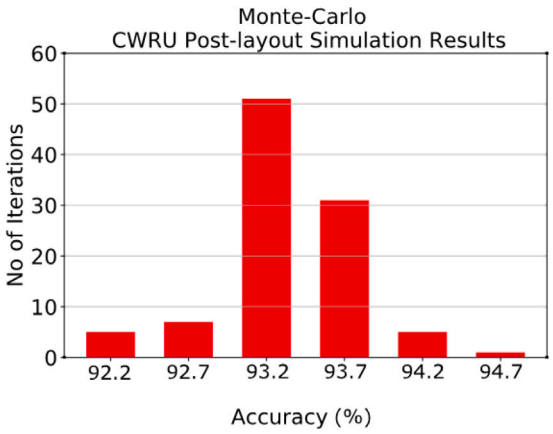

Fig. 12. Post-layout Monte-Carlo simulation results of the proposed architecture on the CWRU dataset (for one of the previous 20 iterations).

Table 5
Accuracy Results on the CWRU dataset (over 20 iterations).

Method	Best (%)	Worst (%)	Mean (%)	Std. (%)
Software	98.28	93.10	96.32	1.12
Proposed	96.55	90.23	93.08	1.64
Baseline	90.23	78.16	85.12	3.40

much less standard deviation (Std.) compared to the baseline GMM-based classifier. A more detailed comparison can be done by examining the classification accuracy histograms of Fig. 11. The Monte-Carlo analysis histogram for the proposed circuit's classification accuracy is shown in Fig. 12. The mean value is $\mu_M = 93.3\%$, and the standard deviation is $\sigma_M = 0.5\%$. This confirms the correct performance and accuracy of the proposed methodology.

5.3. Bearing vibration data under time-varying rotational speed conditions dataset

The second bearing fault management dataset considered is the Bearing Vibration Data under Time-varying Rotational Speed Conditions (VSBD [26]) dataset from the Mendeley Data [27]. It contains vibration signals from bearings that are either operating correctly or are damaged on the inner or the outer raceway. It is used to classify motors into three conditions; operating correctly, faulty with inner raceway defect and faulty with outer raceway defect.

The feature extraction and the testing process is similar to the CWRU dataset. In this case, the time series entries are 10-seconds long and sampled at $200 \cdot 10^3$ samples per second. Both training and test sets are also balanced, since all three classes have 82 training and 36 test vectors. The two tests that were used evaluate the analog classifier for the CWRU dataset, are also used here.

Table 6
Accuracy Results on the VSBD dataset (over 20 iterations).

Method	Best (%)	Worst (%)	Mean (%)	Std. (%)
Software	95.37	87.96	92.41	1.98
Proposed	92.59	80.56	87.04	3.07
Baseline	90.74	78.70	85.19	2.85

The results in terms of classification accuracy are given in Table 6 for the proposed, the baseline and the software-based GMM implementations. In this problem, the proposed classifier is consistently better than the baseline, having 2% better mean accuracy. Its performance is very close to the software base implementation. The classification accuracy histograms in this case are given in Fig. 13. It is seen that most of the baseline accuracies are below 87% whereas a significant portion of the proposed one's accuracies lie above 89%. Fig. 14 depicts the Monte-Carlo analysis results in terms of classification accuracy for the proposed circuit. The average accuracy is $\mu_M = 88.1\%$, and the standard deviation is $\sigma_M = 1.3\%$. Both conducted tests confirm the correct operation of the proposed circuit and its superiority over the baseline one.

6. Discussion

A categorization of analog integrated, mixed-mode ML architectures and the proposed GMM-based classifier is provided in this Section and summarized in Table 7. It is worth noting that the aim of this work is not a comparison between hardware ML implementations, since there are numerous aspects that need to be considered combinatorially, such as the application, power and area specifications, operation speed and so forth. Concerning the proposed architecture, the characteristics listed in Table 7 are extracted from the designed layout, which was tested on the CWRU dataset. In this case more classes and clusters are active and therefore the overall power consumption is higher compared to the other test cases examined in Section 5. Despite that, the proposed classifier's energy per classification is still the lowest in Table 7.

The majority of analog and mixed-mode classifier implementations are based on SVM or RBF models, as shown in Table 7. Crucial design parameters such as the number of dimensions or the processing speed are imposed by the application, with the processing speed of most implementations presented in Table 7 to far exceed the application's required processing speed. The applications considered in the literature, include object recognition [28,34,35], image classification [29,38], scene classification [33] and biometric signature verification [37]. In addition, there are cases where only toy datasets are used [30–32]. By examining Table 7, it is evident that mixed-mode implementations have higher power consumption compared to analog ones, but can easily implement more complex systems (for example higher number of dimensions). With the exception of [38], this work and other analog implementations usually utilize inputs with less than 16 dimensions.

The aim of this work is to introduce an architecture capable of creating both complex and adaptable or simple and efficient classifiers.

Table 7
Analog ML Algorithm summary.

	Technology	Architecture	Classifier	No. of dimensions	Power Consumption	Processing speed	Energy per classification	Area
This work	90 nm	Analog	GMM	*16	12.0 μW	$125K\frac{\text{classifications}}{\text{s}}$	$\frac{96\ \text{pJ}}{\text{classification}}$	0.451 mm^2
[28]	0.5 μm	Mixed-mode	SVM	N/A	5.9 mW	$12.8M\frac{\text{samples}}{\text{s}}$	$\frac{460\ \text{pJ}}{\text{sample}}$	9.000 mm^2
[29]	0.18 μm	Mixed-mode	SVM	64	N/A	$1M\frac{\text{vectors}}{\text{s}}$	N/A	0.125 mm^2
[30]	0.18 μm	Analog	SVM	2	220.0 μW	$870K\frac{\text{vectors}}{\text{s}}$	$\frac{252\ \text{pJ}}{\text{vector}}$	0.060 mm^2
[31]	0.5 μm	Analog	RBF NN (VQ)	2	N/A	$20K$–$40K\frac{\text{classifications}}{\text{s}}$	N/A	2.250 mm^2
[32]	0.18 μm	Analog	SVDD	2	N/A	$26.7M\frac{\text{vectors}}{\text{s}}$	N/A	N/A
[33]	0.13 μm	Mixed-mode	RBF NN	1280 × 720 pixels	2.2 mW	N/A	N/A	0.140 mm^2
[34]	0.13 μm	Mixed-mode	Object Recognition Processor	640 × 480 pixels	496.0 mW	N/A	N/A	49.000 mm^2
[35]	0.13 μm	Mixed-mode	Neuro Fuzzy Processor	N/A	57.0 mW	N/A	N/A	13.500 mm^2
[36]	0.18 μm	Mixed-mode	LSTM	16 × 16 matrix	460.3 mW	N/A	N/A	9.990 mm^2
[37]	0.5 μm	Analog	SVM	14	840.0 nW	$40\frac{\text{classifications}}{\text{s}}$	$\frac{21\ \text{nJ}}{\text{classification}}$	9.000 mm^2
[38]	0.18 μm	Analog	K-means	164	N/A	$10M\frac{\text{vectors}}{\text{s}}$	N/A	N/A

* No. of dimensions designed on the layout.

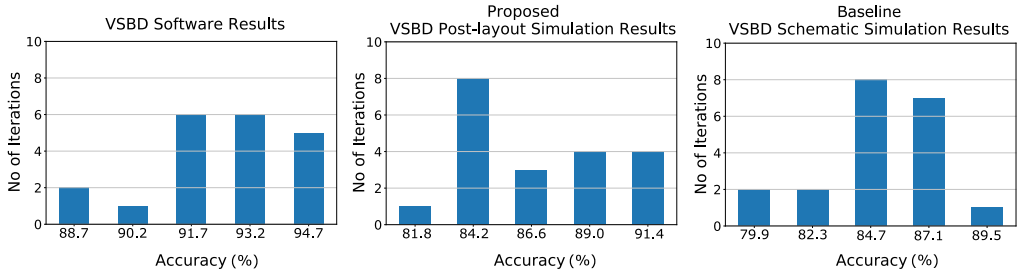

Fig. 13. Classification results, on the VSBD dataset for 20 iterations, of the (left) software implementation (middle) proposed (right) baseline.

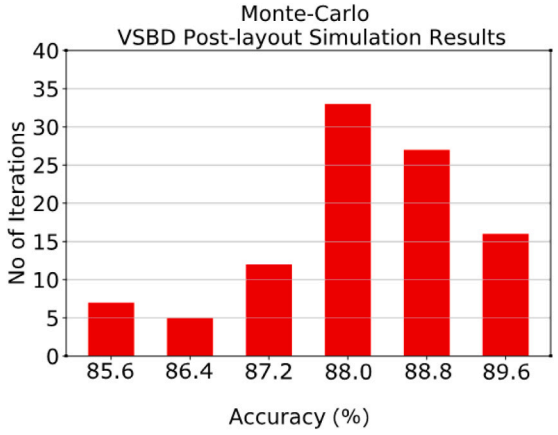

Fig. 14. Post-layout Monte-Carlo simulation results of the proposed architecture on the VSBD dataset (for one of the previous 20 iterations).

The proposed training procedure and the proposed post-layout tunability capabilities are tested through 3 different datasets achieving only 3% – 5% decrease in the classifier's accuracy, in relation to a software implementation. This is also achieved because of the utilized building blocks. The aforementioned modifications on the Bump and the WTA circuits are justified since they result in a (> 2% – 8%) increase to the classifier's accuracy, with minimal increase in the circuit's area and the power consumption. In particular, the Bump circuit of the proposed architecture consumes 3.8nW, whereas the Bump circuit of the baseline consumes 3.9nW (for 1 nA bias current). Similarly, the WTA circuit of the proposed architecture consumes 10.4nW, whereas the WTA circuit of the baseline consumes 4.5nW (for 4 nA bias current). It is worth mentioning that the scope of this work is the minimization of the power consumption while maintaining a highly accurate and area efficient classifier.

7. Conclusion

In this work, a novel architecture for tunable analog integrated GMM-based classifiers was introduced. By modifying and using Gaussian function and WTA circuits, GMM-based classifiers targeting problems with various numbers of classes, clusters and data dimensionalities can be implemented. The proposed architecture is applied on a simple toy and 2 real-world datasets targeting bearing fault diagnosis. Its parameters were generated through offline training of a GMM classifier in software. Extensive analysis and comparisons of the classification results, on these problems, highlight the proper operation of the proposed architecture and justify the applied modifications.

CRediT authorship contribution statement

Vassilis Alimisis: Conceptualization, Methodology, Writing – original draft, Software. **Georgios Gennis:** Conceptualization, Methodology, Writing – original draft, Software. **Konstantinos Touloupas:** Conceptualization, Methodology, Writing – original draft, Software. **Christos Dimas:** Writing – original draft, Supervision, Writing – reviewing and editing. **Marios Gourdouparis:** Writing – reviewing and editing. **Paul P. Sotiriadis:** Supervision, Writing – reviewing and editing.

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.

References

[1] B. Panić, J. Klemenc, M. Nagode, Gaussian mixture model based classification revisited: application to the bearing fault classification, Strojniski Vestnik/J. Mech. Eng. 66 (4) (2020).

[2] N. Bouguila, W. Fan, Mixture Models and Applications, Springer, 2020.

[3] J. Geweke, G. Amisano, Hierarchical Markov normal mixture models with applications to financial asset returns, J. Appl. Econometrics 26 (1) (2011) 1–29.

[4] M.S. Aslanpour, A.N. Toosi, C. Cicconetti, B. Javadi, P. Sbarski, D. Taibi, M. Assuncao, S.S. Gill, R. Gaire, S. Dustdar, Serverless edge computing: vision and challenges, in: 2021 Australasian Computer Science Week Multiconference, 2021, pp. 1–10.

[5] F. Hussain, R. Hussain, S.A. Hassan, E. Hossain, Machine learning in IoT security: Current solutions and future challenges, IEEE Commun. Surv. Tutor. 22 (3) (2020) 1686–1721.

[6] E. Strubell, A. Ganesh, A. McCallum, Energy and policy considerations for deep learning in NLP, 2019, arXiv preprint arXiv:1906.02243.

[7] R. Weber, A. Gothandaraman, R.J. Hinde, G.D. Peterson, Comparing hardware accelerators in scientific applications: A case study, IEEE Trans. Parallel Distrib. Syst. 22 (1) (2010) 58–68.

[8] X. Zhang, J. Wang, C. Zhu, Y. Lin, J. Xiong, W.-m. Hwu, D. Chen, Dnnbuilder: an automated tool for building high-performance dnn hardware accelerators for fpgas, in: 2018 IEEE/ACM International Conference on Computer-Aided Design, ICCAD, IEEE, 2018, pp. 1–8.

[9] A. Haidar, T. Dong, P. Luszczek, S. Tomov, J. Dongarra, Batched matrix computations on hardware accelerators based on GPUs, Int. J. High Perform. Comput. Appl. 29 (2) (2015) 193–208.

[10] C. Ding, A. Ren, G. Yuan, X. Ma, J. Li, N. Liu, B. Yuan, Y. Wang, Structured weight matrices-based hardware accelerators in deep neural networks: Fpgas and asics, in: Proceedings of the 2018 on Great Lakes Symposium on VLSI, 2018, pp. 353–358.

[11] A. Bahai, Ultra-low energy systems: Analog to information, in: ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference, IEEE, 2016, pp. 3–6.

[12] W. Haensch, T. Gokmen, R. Puri, The next generation of deep learning hardware: Analog computing, Proc. IEEE 107 (1) (2018) 108–122.

[13] K.J. Lee, J. Lee, S. Choi, H.-J. Yoo, The development of silicon for AI: Different design approaches, IEEE Trans. Circuits Syst. I. Regul. Pap. 67 (12) (2020) 4719–4732.

[14] A. Wang, B.H. Calhoun, A.P. Chandrakasan, Sub-Threshold Design for Ultra Low-Power Systems, Vol. 95, Springer, 2006.

[15] V. Alimisis, M. Gourdouparis, G. Gennis, C. Dimas, P.P. Sotiriadis, Analog Gaussian function circuit: Architectures, operating principles and applications, Electronics 10 (20) (2021) 2530.

[16] J. Lazzaro, S. Ryckebusch, M.A. Mahowald, C.A. Mead, Winner-Take-All Networks of O (N) Complexity, California Institute of Technology, 1988.

[17] C.M. Bishop, Pattern recognition, Mach. Learn. 128 (9) (2006).

[18] D.A. Reynolds, Gaussian mixture models, Encycl. Biom. 741 (2009) 659–663.

[19] M. Gourdouparis, V. Alimisis, C. Dimas, P.P. Sotiriadis, An ultra-low power, ±0.3 V supply, fully-tunable Gaussian function circuit architecture for radial-basis functions analog hardware implementation, AEU-Int. J. Electron. Commun. 136 (2021) 153755.

[20] V. Alimisis, M. Gourdouparis, C. Dimas, P.P. Sotiriadis, A 0.6 V, 3.3 nW, adjustable Gaussian circuit for tunable kernel functions, in: 2021 34th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design, SBCCI, IEEE, 2021, pp. 1–6.

[21] J. Choi, B.J. Sheu, A high-precision VLSI winner-take-all circuit for self-organizing neural networks, IEEE J. Solid-State Circuits 28 (5) (1993) 576–584.

[22] M. Hock, A. Hartel, J. Schemmel, K. Meier, An analog dynamic memory array for neuromorphic hardware, in: 2013 European Conference on Circuit Theory and Design, ECCTD, IEEE, 2013, pp. 1–4.

[23] A.K. Sharma, M. Madhusudan, S.M. Burns, P. Mukherjee, S. Yaldiz, R. Harjani, S.S. Sapatnekar, Common-centroid layouts for analog circuits: advantages and limitations, in: Proc. DATE. IEEE, Piscataway, NJ, 2021.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[25] Bearing data center, case school of engineering, 2022, https://engineering.case.edu/bearingdatacenter. (Accessed 23 May 2022).

[26] H. Huang, N. Baddour, Bearing vibration data collected under time-varying rotational speed conditions, Data Brief 21 (2018) 1745–1749.

[27] Mendeley data, mendeley, 2022, https://data.mendeley.com/. (Accessed 23 May 2022).

[28] R. Genov, G. Cauwenberghs, Kerneltron: support vector machine in silicon, IEEE Trans. Neural Netw. 14 (5) (2003) 1426–1434.

[29] R. Zhang, T. Shibata, Fully parallel self-learning analog support vector machine employing compact Gaussian generation circuits, Japan. J. Appl. Phys. 51 (4S) (2012) 04DE10.

[30] K. Kang, T. Shibata, An on-chip-trainable Gaussian-kernel analog support vector machine, IEEE Trans. Circuits Syst. I. Regul. Pap. 57 (7) (2009) 1513–1524.

[31] S.-Y. Peng, P.E. Hasler, D.V. Anderson, An analog programmable multidimensional radial basis function based classifier, IEEE Trans. Circuits Syst. I. Regul. Pap. 54 (10) (2007) 2148–2158.

[32] R. Zhang, T. Shibata, 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, pp. 1–6.

[33] K. Lee, J. Park, H.-J. Yoo, A low-power, mixed-mode neural network classifier for robust scene classification, J. Semicond. Technol. Sci. 19 (1) (2019) 129–136.

[34] J.-Y. Kim, M. Kim, S. Lee, J. Oh, K. Kim, H.-J. Yoo, A 201.4 GOPS 496 mW real-time multi-object recognition processor with bio-inspired neural perception engine, IEEE J. Solid-State Circuits 45 (1) (2009) 32–45.

[35] J. Oh, G. Kim, B.-G. Nam, H.-J. Yoo, A 57 mW 12.5 μJ/Epoch embedded mixed-mode neuro-fuzzy processor for mobile real-time object recognition, IEEE J. Solid-State Circuits 48 (11) (2013) 2894–2907.

[36] Z. Zhao, A. Srivastava, L. Peng, Q. Chen, Long short-term memory network design for analog computing, ACM J. Emerg. Technol. Comput. Syst. (JETC) 15 (1) (2019) 1–27.

[37] S. Chakrabartty, G. Cauwenberghs, Sub-microwatt analog VLSI trainable pattern classifier, IEEE J. Solid-State Circuits 42 (5) (2007) 1169–1179.

[38] R. Zhang, T. Shibata, An analog on-line-learning K-means processor employing fully parallel self-converging circuitry, Analog Integr. Circuits Signal Process. 75 (2) (2013) 267–277.