IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 July 2024, accepted 12 August 2024, date of publication 19 August 2024, date of current version 30 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3446370

RESEARCH ARTICLE

An Analog-Digital Hardware for Parzen-Based Nonparametric Probability Density Estimation

DJORDJE STANKOVIĆ[1,2], (Member, IEEE), ANDJELA DRAGANIĆ[1], (Member, IEEE), NEDJELJKO LEKIĆ[1], (Member, IEEE), CORNEL IOANA[2], (Member, IEEE), AND IRENA OROVIĆ[1,3], (Member, IEEE)

[1]Faculty of Electrical Engineering, University of Montenegro, 81400 Podgorica, Montenegro
[2]GIPSA-Laboratory, Grenoble Institute of Technology (Grenoble INP), 38402 Grenoble, France
[3]COPELABS, Universidade Lusofona, 1700-097 Lisbon, Portugal

Corresponding author: Andjela Draganić (andjelad@ucg.ac.me)

This research was conducted as part of the project titled "Advanced methods for data processing and classification using sparse coding with combined transforms and AI (deep unfolding)," which is approved by the Montenegrin Ministry of Education, Science, and Innovation.

ABSTRACT Probability estimation measures the likelihood of different outcomes in a statistical context. It commonly involves estimating either the parameters or the entire distribution of a random variable. Parametric approaches, where a specific functional form is assumed for data distribution, have been used in various fields, particularly in computational statistics for modeling and simulating physical phenomena. However, non-parametric methods have gained prominence, especially in machine learning and signal processing. These methods are focused on estimating or modeling probability density functions without relying on predefined parametric forms. It becomes crucial when faced with unknown or complex distributions, especially if parametric assumptions do not hold. This paper deals with the non-parametric method based on the Parzen window for probability density function estimation, a versatile approach applicable to univariate and multivariate data. Having a sufficient amount of data, this method provides reliable estimates, while at the same time, it is quite suitable for implementation. Considering the advantages of hardware implementations compared to software solutions, this paper introduces analog-digital hardware for the Parzen approach. The proposed solution avoids the need for sorting operations, which are typically challenging to implement in hardware. The simulation is performed using PSpice software (OrCad version 22.1) showing that the required processing time is under 420 ns.

INDEX TERMS Analog hardware, non-parametric approach, Parzen window, probability density estimation.

I. INTRODUCTION

Probability density function (PDF) estimation [1], [2], [3], [4], [5], [6], [7] plays a crucial role in various applications related to computational statistics, from modeling and simulating physical phenomena to statistical pattern recognition. The characteristics of the unknown PDF are inferred from available data samples before making any predictions or inferences. Classical approaches to PDF estimation begin with an initial screening of the data, forming a hypothesis that the data fits a particular parametric family of density curves. Subsequent steps include estimating the parameters of this density family and conducting hypothesis testing.

Generally, PDF estimation approaches are divided into parametric and non-parametric methods. Parametric approaches assume a data structure before estimating the PDF of a dataset. A typical model assumes a Gaussian distribution, allowing the construction of a relevant inference rule with two parameters characterizing the Gaussian data structure. After constructing the PDF, the training data is no longer needed, and the remaining output is the inference rule.

In the absence of prior information, non-parametric approaches for PDF estimation are used [2], [6], [7], without making functional form assumptions. In machine learning, this implies constructing a model without a predefined

The associate editor coordinating the review of this manuscript and approving it for publication was Gian Domenico Licciardo.

data structure. Non-parametric PDF estimation approaches are valuable in scenarios where the data distribution may deviate from standard parametric assumptions. These approaches, widely used in data analysis, machine learning, and signal processing, make predictions by utilizing all available data points and applying an inference rule that considers the relationship between a new data point and the entire training dataset. The simplest non-parametric approach is the histogram, employed for initial data screening, although its performance as a PDF estimate is limited. Other commonly used nonparametric methods for PDF estimation include k-Nearest Neighbor (kNN) methods, Empirical Distribution Function (EDF), and Kernel Density Estimation (KDE) [2].

Nearest neighbor methods [8], [9] estimate the value of density at the observed point based on the distance between the point and its k nearest neighbors. The empirical Distribution Function provides a step function that assigns a probability to each observed data point, forming a staircase-like estimate of the cumulative distribution function (CDF).

Kernel Density Estimation is a smoother method that places a smooth, symmetric function - kernel on each data point and sums it to form an estimate of the PDF. It provides a continuous and more refined representation of the data distribution than histograms. A special type of kernel density estimation, where a fixed window (or ''kernel'') is centered at each data point and the contributions from all windows are summed, is the Parzen window approach. The choice of the window function and its bandwidth affects the smoothness of the estimate. Neural networks [10], [11], [12], [13], [14], [15], as a computational model inspired by the structure of the human brain, commonly employ PDF estimation as a crucial step in uncertainty quantification and data modeling. For example, in [3], the Parzen window estimator and its impact on the effectiveness of Information Theoretic Learning (ITL) training, specifically using Renyi's and Shannon's entropies are considered. By comparing the convergence speed of weights, PDF estimator, and residual error, along with prediction accuracy, across various backpropagation algorithms (MSE, cross-entropy, and quadratic entropy), this study evaluates the performance of a type of neural network called the multi-layer perceptron (MLP) trained on standard datasets like MNIST.

Although software implementations have been favored for their flexibility, there is a growing need for hardware implementations due to several advantages. Namely, hardware implementations provide significantly higher processing speed and improved computational efficiency. This is particularly beneficial for real-time applications and systems with demanding performance requirements. Additionally, hardware implementations optimize resource utilization by customizing hardware architecture, leading to cost savings [16], [17], [18], [19], [20], [21]. Therefore, in this paper, we propose a simple and efficient solution for hardware implementation of the Parzen window-based density estimation approach. The proposed solution is scalable and it is not dependent on the signal size.

The paper is organized as follows. Section II briefly describes the methodology of this work, including the research question, main goals, assumptions, and evaluation steps. Section III provides theoretical aspects of the probability density estimations, while the Parzen window approach is described in Section IV. The proposed hardware architecture is described in Section V. Evaluation of the proposed hardware solution in terms of the hardware complexity, processing time, and comparison with other solutions is provided in Section VI. The results for examples with real-world data are provided in Section VII. The concluding remarks are given in Section VIII.

II. METHODOLOGY

The classification algorithms using PDF estimation of training datasets are of practical interest in many applications but require large training sets to provide reliable results. These algorithms can be used independently or as an auxiliary preprocessing step for neural network-based classifications with large datasets. Therefore, in this work, the commonly used PDF estimators are described in detail as a part of the theoretical background.

To provide real-time classification in the case of large amounts of data, it is of utmost importance to decrease the processing time and number of operations required by the PDF estimators. This is especially emphasized when dealing with various processes in the classification task.

For this purpose, we consider the Parzen window-based estimator and its real-time hardware realization. The implementation that directly follows from the algorithm steps would be very demanding regarding both the hardware complexity (number of components) and processing time. To be specific, the Parzen window-based algorithm includes a demanding sorting operation for large data sequences. For instance, one may consider the realization of the Bitonic sort algorithm ([22], [23]), being very complex and time-demanding for the observed problems, assuming large datasets.

Accordingly, the main research question is focused on the possibility of implementing a hardware solution using analog circuits that can deal with large datasets and various classification processes. Analog solutions may certainly increase the processing speed, but the sorting operation remains a challenge. Hence, the proposed approach offers a solution that allows classification without sorting operation (as presented in Section V). A slight modification of the algorithm is done using analog circuit implementation. Indeed, by processing a signal sample through a single parallel line of analog comparators and analog adders, it is possible to associate the observed sample with one of the classification processes.

Another goal, which is also important to fulfill is a large scalability of the architecture in terms of the number of classification processes, which is also addressed in this work.

The evaluation of the proposed work is performed through the following segments (discussed in Sections VI and VII):

1. Specification of the proposed hardware complexity and processing time;

2. Comparison with the analog solution that is based on the theoretical Parzen-window algorithm without optimizations; comparison with software simulation and digital implementation.

3. Experimental evaluation using real-world signals (two different types of vibrations: earthquake vibration and vehicle-produced vibration).

III. THEORETICAL BACKGROUND—PROBABILITY DENSITY ESTIMATION APPROACHES

For any random variable x, the probability density function $f(x)$ provides a comprehensive description of the distribution of x and facilitates the calculation of probabilities associated with x through the relationship:

$$P(a < x < b) = \int_a^b f(x)dx, \quad \forall a < b. \tag{1}$$

Let set $\{X_1, \ldots, X_n\}$ be a set of randomly distributed points. When $f(x)$ is unknown, where x denotes a new point, its properties need to be deduced from a set $\{X_1, \ldots, X_n\}$ before making any predictions. Then, density estimation involves creating an approximation of the PDF based on the observed data.

Suppose we have an infinite sequence of independent and identically distributed (i.i.d.) random variables $X_1, X_2, \ldots$ with an unknown distribution function f. The goal is to estimate f based on a finite random sample $X_1, \ldots, X_n$, where $n \in N$. In the context of parametric distribution function estimation, the approach involves predefining the model structure before encountering the actual data. This means having prior knowledge that the distribution fits a specific form, such as Gaussian (normal) distribution $\mathbf{N}(\mu, \sigma^2)$, where μ and σ^2 are parameters to be estimated. Hence, the goal is to accurately estimate the distribution parameters based on the observed data.

Nonparametric methods estimate the PDF based on a set of independent and identically distributed samples. The simplest non-parametric approach for PDF estimation is histogram. It divides the domain into intervals (bins) and counts the number of samples n_b falling within each interval. For example, if the variable X belongs to the interval $[a, b]$, it is divided into M non-overlapping bins, and the bin width l_{bin} is described with the following relation:

$$l_{bin} = \frac{b-a}{M}. \tag{2}$$

The bin width represents a smoothing parameter since it regulates the degree of smoothness in the histogram. Namely, smoother histograms have larger values of the bin widths while small values of l_{bin} produce histograms with more variation.

Given a random sample $X_1, \ldots, X_n$ and the extent of a bin $[x_k, x_k+1]$, the local probability density can be obtained as:

$$\hat{f}(x) = \frac{n_x}{n \times l_{bin}}, \quad \text{for} \quad x_k \le X \le x_{k+1} \tag{3}$$

where n_x denotes the number of samples in each bin, n is the total number of samples, the bin width is $l_{bin} = x_{k+1} - x_k$, and $\hat{f}$ denotes the density estimate for the probability density function f. The frequency of each bin corresponds to the number of training data points within the bin divided by the total number of training data points.

The histogram provides simple implementation and clear visualization. However, it suffers from limitations such as a lack of smoothness, with discontinuities at bin transitions [24]. Also, the computational costs become large as the number of variables increases [4]. These drawbacks have led to the development of more sophisticated methods to address these issues effectively.

Another non-parametric approach to the density estimation is the nearest neighbor estimator. It relies on the distance between data points to estimate the density at a given point. Specifically, it estimates the density at a point x by considering the k^{th} distance d_k, i.e. the distance between the point of interest and its nearest neighbor within a certain radius or neighborhood. However, the choice of the neighborhood size or radius can significantly impact the accuracy of the density estimates. The integer k is chosen to be smaller than the sample size n. If the sample size is n, then k is typically chosen to be $k \approx n^{1/2}$ [25]. A hypersphere is created using d_k as its radius and the density $\hat{f}(x)$ can be estimated as:

$$\hat{f}(x) = \frac{k}{V_D n d_k^D(x)}, \tag{4}$$

where n is the sample size, and V_D is the volume of a unit sphere in $\mathbb{R}^D$ and it is described by the following relation:

$$V_D = \frac{\sqrt{\pi^D}}{\Gamma(1 + D/2)}, \tag{5}$$

where Γ is the Gamma function. For the sphere of radius R, relation (5) becomes:

$$V_D(R) = \frac{\sqrt{\pi^D}}{\Gamma(1 + D/2)} R^D. \tag{6}$$

If the distance between two points is denoted as d, $d_1(x) \le d_2(x) \le \ldots \le d_n(x)$, then the k^{th} nearest neighbor density estimate can be calculated according to the relation [25]:

$$\hat{f}(x) = \frac{k}{2n d_k(x)}, \tag{7}$$

where $d_k(x)$ is the distance from x to the k^{th} nearest point.

One generalization of the histogram method is the naive estimator. If the random variable X has density f, then the following relation is valid:

$$f(x) = \lim_{h \to 0} \frac{1}{2h} P(x - h < X < x + h), \tag{8}$$

For a given h, it is possible to estimate $P(x-h< X< x+h)$ by determining the proportion of the sample that falls within the interval $(x-h, x+h)$. The naive estimator is obtained for small h and it is several $X_1, \ldots, X_n$ falling within the interval (x-h, $x+h$) [25]:

$$\hat{f}(x) = \frac{n_x}{2hn}. \tag{9}$$

where n_x is the number of sample points in the interval (x-h, $x+h$). From (3) and (9), it can be seen that the naive estimator is equivalent to the histogram, where the bin width is $2h$ and the center of the bin is the estimation point x. Namely, the density at a point is estimated based on the number of observations within a specified interval width h centered around that point.

The naive estimator can be written using the weight function w as follows:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} w\left(\frac{x-X_i}{h}\right),$$
$$\text{where} \quad w(x) = \begin{cases} 1/2, & \text{if} \quad |x| < 1 \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

and X_i are data samples, i.e., independent and identically distributed (i.i.d.) random variables. The relation (10) means that the estimate is obtained by placing a „box“ whose dimensions are $2h\times(1/2nh)$ on each observation, and the estimate is obtained by summing these boxes.

While this approach resolves the issue of selecting bin locations, it does not mitigate other drawbacks associated with the histogram approach. The naive estimator is not ideal for visual presentation, since $\hat{f}$ is not a continuous function and therefore, has a coarse stepwise nature. The histogram and naive estimator, although subjective in choosing the interval width, laid the foundation for the development of kernel estimators by Rosenblatt, which form a class of univariate estimators for density estimation.

Kernel density estimators benefit from flexible modeling of data distribution and adaptability to the data shape when the distribution is irregular or skewed. This assures more accurate probability estimates of classes and consequently much better classification performance. On the other hand, kernel density estimators increase complexity and computational overhead when dealing with larger datasets, so there is a need for low-complexity solutions based on optimized hardware implementations. Particularly, the Parzen window approach, as one of the most appropriate kernel density estimators, is considered in the next Section.

IV. PARZEN WINDOW APPROACH AND ITS APPLICATION IN SIGNAL CLASSIFICATION

By improving the smoothness of the weighting function, the kernel approach improves the smoothness of the naive estimator. Namely, the weighting function w in the naive estimator is replaced with a smooth symmetric kernel function K. The influence of the kernel is maximal at the point $x = X_i$, while it is lower at the points left and right from the X_i. Hence, the kernel estimator can be defined with the following relation:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right),$$
$$\int_{-\infty}^{\infty} K(t)dt = 1. \tag{11}$$

If we write kernel function as $K_h(x) = \frac{1}{h}K\left(\frac{x}{h}\right)$ then (11) can be written in the following form:

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x-X_i). \tag{12}$$

Parameter h is a smoothing parameter and it is called window width. A narrow window causes the estimate to exhibit a significant amount of noise or erroneous patterns. By using a wider window, the estimate appears smoother, but there is a risk of obscuring important features such as peaks. The estimated probability density function is derived by positioning a weighted kernel function at each data point and subsequently averaging them. The univariate kernel estimator procedure is summarized in the sequel:

- Choose a kernel function, a smoothing parameter h, and the set of x values over which to evaluate $\hat{f}(x)$;
- For each data point X_i, calculate curves according to the relation: $K_i = K\left(\frac{x-X_i}{h}\right)$, $i = 1, \ldots, n$;
- Weight each curve by $1/h$;
- For each x, take the average of the weighted curves.

The Parzen window approach is a specific type of kernel density estimation, where the window function is fixed. Instead of directly classifying data samples, the Parzen window estimates the PDF to provide insights into the distribution of the data. It involves centering a window at each observed point and calculating the weighted average of the values within the window, instead of simply counting the number of samples. With an infinite number of samples, this method converges towards the true density.

The kernel is a symmetric function that satisfies the following conditions:

$$K(x) \geq 0, \quad \forall x$$
$$K(x) = K(-x), \quad \forall x$$
$$\int K(x)dx = 1,$$
$$\int K^2(x)dx < \infty,$$
$$\int K(x)\,|x|^3\,dx < \infty.$$

Let $f(x)$ be the density function to be approximated. Having a set of n i.i.d. $X_1, \ldots, X_n$, the Parzen window estimate of $f(x)$, based on n samples can be defined as:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right). \tag{13}$$

where K is a kernel function.The commonly used kernel function is the Gaussian window function, defined as:

$$K\left(\frac{x-X_i}{h}\right)=\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-X_i}{h}\right)^2}. \tag{14}$$

Other commonly used window functions are rectangular or triangular kernels. The rectangular Parzen window function K is obtained and defined as:

$$K\left(\frac{x-X_i}{h}\right)=\begin{cases}1, & \text{for } |x-X_i|\le\frac{1}{2}, \quad i=1,\ \ldots,\ k\\ 0, & \text{otherwise,}\end{cases} \tag{15}$$

where k denotes the number of samples enclosed by the window and n is the total number of samples. The estimated (simplified) density for the rectangular window is obtained as:

$$\hat{f}(x)=\frac{k/n}{h}, \tag{16}$$

where the number of samples within the window is:

$$k=\sum_{i=1}^{n}K\left(\frac{x-X_i}{h}\right). \tag{17}$$

A. APPLICATION OF THE PARZEN WINDOW IN THE CLASSIFICATION TASK

The application of the Parzen window-based approach for density estimation in the classification task is described in the sequel. Without loss of generalization, let us consider a simplified case assuming two different processes: process P_1 accompanied by the training signal x_1 and process P_2 accompanied by the training signal x_2. The sample value y needs to be classified, i.e. assigned to P_1 or P_2. The rectangular window is used as in (15), and the number of samples inside the window for P_1 and P_2 (k_1 and k_2, respectively) is calculated according to (17).

The steps of the Algorithm are given in the sequel.

The same procedure is applied even when dealing with N processes: $P_1, \ldots, P_N$. Namely, for $k_i=\max\{k_1,\ \ldots,\ k_N\}$, it follows that $y\in P_i$.

Limitations: If the observed processes are characterized by the non-overlapping PDFs, then $|k_1-k_2|>\xi$ where ξ denotes the minimal gap assuring reliable classification. Otherwise, when $|k_1-k_2|\le\xi$ the result may be ambiguous, since it belongs to a kind of neutral zone. However, this is relevant only when observing a single sample classification. In the case of a sequence of samples, the majority of them are accurately classified.

V. A HARDWARE SOLUTION FOR PARZEN-BASED DENSITY ESTIMATION

In applications using streaming data statistics to monitor the system performance (e.g. estimation of channel or network occupancy), high-speed real-time processing is required, and consequently, the density estimation needs to be implemented using high-speed real-time hardware solutions. In the sequel, we propose an efficient hardware implementation of the Parzen window approach, where the optimized performance is achieved by exploiting the advantages of analog circuits.

Algorithm 1

1. Inputs
I) P_1: $x_1=[x_1(1),\ x_1(2),\ \ldots,\ x_1(n)]$
II) P_2: $x_2=[x_2(1),\ x_2(2),\ \ldots,\ x_2(n)]$
III) y – current sample for classification

2. Join y to x_1 and x_2 as:

$$x_{1sort}^{y}=sort\{join(x_1,y)\}$$
$$x_{2sort}^{y}=sort\{join(x_2,y)\}$$

3. Calculate the number of samples within the window:

$$k_1=count\{y-\Delta<x_{1sort}^{y}<y+\Delta\}$$
$$k_2=count\{y-\Delta<x_{2sort}^{y}<y+\Delta\}$$

where $\Delta=h/2$
4. if $k_1>k_2$ **then** $y\in P_1$
else $y\in P_2$
go to the next y

The block diagrams in Fig. 1 illustrate the hardware implementation of the Parzen window-based PDF estimation procedure. In the solution depicted in Fig. 1, the classification of input signal samples is performed using the Parzen window principle to determine association to the most probable process. Two scenarios are covered: Fig. 1(a) shows the solution when the signal is not sampled and therefore, it is fed to the Sampling circuit before the classification. The shadowed part depicted in Fig. 1(b) illustrates the modified part of the architecture from Fig. 1(a), for the case when the signal was sampled in advance, and the samples were fed to the Sample select circuit.

The proposed architecture consists of:

- **Input data part**,
- **Classification part**,
- **Data acquisition part**,
- Relaxation oscillator.

A. INPUT DATA PART OF THE PROPOSED ARCHITECTURE

The inputs of the architecture (**Input data** in Figs. 1(a) and 1(b) are:

- Samples of Process 1, …, N (Process 1, …, N blocks), and
- Input signal samples, whose probability is to be estimated (signal for sampling or data/samples for the classification, depending on what we have at the input). Blocks Process 1, …, Process N consist of sample and hold amplifiers used to store the samples. **Data/samples for the classification block** in the scenario shown in Fig. 1(b) also consist of sample and hold amplifiers.

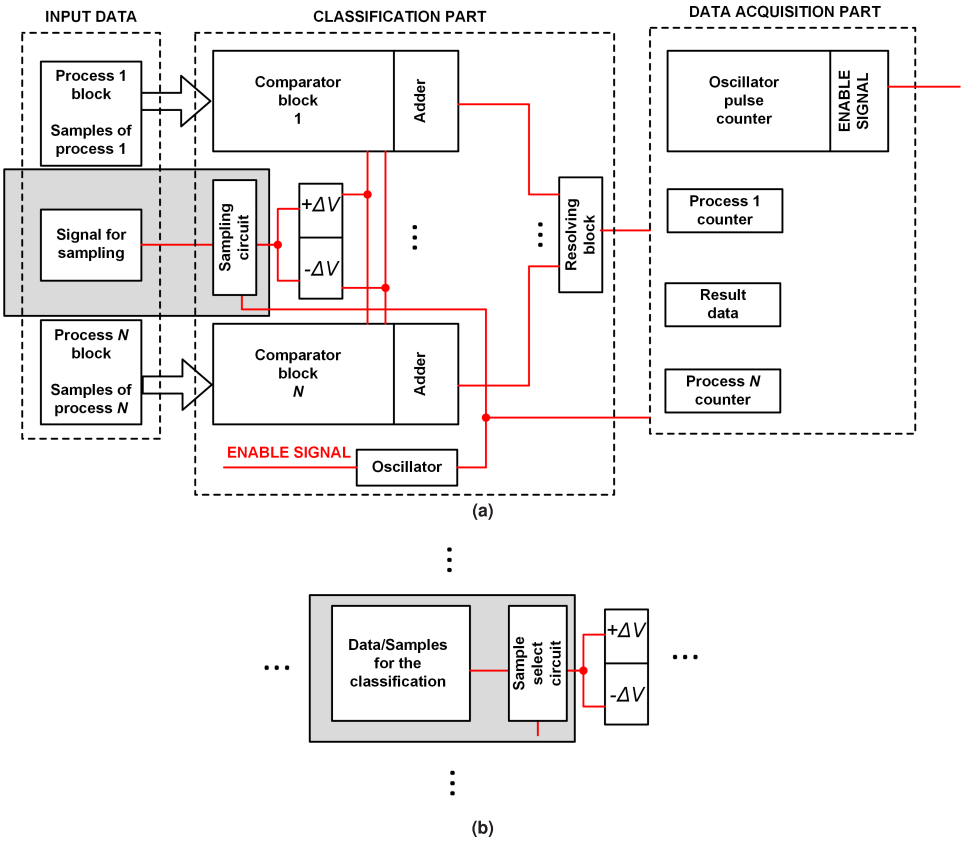


FIGURE 1. Block diagram of the proposed approach; (a) solution with signal sampling circuit in the Classification part; (b) the modified part of the architecture from (a) - here the signal is already sampled and samples are directly forwarded to the Sample select circuit.

An integral part of the architecture is the relaxation oscillator that controls the operation of the architecture. As long as the ENABLE signal of the oscillator is not activated, the architecture is inactive. Before activating the ENABLE signal, the samples are stored in Process 1 and Process N blocks, and the input signal is fed to the architecture. Activated ENABLE signal produces pulses at the oscillator's output, initiating the operation of the architecture.

B. CLASSIFICATION PART OF THE PROPOSED ARCHITECTURE

The **Classification part** consists of:

- Sampling circuit or Sample-select circuit - SSC,
- Circuits for incrementing $(+\Delta V)$ and decrementing $(-\Delta V)$,
- Comparator blocks,
- Adders,
- Resolving block.

In the case that the signal is not sampled in advance (Fig. 1(a)), the Sampling circuit is embedded within the **Classification part** of the architecture. The architecture of the Sampling circuit is shown in Fig. 2. The fundamental component of this circuit is the sample and hold (SHA) amplifier.

The sampling circuit takes a sample of the input signal and holds it at its output long enough to perform its PDF estimation. The CR part is the differentiator that defines the time SHA stays in the sampling state. Due to the CR circuit, this time can be different and shorter than the duration of the high voltage level at the output of the oscillator.

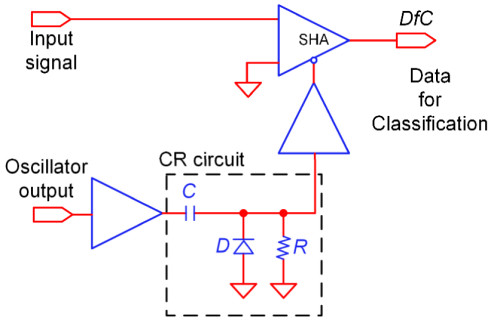


FIGURE 2. Sampling circuit.

Namely, this ensures that the SHA is in the hold state even when there are high voltage levels at the output of the oscillator, which is a fundamental requirement for the operation of the entire architecture. The selected sample at the output of the sampling circuit is denoted by *DfC* (**Data for Classification**), see Fig. 2.

In the case when the sampled signal is fed to the input (Fig. 1(b)), the Sample select circuit (SSC) is provided in the **Classification part.**The realization of the Sample select circuit is shown in Fig. 3. This circuit is used to select and connect a sample that will be classified with the rest of the architecture.

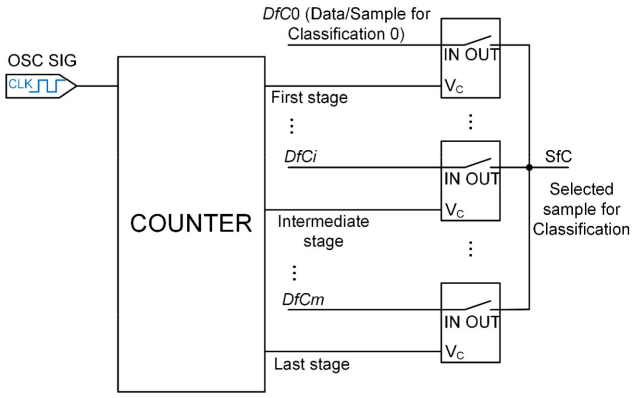


FIGURE 3. Sample select circuit.

It consists of the oscillator, decade counter 4017, logical AND circuits, and analog switches. Every ascending edge of the oscillator signal selects the next data for classification. The decade counter 4017 used in the realization has only one output at a high voltage level. The high voltage level at the output of the 4017 counter closes an analog switch, ensuring that the selected sample is moved forward. The selected sample at the output of the **Sample select circuit** is labeled as *DfC* (selected **Data for Classification**).

In analogy with the Algorithm in Section V-A, we have $\boldsymbol{y}$= *DfC*. Here, to avoid any sorting operation being known as very demanding, we propose the modification using the analog hardware implementation. Accordingly, first, it is necessary to calculate the bounds of the window:

$$\begin{aligned} y + \Delta &= DfC + \Delta V = DfC+, \text{and} \\ y - \Delta &= DfC - \Delta V = DfC - . \end{aligned} \tag{18}$$

Then for each process, the samples x_i are compared (within the **Comparator block**) with *DfC*+ and *DfC*- in parallel for $i = 1, \ldots, n$, producing the outputs according to:

$$C_{out}\,(x_i) = \begin{cases} 1, & \text{for} \quad x_i \in [DfC-, DfC+] \\ 0, & \text{otherwise.} \end{cases} , \tag{19}$$

Finally, it is necessary to count the number of "1" at the outputs of comparators using the analog adders:

$$Sum(x_i) = \sum_{i=1}^{n} C_{out}(x_i). \tag{20}$$

Therefore, according to (18), the *DfC* is firstly fed to the input of the incrementing/decrementing circuit, as shown in Figs 1(a) and 1(b) ($\pm\Delta V$ blocks). The realization of the incrementing/decrementing circuit (operations described by (18)) is shown in Fig. 4. The circuit has two outputs: *DfC*+ and *DfC*-. At the *DfC*+ output, the value of the *DfC* sample is obtained by adding ΔV: $DfC+ = DfC + \Delta V$, while at the *DfC*- output, the value is obtained by subtracting ΔV: $DfC- = DfC - \Delta V$. The results, *DfC*+ and *DfC*-, are further fed to the comparator block. The value ΔV is set empirically. In the experiments, we used the value of parameter $\Delta V = 0.1$.

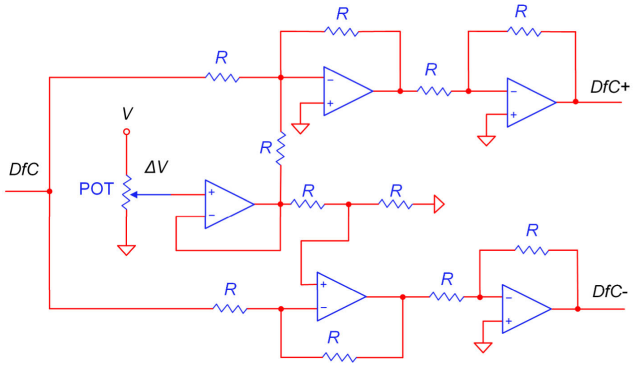


FIGURE 4. Incrementing/decrementing circuit.

The Comparator block is presented in Fig. 5, and it is the same for each process (class) x (the architecture is proposed following (19)). Let us denote the value of each sample from process x as *PxSi*. This value is compared to *DfC*+ and *DfC*-. If it is less than *DfC*+ and greater than *DfC*-, a high voltage level (e.g. 5V) will appear at the output of the NOR gate (Fig. 5). In all other cases, the voltage level at the output of the NOR gate will be low (e.g. 0V). Ultimately, the number of NOR gates with a high voltage level corresponds to the number of samples from process x whose value is within the range $DfC- < PxSi < DfC+$, or within the range of $\pm 10\%$ of the *DfC* sample value.

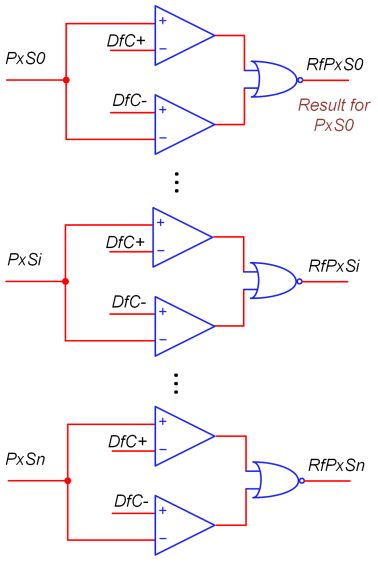


FIGURE 5. Comparator block for process x, where $x \in [1, \ldots, n]$.

The outputs of the NOR gates in the **Comparator block** are labeled as *RfPxSi*, $i \in [1, \ldots, n]$, indicating the "result for

PxSi" sample (comparator output C_{out} in (19)). These labels will be used in the further description of the architecture. The outputs of the NOR gates in the Comparator block *RfPxSi*, $i \in [1, \ldots, n]$, are further led to the inputs of analog adders (Fig. 6), according to (20).

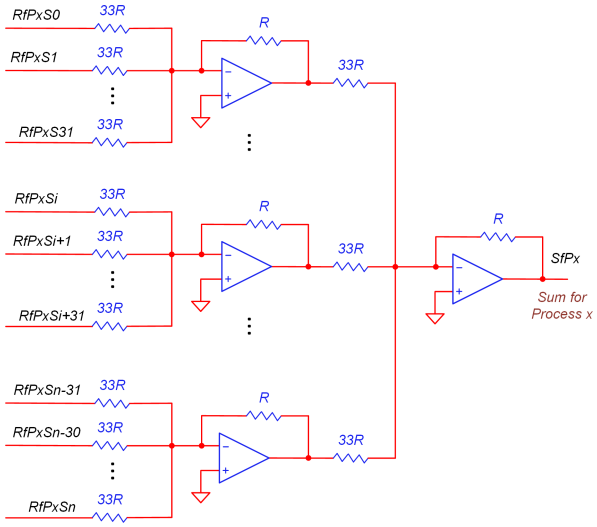


FIGURE 6. Analog adders.

The analog adders are implemented using inverting amplifiers as shown in Fig. 6. The analog voltage obtained at the adder's output (*SfPx* – Sum for process x) is directly proportional to the number of NOR gates in the Comparator block with high voltage levels at the output. For the process with the highest number of high voltage levels at the outputs of NOR gates, the adder's output will have the highest analog voltage level. The presented architecture can cover 1024 outputs of NOR gates. The architecture is scalable and can be expanded or reduced to the required number of inputs. At the output of the Comparator block, there is a decision-making block denoted by the Resolving block.

In the case of estimating the membership of a sample between two processes, the Resolving block is quite simple and consists of only one comparator, as shown in Fig. 7. The decision is straightforward: if the comparator's output voltage is high, there is a higher probability that the sample belongs to Process 1; otherwise, there is a higher probability that it belongs to Process 2.

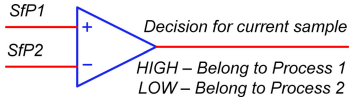


FIGURE 7. Resolving block in the case of two processes.

The complexity of the comparator block is slightly increased when the decision needs to be made among multiple processes. In Fig. 8, the resolving block is depicted in the case of deciding among 4 processes.

Let denote signals that indicate the membership of a sample to a specific process with AP_i, where $i \in \{1, \ldots, 4\}$. The outputs of the comparators K_j, $j \in \{1, 2, 3\}$ determine the membership to a specific process according to Table 1. A high voltage level at the comparator's output is represented by 1, and a low level is represented by 0. Based on the data from Table 1, the following Boolean relations between the signals AP_i, $i \in \{1, \ldots, 4\}$, and K_j, $j \in \{1, 2, 3\}$ can be easily identified:

$$\begin{aligned} AP_1 &= K_1K_3, \quad & AP_2 &= \overline{K_1}K_3, \\ AP_3 &= K_2\overline{K_3}, \quad & AP_4 &= \overline{K_2K_3}. \end{aligned} \tag{21}$$

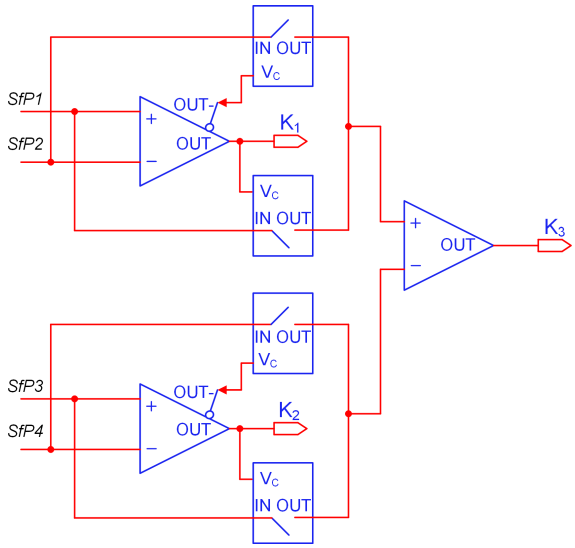


FIGURE 8. Resolving block in the case of four processes.

Fig. 9 shows the realization of the resolving block in the case of eight processes. The Boolean relations between the signals AP_i, $i \in \{1, \ldots, 8\}$, and K_j, $j \in \{1, \ldots, 7\}$ for the block in Fig. 9 are:

$$\begin{aligned} AP_1 &= K_1K_5K_7, \quad & AP_2 &= \overline{K_1}K_5K_7, \\ AP_3 &= K_2\overline{K_5}K_7, \quad & AP_4 &= \overline{K_2K_5}K_7, \\ AP_5 &= K_3K_6\overline{K_7}, \quad & AP_6 &= \overline{K_3}K_6\overline{K_7}, \\ AP_7 &= K_4\overline{K_6K_7}, \quad & AP_8 &= \overline{K_4K_6K_7}. \end{aligned} \tag{22}$$

TABLE 1. Outputs of the comparators in the case of four processes and signals AP indicating membership to a specific process.

K_1	K_2	K_3	AP
0	0	0	AP_4
0	0	1	AP_2
0	1	0	AP_3
0	1	1	AP_2
1	0	0	AP_4
1	0	1	AP_1
1	1	0	AP_3
1	1	1	AP_1

The last part of the architecture is the **Data acquisition part.** This is the digital segment of the architecture that receives data from the output of the **Classification part**, stores the data, and makes them available for further processing to make appropriate decisions. As shown in the block diagram of the proposed architecture, the **Data acquisition part** is designed to record the results of the input signal samples estimation, then count how many samples belong to individual processes, track the total number of processed samples, and finally generate a STOP signal to finish the architecture's operation. This part can be implemented in software, executed by a processor of suitable speed, or by designing an appropriate digital hardware (FPGA design or similar). In this paper, our focus has been on the analog part as the main contribution of the proposed architecture. The **Data acquisition part** involves collecting and interpreting data, which can be done in various ways using fairly standard and well-known procedures, so it is not a significant part of this work.

VI. EVALUATION OF THE PROPOSED HARDWARE SOLUTION

The simulation of the analog part of the proposed architecture is performed using PSpice software, OrCad version 22.1. It is important to emphasize that the presented evaluation of the proposed hardware solution is done using simple, standard, and widely available circuits.

The input sample blocks are composed of sample and hold amplifiers that store the values of process samples. In the simulation, AD783 sample and hold amplifiers are used with typical acquisition times of up to 0.01% 250 ns or up to 0.1% 200 ns.

In the simulation of the sample select circuit, a decade counter 4017 with a maximum transition time of 22 ns is used, along with an analog switch MAX4645 whose typical contact closure time is 12 ns and a typical contact opening time is 8 ns. For the increment-decrement circuit simulation, as well as in the analog adder simulation, the ultra-high-speed operational amplifier LT1191 is used, with a settling time (up to 0.1%) of less than 110 ns and a slew rate of 450 V/μs. The LT1394 comparator circuit is used in the simulation of the comparator and decision-making block (Resolving block), with a settling time of 7 ns.

The MC74HC1G02 logic NOR gate with a total propagation time and output transition time of less than 20 ns is used as the logical NOR gate in the simulation. In the decision-making block (Resolving block) simulation, the MAX4645 is used as an analog switch.

Next, we provide the achieved results in terms of the required processing time, taking into account each of the hardware architecture parts, as follows:

- The required time for sampling the input signal is less than 40 ns (architecture part in Fig. 2);
- Incrementing/decrementing the voltage level from the output of the selection circuit is performed in less than 270 ns (architecture part in Fig. 4).

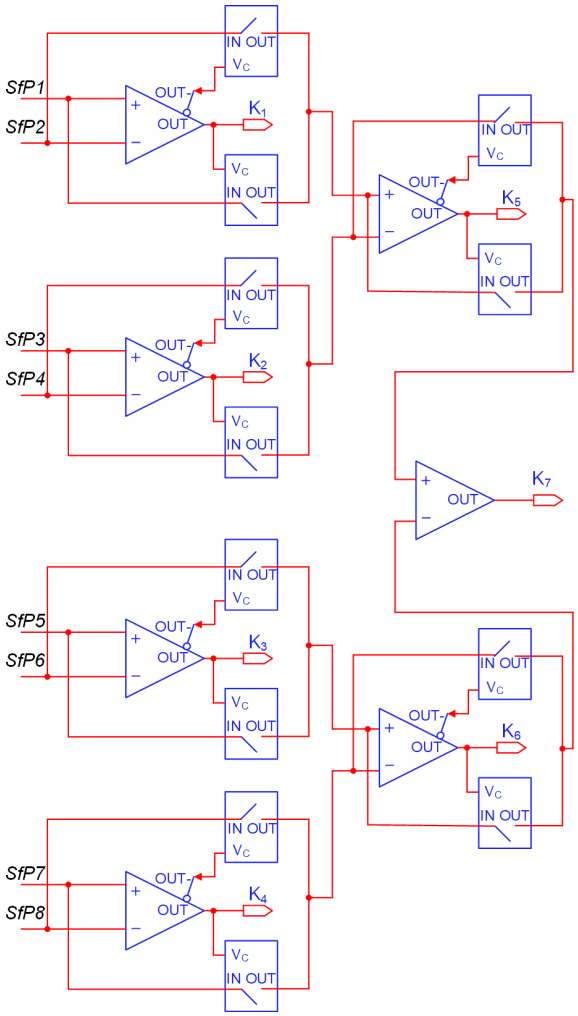


FIGURE 9. Resolving block in the case of eight processes.

- The voltage levels at the outputs of the Comparator block (Fig. 5) are established in less than 30 ns.
- The cumulative voltage level at the output of the adder (Fig. 6) is established at less than 170 ns.
- The voltage level at the output of the Resolving block (Figs. 7, 8, and 9) for 2 processes is obtained in 7 ns, for 4 processes in less than 40 ns, and for 16 processes in less than 70 ns, respectively.

By integrating these elements, the simulation results show that the required time for sampling the input signal and obtaining voltage levels at the outputs of the resolving block and comparator part of the architecture is less than 420 ns. This means that the proposed hardware can process approximately 2 million of samples per second, which further means that it can be suitable even for real-time processing of high-resolution images. It opens the possibility to use this approach for pre-classification, as an auxiliary tool for neural network applications with large-size images.

Additional improvements can be achieved using specialized and optimized circuits, as discussed later in the text.

A. COMPARISON OF RESULTS

Case 1: For comparison, let us first observe the case of analog circuit implementation that follows straightforwardly from the Parzen-window classification algorithm in Section IV. Since the algorithm includes the sorting operation, the implementation using the Bitonic sort algorithm with analog circuits is observed as the most convenient solution. PSpice simulation using analog Bitonic sort operation for 1024 samples requires approximately an additional 1000 ns, i.e. 2.5 times higher processing time compared to the proposed solution with standard circuits.

Case 2: Next, we provide the comparison with the software realization. Note that the software realization highly depends on the clock speed. The software code is executed on a CPU with a frequency of 4 GHz. The required time for 1024 samples is above 150 μs because the sorting operation requires 112 μs*.[1] Thus, the software algorithm requires approximately 300 times higher processing time.

Case 3: The proposed hardware solution can be additionally improved in terms of execution time, by using advanced and faster analog components, instead of standard ones. For instance:

- instead of sampling and hold amplifiers AD783 one can use HMC661LC4B with acquisition time reduced to 3 ns,
- instead of analog switch MAX4645, faster circuit ADG918 can be used,
- the operational amplified LT1191 can be replaced by LMH6702 to reduce a settling time to 13.4 ns,
- comparator circuit LT1394 can be replaced by ADCMP580 to reduce a settling time to 0.2 ns,
- logic NOR gate MC74HC1G02 can be replaced by 74AC02 to reduce the total propagation time to 3.5 ns.

The hardware implementation using considered high-speed circuits reduces the processing time from 420 ns to 50 ns.

When comparing this high-speed analog version with advanced digital implementation, the processing time is still more than twice as low in the case of the proposed solution. Moreover, the digital sorters in this case are pretty demanding regarding the number of gates (1.6 million gates for 1024 samples**[2]), which means that very advanced chips (e.g. Stratix 10 family) are required.

[1]Theoretically, the required time for sorting operation is: $T_{sort} = \frac{N}{2}\left(T_{comp} + T_{swap}\right)\sum_{i=1}^{\log_2 N} i$, where T_{comp} is the time required for comparison and T_{swap} the time required to swap positions. In hardware realization of a sort operation, the same relation holds without the scaling term $N/2$ because a parallel realization is considered.

[2]The total number of gates can be calculated according to $G = N(\log N)^2 b$, where N is the number of samples and b is the number of bits, e.g., $N = 1024$ and $b = 16$.

VII. EXPERIMENTAL RESULTS WITH REAL-WORLD SIGNALS

The application of Parzen-based PDF estimation in real-world applications is considered to illustrate the efficiency of the proposed approach. The goal is to choose real-world signals with distributions having certain mutual statistical overlapping. In that sense, the considered signals manifest similar behavior, and it is of practical importance to prove that the proposed method can make a distinction between the two observed processes.

Let us therefore consider two types of vibration signals: artificial vibrations caused by various vehicles and natural seismic vibrations resulting from earthquakes. Various vehicle vibration signals are used in the training process, such as ones produced during the airplane takeoff or truck transition. The samples are recorded using vibration sensors. The training processes are defined as Process 1 containing 10000 samples of different seismic signals, and Process 2 containing 10000 samples of different vehicle-caused vibrations. The classification is done for a sequence of 800 samples belonging to an unknown test seismic signal.

In Fig. 10(a), the seismic signal is depicted. The classification is done on a sample-by-sample basis. The red line in Fig. 10(b) represents correctly classified samples compared to the original signal (blue line). The results indicate that 97.9% of the samples are accurately classified as seismic, while 2.1% are incorrectly classified as artificial vibrations. Therefore, based on these numerical outcomes, we can conclude that the signal falls within the category of natural, seismic vibrations. We can also observe that this approach can be suitable for the detection of low-frequency seismographic spikes. Particularly, for the tested low-frequency seismic signals (with the frequency components ranging between 1 and 3 Hz), all samples are accurately classified (100% of success).

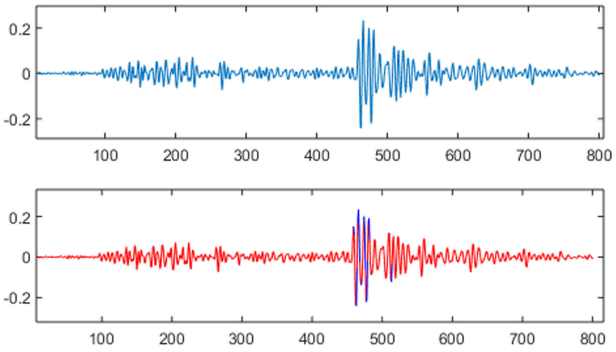


FIGURE 10. First row - Signal for testing, Second row - Original signal (blue line) and samples correctly classified as seismic (red line).

Additionally, Tables 2 and 3 show results from several experiments where various seismic and vehicle vibration signals are considered test signals. Results from Table 2 are related to the seismic signals used as test signals,

TABLE 2. Classification results when utilizing seismic signals for testing purposes.

Test signal (seismic)	% of samples classified as seismic samples (True classification)	% of samples classified as vehicle vibration samples (False classification)
Signal 1	97.88 %	2.12 %
Signal 2	98.55 %	1.45 %
Signal 3	99.34 %	0.66 %

TABLE 3. Classification results when utilizing vehicle vibration signals for testing purposes.

Test signal (vehicle vibration)	% of samples classified as vehicle vibration samples (True classification)	% of samples classified as seismic samples (False classification)
Signal 1	94.76 %	5.24 %
Signal 2	95.90 %	4.10 %
Signal 3	99.14 %	0.86 %

while Table 3 shows the outcomes in the case when vehicle vibration signals are considered as test ones.

VIII. CONCLUSION

This paper focuses on the implementation of the Parzen window approach for the estimation of the unknown probability density functions, highlighting its versatility and efficiency for this purpose. The proposed solution covers two scenarios – first when the signal has to be sampled before classification, and second, when the signal is already sampled and samples are directly forwarded to the classification part of the architecture. The proposed solution avoids the complexity of sorting operations. The theory is verified experimentally using PSpice software (OrCad version 22.1), exploiting standard circuits according to the depicted block schemes. Simulation results demonstrate the effectiveness of the proposed solution, with the processing time of 420 ns being within acceptable limits, therefore, proving the potential of the proposed solution for practical applications. Additionally, the classification accuracy is experimentally verified on several examples, considering the differentiation between natural and artificial vibrations. As natural vibrations, the seismic signals are considered while vehicle vibrations are observed as artificial ones. In all considered cases, the classification was successful with an accuracy of over 94%.

REFERENCES

[1] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 1065–1076, Sep. 1962, doi: 10.1214/aoms/1177704472.

[2] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *Ann. Math. Statist.*, vol. 27, no. 3, pp. 832–837, Sep. 1956, doi: 10.1214/aoms/1177728190.

[3] T. Ogunfunmi and M. Deb, "On the PDF estimation for information theoretic learning for neural networks," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Honolulu, HI, USA, Nov. 2018, pp. 1215–1221, doi: 10.23919/APSIPA.2018.8659642.

[4] Z. R. Yang, *Machine Learning Approaches to Bioinformatics*, 1st ed., Singapore: World Scientific, May 2010.

[5] W. L. Martinez and A. R. Martinez, *Computational Statistics Handbook With MATLAB*, 3rd ed., Boca Raton, FL, USA: CRC Press, Sep. 2021.

[6] M. Rouhani, M. Mohammadi, and A. Kargarian, "Parzen window density estimator-based probabilistic power flow with correlated uncertainties," *IEEE Trans. Sustain. Energy*, vol. 7, no. 3, pp. 1170–1181, Jul. 2016, doi: 10.1109/TSTE.2016.2530049.

[7] G. Gao, "A parzen-window-kernel-based CFAR algorithm for ship detection in SAR images," *IEEE Geosci. Remote Sens. Lett.*, vol. 8, no. 3, pp. 557–561, May 2011, doi: 10.1109/LGRS.2010.2090492.

[8] Y. Zhang and J. Wang, "GEFCom2014 probabilistic solar power forecasting based on k-nearest neighbor and kernel density estimator," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Denver, CO, USA, Jul. 2015, pp. 1–5, doi: 10.1109/PESGM.2015.7285696.

[9] C.-H. Park and J.-H. Chang, "Robust localization method based on non-parametric probability density estimation," *IEEE Access*, vol. 11, pp. 61468–61480, 2023, doi: 10.1109/ACCESS.2023.3287140.

[10] D. Mandic and J. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*. Hoboken, NJ, USA: Wiley, 2001, doi: 10.1002/047084535X.

[11] Y. Khalifa, D. Mandic, and E. Sejdić, "A review of hidden Markov models and recurrent neural networks for event detection and localization in biomedical signals," *Inf. Fusion*, vol. 69, pp. 52–72, May 2021, doi: 10.1016/j.inffus.2020.11.008.

[12] M. DelPozo-Banos, A. John, N. Petkov, D. M. Berridge, K. Southern, K. LLoyd, C. Jones, S. Spencer, and C. M. Travieso, "Using neural networks with routine health records to identify suicide risk: Feasibility study," *JMIR Mental Health*, vol. 5, no. 2, Jun. 2018, Art. no. e10144, doi: 10.2196/10144.

[13] M. Thornton, D. Mandic, and T. Reichenbach, "Robust decoding of the speech envelope from EEG recordings through deep neural networks," *J. Neural Eng.*, vol. 19, no. 4, Jul. 2022, Art. no. 046007, doi: 10.1088/1741-2552/ac7976.

[14] C. Liu, X. Ma, Y. Zhan, L. Ding, D. Tao, B. Du, W. Hu, and D. P. Mandic, "Comprehensive graph gradual pruning for sparse training in graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 27, 2023, doi: 10.1109/TNNLS.2023.3282049.

[15] D. Selimović, F. Hržić, J. Prpić-Oršić, and J. Lerga, "Estimation of sea state parameters from ship motion responses using attention-based neural networks," *Ocean Eng.*, vol. 281, Aug. 2023, Art. no. 114915, doi: 10.1016/j.oceaneng.2023.114915.

[16] N. Nedjah, R. M. Da Silva, and L. D. M. Mourelle, "Analog hardware implementations of artificial neural networks," *J. Circuits, Syst. Comput.*, vol. 20, no. 3, pp. 349–373, May 2011, doi: 10.1142/s0218126611007347.

[17] I. Orović, N. Lekić, and S. Stanković, "An analog–digital hardware for L-estimate space-varying image filtering," *Circuits, Syst., Signal Process.*, vol. 35, no. 2, pp. 409–420, Feb. 2016, doi: 10.1007/s00034-015-0083-8.

[18] S. Spanò, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Matta, A. Nannarelli, and M. Re, "An efficient hardware implementation of reinforcement learning: The Q-learning algorithm," *IEEE Access*, vol. 7, pp. 186340–186351, 2019, doi: 10.1109/ACCESS.2019.2961174.

[19] I. Orović, N. Lekić, M. Beko, and S. Stanković, "An analog hardware solution for compressive sensing reconstruction using gradient-based method," *EURASIP J. Adv. Signal Process.*, vol. 2019, no. 1, Dec. 2019, Art. no. 61, doi: 10.1186/s13634-019-0656-y.

[20] N. Lekić, M. L. Žarić, I. Orović, and S. Stanković, "Adaptive gradient-based analog hardware architecture for 2D under-sampled signals reconstruction," *Microprocessors Microsystems*, vol. 62, pp. 72–78, Oct. 2018, doi: 10.1016/j.micpro.2018.07.009.

[21] P. Dhilleswararao, S. Boppu, M. S. Manikandan, and L. R. Cenkeramaddi, "Efficient hardware architectures for accelerating deep neural networks: Survey," *IEEE Access*, vol. 10, pp. 131788–131828, 2022, doi: 10.1109/ACCESS.2022.3229767.
[22] K. E. Batcher, "Sorting networks and their applications," in *Proc. AFIPS Spring Joint Comput. Conf.*, 1968, pp. 307–314, doi: 10.1145/1468075.1468121.
[23] N. Žarić, S. Stanković, and Z. Uskoković, "Hardware realization of the robust time–frequency distributions," *Annales des Télécommunications*, vol. 69, nos. 5–6, pp. 309–320, Jun. 2014, doi: 10.1007/s12243-013-0366-7.
[24] A. R. Londhe, "Nonparametric density estimation using kernels with variable size windows," Ph.D. dissertation, Dept. Statist., Iowa State Univ., Ames, IA, USA, 1980, doi: 10.31274/rtd-180813-3561.
[25] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, 1st ed., London, U.K.: Chapman & Hall, Apr. 1986, doi: 10.1007/978-1-4899-3324-9.

DJORDJE STANKOVIĆ (Member, IEEE) was born in Podgorica, Montenegro, in September 1997. He finished high school with honors, including the prestigious "Conestoga High School," USA, and studies in Grenoble. In 2016 and 2017 academic year, he began his bachelor studies at the Faculty of Electrical Engineering, University of Montenegro, in Podgorica and received the bachelor's degree in 2019. After graduating in 2019, he pursued master's degree in mathematical engineering from the University of Buenos Aires, completing his exams and a thesis in 2022. He is currently pursuing the Ph.D. degree. Starting his doctoral studies, he excelled in exams, conducted impactful research, and submitted articles to leading journals. Fluent in English and Spanish, with programming expertise, he also commands French, Portuguese, and Italian (B2 level).

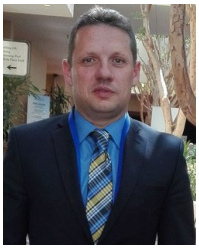
ANDJELA DRAGANIĆ (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the University of Montenegro (UoM), Podgorica, Montenegro, in 2011, 2013, and 2018, respectively.

From 2011 to 2021, she was a Teaching and Research Assistant with the University of Montenegro, where she has been an Assistant Professor with the Faculty of Electrical Engineering, since March 2021. She has published around 40 papers in international and regional journals and at international and regional conferences. Her research interests include time–frequency signal analysis, hardware architecture design, sparse signal processing, and time-frequency analysis with applications in multimedia signals and systems.

NEDJELJKO LEKIĆ (Member, IEEE) was born in Podgorica, Montenegro, in March 1968. He received the B.S. degree in electrical engineering, the M.Sc. degree in microcontroller-based design, and the Ph.D. degree from the Faculty of Electrical Engineering, University of Montenegro, Podgorica, Montenegro, in 1993, 1999, and 2006, respectively.

Since 2018, he has been a Full Professor with the University of Montenegro. His research interests include applied electronics (analog and digital), microcontroller-based design, identification systems, instrumentation and measurements, embedded systems, hardware-software co-design, and the Internet of Things.

Dr. Lekić received the Award for Innovation, in 2009, from the Chamber of Economy of Montenegro.



CORNEL IOANA (Member, IEEE) received the Dipl.-Eng. degree in electrical engineering from Romanian Military Technical Academy of Bucharest, Romania, in 1999, and the M.S. degree in telecommunication science and the Ph.D. degree in electrical engineering from the University of Brest-France, in 2001 and 2003, respectively.

Between 1999 and 2001, he acted as a Military Researcher in a research institute of the Romanian Ministry of Defense (METRA), Bucharest, Romania. Since 2006, he has been an Associate Professor-Researcher with Grenoble Institute of Technology/GIPSA-Laboratory. His scientific and research interests include nonstationary signal processing, natural process characterization, underwater systems, electronic warfare, and real-time systems.

IRENA OROVIĆ (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from the University of Montenegro (UoM), Podgorica, Montenegro, in 2005, 2006, and 2010, respectively.

She is currently a Full Professor with the Faculty of Electrical Engineering, UoM. She has published more than 130 articles, including more than 70 journal articles and co-authored several books and book chapters. Her research interests include compressive sensing, hardware architecture design, multimedia signals and systems, and time-frequency analysis.

Dr. Orović was the Vice President of the Council for Scientific Research Activity in Montenegro. She received the Award for the best Ph.D. thesis, in 2010, by the TRIMO Award Slovenia and the Award for the Best Woman Scientist in Montenegro, in 2012, by the Ministry of Science of Montenegro.