big data and cognitive computing

MDPI

Article

No-Code Edge Artificial Intelligence Frameworks Comparison Using a Multi-Sensor Predictive Maintenance Dataset

Juan M. Montes-Sánchez [1,2,*], Plácido Fernández-Cuevas [1], Francisco Luna-Perejón [1], Saturnino Vicente-Diaz [1] and Ángel Jiménez-Fernández [1,2]

[1] Robotics and Technology of Computers Laboratory, ETSII-EPS, Universidad de Sevilla, 41004 Sevilla, Spain; pfernandez9@us.es (P.F.-C.); fluna1@us.es (F.L.-P.); satur@us.es (S.V.-D.); angel@us.es (Á.J.-F.)
[2] Smart Computer Systems Research and Engineering Laboratory (SCORE), I3US, Universidad de Sevilla, 41012 Sevilla, Spain
* Correspondence: juanmanuelmontes@us.es

Abstract: Edge Computing (EC) is one of the proposed solutions to address the problems that the industry is facing when implementing Predictive Maintenance (PdM) implementations that can benefit from Edge Artificial Intelligence (Edge AI) systems. In this work, we have compared six of the most popular no-code Edge AI frameworks in the market. The comparison considers economic cost, the number of features, usability, and performance. We used a combination of the analytic hierarchy process (AHP) and the technique for order performance by similarity to the ideal solution (TOPSIS) to compare the frameworks. We consulted ten independent experts on Edge AI, four employed in industry and the other six in academia. These experts defined the importance of each criterion by deciding the weights of TOPSIS using AHP. We performed two different classification tests on each framework platform using data from a public dataset for PdM on biomedical equipment. Magnetometer data were used for test 1, and accelerometer data were used for test 2. We obtained the F1 score, flash memory, and latency metrics. There was a high level of consensus between the worlds of academia and industry when assigning the weights. Therefore, the overall comparison ranked the analyzed frameworks similarly. NanoEdgeAIStudio ranked first when considering all weights and industry only weights, and Edge Impulse was the first option when using academia only weights. In terms of performance, there is room for improvement in most frameworks, as they did not reach the metrics of the previously developed custom Edge AI solution. We identified some limitations that should be fixed to improve the comparison method in the future, like adding weights to the feature criteria or increasing the number and variety of performance tests.

Keywords: no-code; Edge AI; Predictive Maintenance; AHP; TOPSIS

check for updates

Academic Editor: Yin Zhang

Received: 24 April 2025
Revised: 19 May 2025
Accepted: 22 May 2025
Published: 26 May 2025

Citation: Montes-Sánchez, J.M.; Fernández-Cuevas, P.; Luna-Perejón, F.; Vicente-Díaz, S.; Jiménez-Fernández, Á. No-Code Edge Artificial Intelligence Frameworks Comparison Using a Multi-Sensor Predictive Maintenance Dataset. *Big Data Cogn. Comput.* **2025**, *9*, 145. https://doi.org/10.3390/bdcc9060145

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Edge AI

Artificial intelligence (AI) algorithms are experiencing an accelerated development that makes them more powerful and easier to use every year [1]. This also raises some concerns due to the large amount of energy and resources required to train and run these algorithms [2].

The use of Edge Computing (EC), in both standalone and cloud-based configurations, is one of the proposed solutions to address these concerns, since it allows some AI algorithms to run on small and simple computing devices that are also power-efficient, such as microcontrollers (MCUs) [3,4]. This is usually called Edge AI but is also called Tiny

Machine Learning (TinyML), and it provides additional benefits, such as improved data privacy and reduced network traffic [5]. Most Edge AI implementations are possible due to the development in 2017 of TensorFlow Lite, now renamed LiteRT [6], which is by far the most popular set of libraries for this purpose. The impact of Edge AI in the academic world also reflects a growing interest in Edge AI for various applications. Figure 1a shows the number of publications indexed in Web of Science that mention this topic (Edge AI, embedded AI, or TinyML).

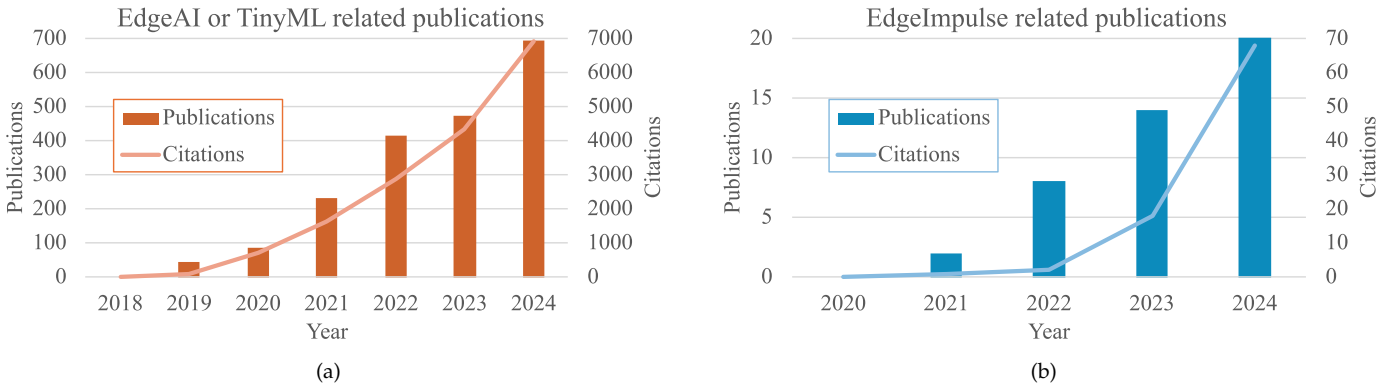

Figure 1. Number of publications and citations indexed in Web of Science with Edge AI, TinyML, or Embedded AI as topics (**a**) and Edge Impulse as topic (**b**). We found no publications with other frameworks' names as topics.

Despite being possible thanks to TensorFlow Lite since 2017, for years, deploying an Edge AI solution required the use of several different software and hardware tools that usually demanded specialized training in very different fields, such as MCU programming, data handling, AI configuration and training, and sometimes basic electronics, among others. This is one of the problems that the industry is facing when implementing Predictive Maintenance (PdM) solutions that can benefit from Edge-AI-based systems. In addition, training AI algorithms for a specific PdM application requires a large amount of data and several manual configuration processes that, at this time, are slowing down the adoption of PdM systems [7].

To address this challenge, some companies have developed software frameworks based on a graphical user interface (GUI) designed to allow Edge AI developers to collect data, train algorithms, and deploy them in Edge AI devices, all in the same software, with little or no programming at all, following the low-code or no-code philosophy [8]. Most of these frameworks are endorsed by the Edge AI Foundation (formerly called TinyML Foundation until November 2024) [9] and also by some MCU manufacturers.

The no-code philosophy has been successfully applied to other solutions that normally require programming skills, especially for the creation of web pages and desktop software [10]. The spread of AI Large Language Models (LLMs) has also led to a new form of creating programs without programming skills that is sometimes referred to as vibe coding. This method has been applied recently not only to create web pages and programs [11] but also to create tools for teaching the principles of machine learning (ML) [12] or the proper clinical response in medical faculties [13].

Edge AI no-code frameworks are not typically used or mentioned in major academic publications, with a few exceptions for Edge Impulse [14], as can be seen in Figure 1b. This could be due to the company being involved in providing materials and support to some researchers in the Edge AI field [15]. Although all analyzed frameworks have free or trial versions, some authors acknowledge the limitations of free versions in their

research [16], which could be one of the reasons why academics have not used these frameworks extensively yet, among others, as we will discuss later.

In this work, we compared six of the most popular no-code Edge AI frameworks in the market. We evaluated the performance of the frameworks in the same PdM scenario, taking into consideration the classification results from a previously developed custom solution, and tried to extract several metrics after experiencing the full development process with each one.

1.2. Multi-Attribute Decision Making (MADM)

For the final results, the metrics obtained, along with some additional information such as the license cost, were used as input criteria for the technique for order performance by similarity to the ideal solution (TOPSIS). This is a well-known method for multi-attribute decision-making (MADM) developed by Hwang and Yoon in 1981 [17]. It is suitable for software evaluation and is also faster than other common methods based on pairwise comparison, such as the analytic hierarchy process (AHP), because it significantly reduces paired comparisons [18]. AHP can also be used to decide the weights in TOPSIS, combining the two methods [19]. Despite the fact that new methods have been developed in recent years, such as fuzzy-based or data-driven approaches, AHP and TOPSIS are still in use today [20]. Other relevant applications of MADM methods, especially AHP and TOPSIS, include the COVID-19 pandemic [21] and 3D printing [22].

1.3. MADM and Edge AI: Closest Related Works

Few studies associate MADM with edge AI applications, and we found none that address no-code Edge AI frameworks. The closest work we found performs MADM for selecting edge AI hardware for different scenarios, following a workflow similar to the present study [23]. But that work does not mention any Edge AI software in addition to the libraries that are used in custom solutions.

Moreover, we found a good example of how MADM techniques can help in complex problems in the work [24], whose author proposes a decision-making application for healthcare. This work is highlighted in our study, since the author states that his approach could also benefit from Edge AI for future improvements but was unable to implement it at that time.

1.4. Structure of This Work

This work is structured as follows:

1. First, an introduction to Edge AI, PdM, and MADM is presented.
2. Then, the study workflow is described. Each agent of the workflow is detailed, including the no-code Edge AI frameworks evaluated, the weight selection with the AHP method, the TOPSIS technique, the dataset used in the tests, and the custom solution previously developed for this dataset.
3. The results of our study are presented next. This includes the performance metrics of the tests, the weights assigned by the experts with AHP, and the results of the TOPSIS comparisons.
4. These results are then analyzed in Section 4.
5. Finally, we present a conclusion where we point out the limitations of this study, some future improvements, and our prediction for the future of no-code Edge AI frameworks.

1.5. Contributions of This Work

The main contributions of this work include the following:

- An overview of the state of the art in the currently available no-code Edge AI frameworks and what each one offers.
- A proposal on how to apply AHP-TOPSIS to compare these frameworks.
- An insight into the opinions of both academia and industry experts regarding no-code Edge AI frameworks, which was also used in the AHP-TOPSIS method.
- The performance results with two already validated tests, and how each frameworks compare with each other in these PdM scenarios.
- A comparison between these results and a custom solution.
- A discussion about the future of no-code Edge AI frameworks.

2. Materials and Methods

2.1. Workflow

Figure 2 represents a graphical summary of the comparison workflow for this paper, with all actors involved. Each action represented in this figure is described in the following sections.

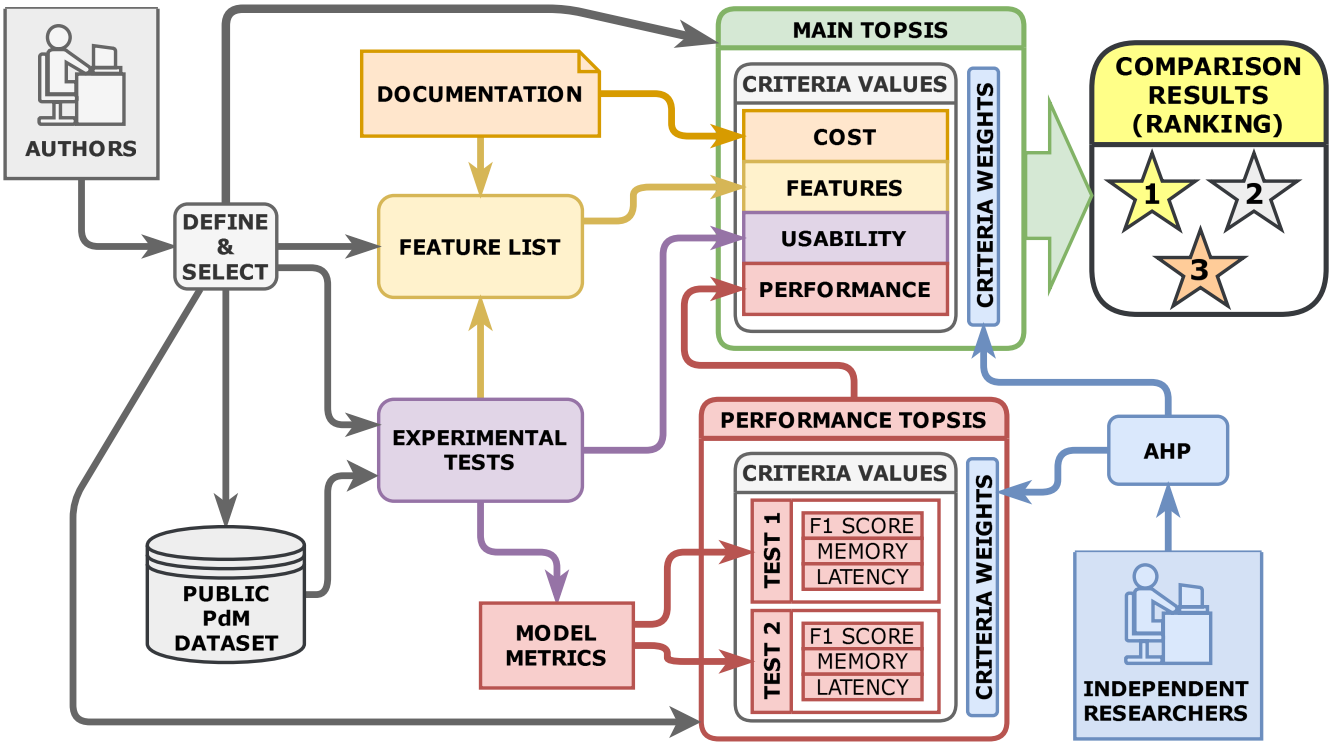

Figure 2. Workflow chart for no-code Edge AI frameworks ranking. The numbers 1, 2, 3 are intended to be a graphical representation of the top three positions of a ranking only.

2.2. No-Code Edge AI Frameworks

We have selected six different frameworks, which are detailed in Table 1. All of them offer a similar workflow for training and deploying neural networks on EC devices. These solutions can be used at no cost, but since some of them have commercially licensed versions, there may be some limitations. For the purpose of this work, we have only used the free or trial versions, but we highlighted when a limitation can be avoided by purchasing a license. All information and tests regarding this work are based on the latest available version of each framework in May 2025.

Table 1. Edge AI frameworks studied in this work.

Reference	Name	Company	Platform	Cost of Paid Version	Limitations of Free Version
[25]	DEEPCRAFT Studio (Imagimob)	Infineon	Local (Windows) with cloud training.	Quotation needed.	Limited training time to 3000 min/month. Limited license.
[14]	Edge Impulse	Qualcomm	Cloud	Quotation needed for Enterprise version.	Limited number of private projects. Limited compute resources.
[26]	NanoEdgeAIStudio	ST	Local (Windows, Linux)	Free	Commercial license included but only for STM32 microcontrollers.
[27]	Neuton.AI	Neuton.AI	Cloud	Free, but uses Google Cloud Platform which needs credits.	100 h of training included.
[28]	SensEI AutoML	TDK	Cloud	USD 100/month	No free version but offers a 30-day trial.
[29]	SensiML	QuickLogic	Cloud	From USD 291 monthly	Limited cloud storage and training time. Only public datasets. Only demo outputs.

2.3. Multi-Sensor Public Dataset for PdM

For the comparison, we used a public dataset for a PdM application that was recorded in a previous work. The dataset contains data obtained using a hydraulic block from biomedical equipment. This part has several peristaltic pumps whose condition worsens with use and time, and the dataset is intended to detect this aging process before a catastrophic failure occurs. It contains data from six different sensors, recorded simultaneously. It is available to the public in [30]. A thorough comparison of networks trained with it, with different configurations, has already been published [31]. Therefore, we can use the results of this previous implementation of neural networks, which was based on custom Python code using Tensorflow libraries, as a reference point. This solution is analyzed in Section 2.4.

In [31], the best results were obtained with magnetometer and accelerometer data. Consequently, for our test, we selected the LIS2MSL magnetometer dataset for performing *Test 1* and the LSM6DSOX accelerometer dataset for *Test 2* from the public repository. Following the dataset instructions, we also left out as holdout test data the samples suggested by the authors, and we will try to use them as testing data in each framework. All data will be imported into each framework as it is published, trying to avoid using external tools to reshape the files. Data preprocessing will be carried out externally only if the framework does not include internal tools to do that.

2.4. Custom Solution Based on Tensorflow/Keras

In the previous work with the selected dataset [30] the implementations of the Edge AI devices were performed without using any of the frameworks that we analyze in this work. Instead, a custom solution was programmed in Python using the Keras libraries [31] in the following sequence:

- First, a public dataset was created in CSV format using a commercial sensing device.
- Then, a machine learning network that includes a recurring neural network (RNN) with one or two layers was defined.
- After that, several combinations of RNN for each sensor, with different sampling rates, were trained locally using a GPU.
- Subsequently, the different results were analyzed to choose the best network for their application.

- Finally, by using external tools from the vendor ST (ST-CUBE-AI), the selected network was deployed on a microcontroller. This was only tested on ST boards (ARM architecture).

The results of this study were satisfactory in terms of accuracy, latency, memory, and power consumption:

- For the LIS2MSL magnetometer dataset at 100 Hz (Test 1 in our work), the best result metrics are as follows.
  - Model reference in the article: M4.
  - F1 weighted average (with holdout data): 1.00.
  - Memory: 21,624 bytes.
  - Latency (in ARM Cortex M4 80 Mhz): 3.85 ms.
- For the LSM6DSOX accelerometer at 6667 Hz (Test 2 in our work), the best result metrics are as follows.
  - Model reference in the article: A5.
  - F1 weighted average (with holdout data): 1.00.
  - Memory: 20,480 bytes.
  - Latency (in ARM Cortex M4 80 Mhz): 164.05 ms.

Despite the results being excellent, it is important to note that the effort and knowledge required for all steps cannot be achieved without deep ML and Python programming skills and several iterations. Even for someone with these skills, developing the code from scratch is very time-consuming.

This custom solution was not included in the TOPSIS comparison of the present study with the rest of the alternatives, as it was limited to no-code Edge AI frameworks only. However, their metrics were considered in the conclusions of this work and are also shown as a reference when needed.

2.5. Proposed Experimental Test

Since all frameworks offer the user a similar workflow, we defined the following steps as a training example:

1. Creating a new classification project for three classes. If the target needs to be selected in this step, we will try to select a generic ARM Cortex M4 processor running at 80 MHz, or similar. By doing so, we can fairly compare the metrics with the custom solution developed previously.
2. Dataset loading (magnetometer data for test 1, or accelerometer data for test 2). The preprocessing of the data is performed previously previously only if required, trying to use the tools included with the framework if any.
3. Model configuration. We will accept all suggestions made from the tool, if any.
4. Training models using all samples but the ones marked as recommended for holdout in the dataset. We will also accept all suggestions the tool makes for the selection of training parameters.
5. Validation of the models with holdout data, if possible.
6. Best model selection based on results with holdout data. Memory usage and inference time, when available, will also be considered.
7. Generating code for deployment. This will be performed for a generic ARM Cortex M4 architecture when possible.

The metrics obtained after this process will later be used in a first TOPSIS comparison that we called *performance TOPSIS*, and these results will be the performance criteria values in the final TOPSIS comparison of all frameworks (*main TOPSIS*), as described later in Section 2.7.

2.6. Feature List

We elaborated a list of features that the framework should have to perform an ideal PdM classification and deployment with the selected dataset. This list was decided by the consensus of all the authors of this work. We gave a punctuation mark to each framework based on the number of features in the list that they offer. Although we only used the free or trial versions in the experimental tests, we also included in this list a separate entry for each paid version, if available, for comparison. This score was later used as input for one of the criteria defined for the main TOPSIS analysis. All features are listed in Table 2. It is important to mention that the independent researchers received this list of features during the interview process (see Section 2.7.2), yet no weights were assigned to any features, and neither the authors nor the researchers were asked to rank the importance of the features. Each was equally considered in determining the final feature value used in the final comparison.

Table 2. Feature list for all frameworks. "N" means "no", "Y" means "yes".

			Data Handling											**Network Training**						**Validation**			**Deployment**		
	Price per Month ($)	**Total Features**	**Data Logger Included**	**CSV Data Support**	**Audio Data Support**	**Data Formatting Tools Included**	**Data Previewer and Analyzer**	**Allows Multiple Files for Each Class**	**Data Augmentation for All Data**	**Feature Extraction**	**Multi-Sensor Data Support**	**N-Class Classification Support**	**Custom Network Architecture Support**	**Unlimited Training Time**	**Cloud Training**	**Local Training**	**GPU Training**	**Suggests and Trains Network Variations**	**Holdout Validation Support**	**Memory Consumption Estimation**	**Inference Time Estimation**	**Generates Generic Code**	**Advanced Microcontroller Skills Not Required**	**Allows Commercial Use**
DEEPCRAFT Studio (free version)	0	15	Y	Y	Y	N	Y	Y	N	Y	Y	Y	Y	N	Y	N	N	Y	Y	Y	Y	Y	N	N
DEEPCRAFT Studio (paid version)	(1)	17	Y	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	N	Y
Edge Impulse (free version)	0	18	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	Y	N	Y	Y	Y	Y	Y	Y	N	Y
Edge Impulse (paid version)	(1)	19	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y
NanoEdgeAIStudio	0	16	Y	Y	Y	Y	Y	Y	N	N	Y	Y	N	Y	N	Y	N	Y	Y	Y	Y	Y	N	Y
Neuton.AI	0(2)	14	N	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	N	Y	Y	Y	Y	N	Y	N	Y
SensEI AutoML (pro version)	100	16	Y	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	N(3)	Y	Y	N
SensEI AutoML (Enterprise version)	(1)	19	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	N(3)	Y	Y	Y
SensiML (free version)	0	17	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	Y	N	Y	Y	Y	Y	Y	Y	N	N
SensiML (Proto version)	291	19	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y

(1) Unknown since it is a custom pricing. (2) Platform is free, but the user must pay for their use of the Google Cloud services. (3) Does not include estimation but incorporates inference results on actual hardware if available to the user.

2.7. TOPSIS Analysis

To obtain a final decision, we opted to use the TOPSIS method. This analysis requires the user to define the comparison criteria and assign weights to each one based on their importance.

2.7.1. Selecting Criteria

The criteria were determined by consensus by the authors of this work based on their experience as Edge AI researchers and are detailed in Figure 2: cost, features, usability and performance. The performance criteria value is the result of a nested TOPSIS analysis performed on the metrics that we call the *performance TOPSIS*. We also determined the performance TOPSIS criteria (F1 score, memory and latency for both tests) in the same way as we did with the main TOPSIS.

2.7.2. Selecting Weights with AHP

We tried to minimize the subjectivity of these decisions by fixing the weights using the AHP method. Ten independent Edge AI researchers from academia and the industry sector (see Table 3) were asked to perform AHP among the different criteria. This method has previously been used by other researchers in combination with TOPSIS [19]. We used the tools provided in the work [32], which also describes in detail the necessary steps and formulas.

Table 3. Profile summary of the experts consulted for the AHP weighting process.

Researcher ID	Sector	Academic Degree	Main Field of Application
R1	Academia	PhD Computer Science	Medical and neuromorphic
R2	Academia	PhD Computer Science	Medical
R3	Academia	PhD Computer Science	Neuromorphic
R4	Academia	PhD Computer Science	Wearables
R5	Academia	PhD Computer Science	Wearables, IoT
R6	Academia	PhD Computer Science	Medical
R7	Industry	Msc Computer Science	Biomedical devices
R8	Industry	Msc Computer Science	Biomedical devices
R9	Industry	Msc Computer Science	Aeronautics
R10	Industry	Bachelor Degree Computer Science	Elevators

Since we have two TOPSIS comparisons, called the main TOPSIS (for obtaining the final ranking) and the performance TOPSIS (for obtaining the performance values used in the main TOPSIS), we had to perform the AHP method two times per researcher. We presented each researcher with a detailed explanation of the goal, what each criterion means, and where its value comes from:

- AHP for the main TOPSIS:
  - Goal: Determine the most suitable no-code Edge AI framework among all alternatives in a PdM scenario for the biomedical field.
  - Alternatives: All frameworks from Table 1.
  - Criteria: Cost, number of features (the feature list is provided), usability, and performance. It is explained that performance is a value obtained from the sub-criteria in the performance TOPSIS.
- AHP for the performance TOPSIS:
  - Goal: Determine the best-performing no-code edge AI framework when tested with two datasets coming from the same PdM scenario in terms of accuracy, memory size, and latency.
  - Alternatives: All frameworks from Table 1.
  - Criteria: Test 1 F1 score, Test 1 Memory, Test 1 Latency, Test 2 F1 score, Test 2 Memory, and Test 2 Latency. Each researcher is provided with information about what each criterion represents and the nature of Tests 1 and 2.

The AHP method requires the following steps and equations. For each researcher, we calculated and normalized an independent matrix and then consolidated all matrices in the last step:

1. First, we structured the decision hierarchy. In this case, we decided to assign weights using AHP to the main TOPSIS and the performance TOPSIS, as previously stated. This hierarchy is also described in Section 2.1. Therefore, two different AHP matrices were obtained for each researcher. The following steps must be repeated for each TOPSIS.
2. For each k researcher participant, we constructed a pairwise comparison matrix $A_{(k)}$. Given n attributes, we organized the pairwise comparison of all attributes with each other in a square matrix, where $a_{ij(k)}$ represents how attribute i is prioritized relative to attribute j on a scale of 1 to 9. These values were obtained from independent researchers during the interview process.

$$A_{n\times n} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \tag{1}$$

   Note that $a_{ij(k)} = 1$ when $i = j$ (comparison of the importance of a criterion with itself). Also, $a_{ji(k)} = 1/a_{ij(k)}$.
3. We computed the principal eigenvector w_k (weights) for each $A_{(k)}$:

$$A_{(k)}w_{(k)} = \lambda_{\max(k)}w_{(k)} \tag{2}$$

   where $\lambda_{\max(k)}$ is the largest eigenvalue of $A_{(k)}$.
4. The weights were normalized to achieve

$$\sum_{i=1}^{n} w_{i(k)} = 1 \tag{3}$$

5. We searched for inconsistencies in the answers of the researchers. If the consistency ratio CR was greater than 10%, then we prompted the interviewee to review their answers. To verify consistency,

$$CI_{(k)} = \frac{\lambda_{\max(k)} - n}{n-1} \tag{4}$$

$$CR_{(k)} = \frac{CI_{(k)}}{RI} \tag{5}$$

   where CI is the Consistency Index and RI is the Random Index, which are predefined for a matrix of size n. For example, for $n = 3$, $RI = 0.58$. For $n = 4$, $RI = 0.90$.
6. We obtained the desired consolidated matrix C which is a combination of several decision matrices $A_{(k)}$. We obtained three different C matricies for each AHP: using all participants $A_{(k)}$, using only participants from the academic world $A_{(k_{academia})}$, and using only participants from the industry $A_{(k_{industry})}$. Since we used AHP for the weights of the performance TOPSIS and also for the weights of the main TOPSIS, this results in six different C matrices in total. Each C is obtained by using a geometric mean of $a_{ij(k)}$ elements of the selected $A_{(k)}$:

$$C_{n\times n} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix} \tag{6}$$

$$c_{ij} = \left(\prod_{n=1}^{k} a_{ij(k)} \right)^{\frac{1}{k}} = \exp\left(\frac{1}{k} \sum_{n=1}^{k} \ln a_{ij(k)} \right) \tag{7}$$

The C matrix also follows the same rules as $A_{(k)}$ matrices, so $c_{ij} = 1$ when $i = j$, and $c_{ji} = 1/c_{ij}$.

7. Finally, the weights were computed in the same way as we they were for each $A_{(k)}$ previously, by finding the eigenvector w and normalizing it to sum to 1 (see Equations (2) and (3)).

2.7.3. Criteria Values

The values of usability criteria were determined by the authors' experience using each tool (a mean of five punctuation marks between 1 and 9, with 9 being the best score). We explain the reasons behind this score in Section 3. All other criteria values were obtained from metrics or the information provided in the framework documentation, as previously described in Sections 2.5 and 2.6.

2.7.4. Calculation of TOPSIS Results

We added the weights (obtained with the AHP method as described in Section 2.7.2) and the values of all criteria into the TOPSIS analysis. Cost, memory, and latency were considered as negative criteria, since a lower value is better. After that, we performed a series of calculations to obtain the relative closeness value $\mathcal{C}$ for each alternative in the performance TOPSIS first and then in the main TOPSIS. The higher this value, the closer to the ideal solution. The authors of [18,19] described these calculations in detail.

TOPSIS steps and equations are as follows:

1. Normalize the decision matrix:

$$R_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{n} x_{ij}^2}} \tag{8}$$

2. Calculate the weighted normalized decision matrix:

$$V_{ij} = w_j \cdot R_{ij} \tag{9}$$

3. Determine the positive ideal (A^+) and negative ideal (A^-) solutions:

$$A^+ = \{\max(V_{ij}) | j \in J, \min(V_{ij}) | j \in J'\} \tag{10}$$

$$A^- = \{\min(V_{ij}) | j \in J, \max(V_{ij}) | j \in J'\} \tag{11}$$

4. Calculate the separation measures:

$$S_i^+ = \sqrt{\sum_{j=1}^{m} (V_{ij} - A_j^+)^2} \tag{12}$$

$$S_i^- = \sqrt{\sum_{j=1}^{m} (V_{ij} - A_j^-)^2} \tag{13}$$

5. Finally, calculate the relative closeness to the ideal solution:

$$C_i^* = \frac{S_i^-}{S_i^+ + S_i^-} \tag{14}$$

3. Results

3.1. AHP Results (Assigned Weights)

Table 4 presents the criteria and the weights assigned by the AHP method for the performance TOPSIS and the main TOPSIS. We computed the results for all participants but also the separated weights for academia researchers only and for industry researchers only. The individual weights are also represented for reference in Figure 3.

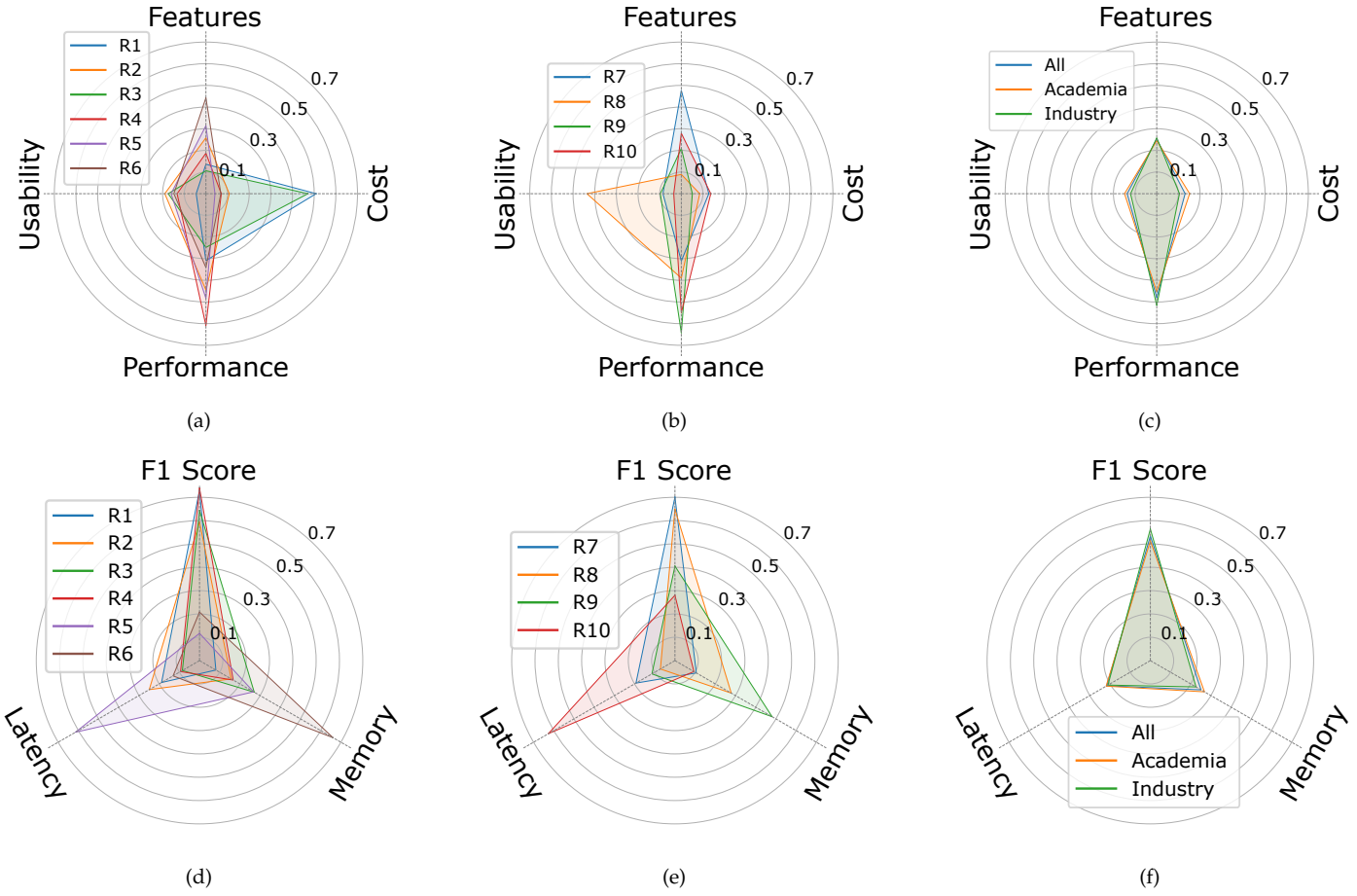

Figure 3. AHP results for the calculation of weights. Charts (**a**,**d**) illustrate the individual weights assigned by academic researchers (R1–R6) for the main TOPSIS criteria and the performance TOPSIS criteria (combining values for Test 1 and Test 2), respectively. Charts (**b**,**e**) present the weights assigned by industry researchers (R7–R10) for the same criteria categories. Finally, charts (**c**,**f**) show the consolidated weights obtained by aggregating the weights from academic, industry, and all researchers.

A quick analysis of these results shows that there was consensus among most researchers. Since we used a geometric mean for consolidation, the few extreme values of some researchers were mitigated. Therefore, the consolidated weights were very similar in all situations.

Table 4. TOPSIS criteria with AHP results for the calculation of weights.

	Criteria	Unit	Value Source	Consolidated Weight		
				w_{all}	$w_{academia}$	$w_{industry}$
Main TOPSIS	Cost	USD/month	Documentation	0.131	0.152	0.105
	Features	Integer	Feature list	0.252	0.245	0.258
	Usability	1 to 9 scale	Mean of authors opinions	0.137	0.149	0.121
	Performance	Relative closeness $\mathcal{C}$	Performance TOPSIS	0.480	0.453	0.516
Performance TOPSIS	Test 1 F1 score	Real number	Metrics from magnetometer experimental test	0.267	0.257	0.282
	Test 1 Memory	Bytes		0.126	0.134	0.113
	Test 1 Latency	ms		0.108	0.110	0.105
	Test 2 F1 score	Real number	Metrics from accelerometer experimental test	0.267	0.257	0.282
	Test 2 Memory	Bytes		0.126	0.134	0.113
	Test 2 Latency	ms		0.108	0.110	0.105

3.2. Usability

We gave a score between 1 and 9, with 9 being the best possible, based on our own experience when performing the two experimental tests. The main problems we faced that lowered the usability scores of the frameworks and the final score for each one were the following:

1. The DEEPCRAFT Studio requires an installation on a Windows PC. It has little flexibility when importing data since it requires a very specific format and does not include tools for this preparation. Some AI training knowledge is required to select the training settings. The latency estimation is given in cycles instead of in seconds. The paid version does not offer improvements to any of these limitations. The usability score was 4 for both the free and paid versions.
2. Edge Impulse offered the best user experience, and the process could be completed even without reading the documentation. Therefore, the usability score was 9 (maximum) for both the free and paid versions.
3. NanoEdgeAIStudio requires an installation on Windows or Linux. It runs locally and only has CPU training support, which could mean slower training time in some systems. Data formatting tools are included, but it is not clear how to use them to obtain the desired input format. The usability score was 6.
4. The Neuton.AI data loading process was difficult, as it required a specific format and did not include data formatting tools to perform it. The training time for Test 2 was very long (more than 18 h). The metrics do not include latency estimations. The usability score was 5.
5. SensEI AutoML asks for credit card information even for the trial version. It requires a desktop tool (available for Mac and Windows) and a compatible browser (only Chrome or Edge). External data import is difficult, since it only supports integer data, and the conversion from real numbers to integer numbers of the readings must be performed with external tools. Initially, the data import proceeded without any warnings. However, due to an incorrectly formatted timestamp, the build phase failed, failing to

identify the specific issue, which we discovered through our own investigation. The training is easy to set up, and the tool provides automatic suggestions for all settings. We experienced some graphical bugs after training in the tables of results. Finally, latency estimations, although possible to obtain according to the documentation, seem to require specific physical hardware. Therefore, it is not an estimation, but a real test, and we think this is a limitation. For these reasons, the usability score was low: 2 for the Pro version and 3 for the Enterprise version (slightly higher since it is stated to include local training that should improve the experience in some cases).

6. The SensiML workflow was perplexing because it suggests installing a required desktop application for successful data loading, yet the remaining steps must be completed online. We think this was not clearly stated in the documentation. Using the documentation as a guide and working with the website, we were able to conduct the tests smoothly. However, having only 1 h of training per month restricts its practicality for more than one or two test runs. The usability score was 4 for the free version and 5 for the Proto version (slightly higher since it is stated to include more training time).

3.3. Experimental Test Results and Performance TOPSIS

We were unable to obtain latency estimations from the SensEI AutoML or Neuton.AI frameworks. Therefore, neither of those received any latency score. The metrics obtained and the final $\mathcal{C}$ for each tool can be seen in Table 5. As stated in Section 2.7.4, $\mathcal{C}$ is a mathematical value that determines how close each framework is to the ideal solution. The maximum value that $\mathcal{C}$ can be is 1; therefore, the higher the better.

Table 5. Performance TOPSIS results.

		Weight W_t (All)	Weight W_{ta} (Academia)	Weight W_{ti} (Industry)	DEEPCRAFT Studio	Edge Impulse	NanoEdgeAI Studio	Neuton.AI	SensEI AutoML	SensiML	Custom Solution *
Test 1	**F1 score**	0.267	0.257	0.282	0.99	0.37	1.00	0.69	1.00	0.39	1.00
	Memory (Bytes)	0.126	0.134	0.113	54,488	15,974	40,550	2500	37,417	3198	21,624
	Latency (ms)	0.108	0.113	0.105	17.42	11	6.1	None	None	0.09	3.85
Test 2	**F1 score**	0.267	0.257	0.282	0.45	0.97	0.99	0.51	0.78	0.51	1.00
	Memory (Bytes)	0.126	0.134	0.113	12,164	17,920	17,510	2600	16,015	4140	20,480
	Latency (ms)	0.108	0.113	0.105	183.45	623.00	2.30	None	None	64.20	164.05
Closeness value $\mathcal{C}$	**Using W_t**				0.55	0.57	0.67	0.44	0.42	0.57	
	Using W_{ta}				0.55	0.58	0.65	0.45	0.41	0.59	
	Using W_{ti}				0.56	0.57	0.69	0.42	0.45	0.54	
Rank position	**Using W_t**				4	2	1	5	6	3	
	Using W_{ta}				4	3	1	5	6	2	
	Using W_{ti}				3	2	1	6	5	4	

* Metrics obtained in a previous study [31]. Not part of the TOPSIS comparison since it is not based on a no-code Edge AI framework.

3.4. Main TOPSIS Results

We compared all frameworks in the main TOPSIS by adding the cost (see Table 1), the features (see Table 2), the usability score, and the performance value (see Table 5). The

weights assigned by ten independent researchers were also added (see Table 4). Since we did not have access to the paid versions of the tools, we considered that the metrics of the paid versions are exactly the same as those obtained with their free versions, and therefore, the performance value will also be the same.

The main TOPSIS with the final $\mathcal{C}$ results can be seen in Table 6. We also obtained the results of the main TOPSIS but using only the consolidated weights of the industry and the academic researchers only to see if a difference in the ranking positions appears when doing so.

Table 6. Main TOPSIS Results.

		DEEPCRAFT Studio (Free Version)	DEEPCRAFT Studio (Paid Version)	Edge Impulse (Free Version)	Edge Impulse (Paid Version)	NanoEdgeAIStudio	Neuton.AI	SensEI AutoML (Pro Version)	SensEI AutoML (Enterprise Version)	SensiML (Free Version)	SensiML (Proto Version)
Cost		0	300 *	0	300 *	0	0 **	100	300 *	0	291
Features		15	17	18	19	16	14	16	19	17	19
Usability		4	4	9	9	6	5	2	3	4	5
Performance	**All**	0.55	0.55	0.57	0.57	0.67	0.44	0.42	0.42	0.57	0.57
	Academia	0.55	0.55	0.58	0.58	0.65	0.45	0.41	0.41	0.59	0.59
	Industry	0.56	0.56	0.57	0.57	0.69	0.42	0.45	0.45	0.54	0.54
$\mathcal{C}$	**All**	0.59	0.34	0.78	0.51	0.79	0.48	0.33	0.19	0.62	0.41
	Academia	0.61	0.32	0.84	0.50	0.78	0.55	0.36	0.18	0.66	0.41
	Industry	0.55	0.39	0.69	0.52	0.80	0.38	0.29	0.21	0.53	0.40
Rank position	**All**	4	8	2	5	1	6	9	10	3	7
	Academia	4	9	1	6	2	5	8	10	3	7
	Industry	3	7	2	5	1	8	9	10	4	6

* Since it is a custom pricing we had to set an arbitrary value higher than the rest that might not be accurate. ** Up to USD 500 in Google Cloud services are included, but extensive use might not be free.

4. Discussion

In this work, we have compared six different no-code Edge AI frameworks using the AHP-TOPSIS method. The comparison took into account the economic cost, the number of features, usability, and performance.

Cost information was not available for some options because they are custom-priced. Since the comparison method expects a value for all criteria, we had to assume that the cost of those options was the most expensive. This should be improved in future comparisons.

We defined a list of desirable features that each tool should have to be used in PdM Edge AI scenarios. Among the free versions of the tools, Neuton.AI was the one that offered fewer features (14), and EdgeImpulse was the one with the most (18). In the feature list, we also included the number of features of the paid versions of some frameworks based on each company's claims.

For performance criteria, we designed an experimental test using multisensor PdM data from a public dataset. We conducted two different classification tests on each framework platform: one with magnetometer data (test 1) and one with accelerometer data

(test 2). We obtained the following metrics after each test directly from the framework: F1 score, flash memory consumption, and latency. We were unable to obtain latency estimation from the Neuton.AI and SensEI AutoML frameworks. Therefore, they were penalized in the performance comparison, which was also the case with the AHP-TOPSIS method. In terms of performance only, the best results were obtained with the NanoEdgeAIStudio framework, with a closeness value $\mathcal{C}$ of 0.65.

In general, the models we obtained using these tools have room for improvement, as none of them reached the metrics of the previous study that used a custom solution [31], at least for the two tests at the same time. The best solution for Test 2 from NanoEdgeAI Studio was the only one that we can consider better, since it improved the memory and latency of the custom solution while maintaining a high F1 score of 0.99, very close to the 1.00 score achieved by the previous work.

In terms of usability, the frameworks in which we faced the most problems during the tests were the most penalized. We gave a rating from 1 to 9 based on our own user experience during the tests. SensEI AutoML was ranked worst in terms of usability due to problems with data import and some graphical bugs, and Edge Impulse was the framework that offered the best user experience, in the authors' opinion. Despite offering a good user experience overall, the training time limitations of the free version of SensiML were so strict that they made it impractical for real use. This was also reflected in the usability value.

The overall comparison ranked NanoEdgeAIStudio first, followed by the free version of Edge Impulse, and the free version of SensiML in third place. Since we considered that the performance results would remain unchanged by using the paid versions, and the number of features would not not increase significantly, all free versions were ranked above the paid versions of the same framework. These results could be made more accurate by adding weights to the features instead of using a feature counter.

Since the consolidated weights for all participants were very similar to the consolidated weights for academia only and industry only, the closeness value also remained similar, as can be seen in Figure 4. In this figure, it is also very clear that most frameworks were very close to each other, so a small change in closeness slightly changed the ranking positions. For example, for academia researchers, EdgeImpulse (free version) was ranked first instead of NanoEdgeAIStudio, and industry researchers ranked DEEPCRAFT Studio (free version) in third place instead of SensiML (free version).

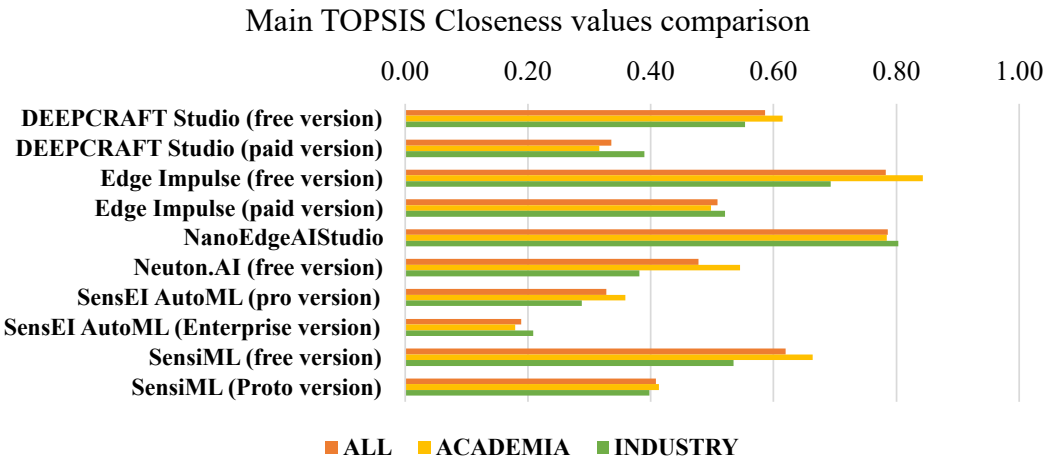

Figure 4. Closeness value $\mathcal{C}$ from the main TOPSIS in a graphical comparison.

5. Conclusions

The application of the AHP-TOPSIS method has been proven useful for MADM problems, and it was also successfully applied to our novel case scenario: determining

which no-code Edge AI framework we should use in a PdM scenario for the biomedical field in both academic and industry applications. We presented a series of steps that we think are easy to apply but at the same time reduce subjectivity.

During the process, we obtained useful information not only about the characteristics of the frameworks but also the valuable opinions of ten independent researchers from academia and industry. Despite coming from different worlds, the average opinion on the importance of each criterion was very similar, so the ranking was also very similar for both worlds.

After carrying out the whole process for all frameworks, NanoEdgeAIStudio was the one that ranked first, followed by Edge Impulse and SensiML. Despite these results, the custom solution proposed in the original article for the same data was better in performance than all models trained during all tests but one (the Test 2 model from NanoEdgeAIStudio). In summary, the main advantage of these frameworks continues to be their ease of use rather than their performance; however, they are on track to become a superior option in nearly all aspects in the near future for most simple applications.

We also acknowledge some limitations in our study that should be considered in a future improved version. More tests with different data would make the performance metric more accurate. Since most frameworks had issues with data import, carrying this out could be challenging. In addition, there is still a lack of quality public PdM datasets available, but this is currently improving. We should also carry out the performance tests separately in the paid versions of each framework to fully reflect the possible variations in the results. The feature list could also be improved by assigning weights to each feature using the AHP method, and the weights already assigned could also be improved by performing more interviews with different experts. Finally, other MADM techniques could be applied for comparison instead of AHP-TOPSIS, like fuzzy-based or data-driven approaches.

Despite the weak points detected, we anticipate that these tools will undergo enhancements and expansions in the future, with new additional Edge AI frameworks likely to be introduced in the upcoming years. In fact, during the time we were writing this study, the EdgeImpulse framework expanded its free plan to include GPU training and a commercial license at no cost. This has already been incorporated into the present work. We can expect these tools to introduce LLMs as a new interaction with users, as other no-code tools are already implementing [13], to add more complex models that are suitable for more powerful devices, and to focus on multi-platform cloud-based solutions.

Author Contributions: Conceptualization, J.M.M.-S., S.V.-D. and Á.J.-F.; Data curation, J.M.M.-S. and P.F.-C.; Formal analysis, J.M.M.-S., P.F.-C. and F.L.-P.; Funding acquisition, S.V.-D. and Á.J.-F.; Investigation, J.M.M.-S., P.F.-C. and F.L.-P.; Methodology, J.M.M.-S., S.V.-D. and Á.J.-F.; Project administration, S.V.-D. and Á.J.-F.; Resources, J.M.M.-S., S.V.-D. and Á.J.-F.; Software, J.M.M.-S.; Supervision, S.V.-D. and Á.J.-F.; Validation, J.M.M.-S.; Visualization, J.M.M.-S. and F.L.-P.; Writing—original draft, J.M.M.-S. and P.F.-C.; Writing—review and editing, J.M.M.-S., F.L.-P., S.V.-D. and Á.J.-F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by Proyecto PREDICAR (ref. PID2023-149777OB-I00) from Agencia Estatal de Investigación, Gobierno de España.

Data Availability Statement: The data used for the performance tests are available at idUS (Universidad de Sevilla public research data repository) at https://doi.org/10.12795/11441/162880 (accessed on 16 May 2025). This data are also cited in the text as [30]. The data recording is thoroughly explained in the article [31], which is available at https://doi.org/10.1109/TR.2024.3488963 (accessed on 16 May 2025). The latter is the one the authors recommend when citing the source of the data.

Acknowledgments: We express our gratitude to the independent researchers from both academia and industry who willingly participated in our interviews for this work.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
AHP	Analytic Hierarchy Process
CPU	Central Processing Unit
CSV	Comma-separated Values
EC	Edge Computing
Edge AI	Edge Artificial Intelligence
GPU	Graphics Processing Unit
GUI	Graphical User Interface
LLM	Large Language Model
MADM	Multi-attribute Decision Making
MCU	Microcontroller Unit
ML	Machine Learning
PdM	Predictive Maintenance
RNN	Recurrent Neural Network
TinyML	Tiny Machine Learning
TOPSIS	Technique for Order Performance by Similarity to the Ideal Solution

References

1. Rammer, C.; Fernández, G.P.; Czarnitzki, D. Artificial intelligence and industrial innovation: Evidence from German firm-level data. *Res. Policy* **2022**, *51*, 104555. [CrossRef]
2. Jiang, P.; Sonne, C.; Li, W.; You, F.; You, S. Preventing the Immense Increase in the Life-Cycle Energy and Carbon Footprints of LLM-Powered Intelligent Chatbots. *Engineering* **2024**, *40*, 202–210. [CrossRef]
3. Saso, K.; Hara-Azumi, Y. Revisiting Simple and Energy Efficient Embedded Processor Designs Toward the Edge Computing. *IEEE Embed. Syst. Lett.* **2020**, *12*, 45–49. [CrossRef]
4. Nain, G.; Pattanaik, K.K.; Sharma, G.K. Towards Edge Computing in intelligent manufacturing: Past, present and future. *J. Manuf. Syst.* **2022**, *62*, 588–611. [CrossRef]
5. Ghosh, A.M.; Grolinger, K. Edge-Cloud Computing for Internet of Things Data Analytics: Embedding Intelligence in the Edge With Deep Learning. *IEEE Trans. Ind. Inform.* **2021**, *17*, 2191–2200. [CrossRef]
6. Google LLC. LiteRT for Microcontrollers. 2024. Available online: https://ai.google.dev/edge/litert/microcontrollers/overview (accessed on 16 May 2025).
7. Scaife, A.D. Improve Predictive Maintenance through the application of artificial intelligence: A systematic review. *Results Eng.* **2024**, *21*, 101645. [CrossRef]
8. Guthardt, T.; Kosiol, J.; Hohlfeld, O. Low-code vs. the developer: An empirical study on the developer experience and efficiency of a no-code platform. In Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, Linz, Austria, 22–27 September 2024; pp. 856–865.
9. Edge AI Foundation. Edge AI Foundation Webpage. 2025. Available online: https://www.edgeaifoundation.org/ (accessed on 16 May 2025).
10. Silva, J.X.; Lopes, M.; Avelino, G.; Santos, P. Low-code and No-code Technologies Adoption: A Gray Literature Review. In Proceedings of the XIX Brazilian Symposium on Information Systems, Maceió, Brazil, 29 May–1 June 2023; SBSI '23, pp. 388–395. [CrossRef]
11. Monteiro, M.; Branco, B.C.; Silvestre, S.; Avelino, G.; Valente, M.T. NoCodeGPT: A No-Code Interface for Building Web Apps With Language Models. *Softw. Pract. Exp.* 2025, *online version*. [CrossRef]
12. Sundberg, L.; Holmström, J. Teaching tip: Using no-code AI to teach machine learning in higher education. *J. Inf. Syst. Educ.* **2024**, *35*, 56–66. [CrossRef]
13. Chow, M.; Ng, O. From technology adopters to creators: Leveraging AI-assisted vibe coding to transform clinical teaching and learning. *Med. Teach.* **2025**, 1–3. [CrossRef] [PubMed]
14. Edge Impulse. Edge Impulse—The Leading Platform for Embedded Machine Learning. 2025. Available online: https://edgeimpulse.com/ (accessed on 16 May 2025).

15. Okoronkwo, C.; Ikerionwu, C.; Ramsurrun, V.; Seeam, A.; Esomonu, N.; Obodoagwu, V. Optimization of Waste Management Disposal Using Edge Impulse Studio on Tiny-Machine Learning (Tiny-ML). In Proceedings of the 2024 IEEE 5th International Conference on Electro-Computing Technologies for Humanity (NIGERCON), Ado Ekiti, Nigeria, 26–28 November 2024; pp. 1–5.
16. Diab, M.S.; Rodriguez-Villegas, E. Performance evaluation of embedded image classification models using edge impulse for application on medical images. In Proceedings of the 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Glasgow, UK, 11–15 July 2022; pp. 2639–2642.
17. Hwang, C.L.; Yoon, K.; Hwang, C.L.; Yoon, K. Methods for multiple attribute decision making. In *Multiple Attribute Decision Making: Methods and Applications a State-of-the-Art Survey*; Springer: Berlin/Heidelberg, Germany, 1981; pp. 58–191. [CrossRef]
18. Zaidan, A.; Zaidan, B.; Hussain, M.; Haiqi, A.; Kiah, M.M.; Abdulnabi, M. Multi-criteria analysis for OS-EMR software selection problem: A comparative study. *Decis. Support Syst.* **2015**, *78*, 15–27. [CrossRef]
19. Karim, R.; Karmaker, C.L. Machine selection by AHP and TOPSIS methods. *Am. J. Ind. Eng.* **2016**, *4*, 7–13.
20. Sahoo, S.K.; Goswami, S.S. A comprehensive review of multiple criteria decision-making (MCDM) Methods: Advancements, applications, and future directions. *Decis. Mak. Adv.* **2023**, *1*, 25–48. [CrossRef]
21. Sotoudeh-Anvari, A. The applications of MCDM methods in COVID-19 pandemic: A state of the art review. *Appl. Soft Comput.* **2022**, *126*, 109238. [CrossRef] [PubMed]
22. Qin, Y.; Qi, Q.; Shi, P.; Lou, S.; Scott, P.J.; Jiang, X. Multi-attribute decision-making methods in additive manufacturing: The state of the art. *Processes* **2023**, *11*, 497. [CrossRef]
23. Canpolat Şahin, M.; Kolukısa Tarhan, A. Evaluation and Selection of Hardware and AI Models for Edge Applications: A Method and A Case Study on UAVs. *Appl. Sci.* **2025**, *15*, 1026. [CrossRef]
24. Aljohani, A. AI-Driven decision-making for personalized elderly care: A fuzzy MCDM-based framework for enhancing treatment recommendations. *BMC Med. Inform. Decis. Mak.* **2025**, *25*, 119. [CrossRef] [PubMed]
25. Imagimob. Imagimob's DEEPCRAFT™ Studio. 2025. Available online: https://www.imagimob.com/deepcraft/ (accessed on 16 May 2025).
26. STMicroelectronics. NanoEdge AI Studio—Development Tool. 2025. Available online: https://www.st.com/en/development-tools/nanoedgeaistudio.html (accessed on 16 May 2025).
27. Neuton AI. Neuton AI—Tiny Machine Learning Platform. 2025. Available online: https://neuton.ai (accessed on 16 May 2025).
28. TDK Corporation. TDK Sensei—AutoML for Embedded AI. 2025. Available online: https://sensei.tdk.com/automl (accessed on 16 May 2025).
29. SensiML Corporation. SensiML—AI Development Tools for Edge Devices. 2025. Available online: https://sensiml.com/ (accessed on 16 May 2025).
30. Montes-Sánchez, J.; Uwate, Y.; Nishio, Y.; Jiménez-Fernández, A.; Vicente-Díaz, S. Peristaltic pump aging detection dataset. *idUS (Depósito Investig. Univ. Sevilla)* **2024**. [CrossRef]
31. Montes-Sánchez, J.M.; Uwate, Y.; Nishio, Y.; Vicente-Díaz, S.; Jiménez-Fernández, Á. Predictive Maintenance Edge Artificial Intelligence Application Study Using Recurrent Neural Networks for Early Aging Detection in Peristaltic Pumps. *IEEE Trans. Reliab.* 2024, *early access*, 1–15. [CrossRef]
32. Goepel, K.D. Implementing the analytic hierarchy process as a standard method for multi-criteria decision making in corporate enterprises—A new AHP excel template with multiple inputs. In Proceedings of the International Symposium on the Analytic Hierarchy Process, Kuala Lumpur, Malaysia, 19–23 June 2013; Creative Decisions Foundation: Kuala Lumpur, Malaysia, 2013; Volume 2, pp. 1–10.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.