LPES: Tiny ML Rain Detector

Tommaso Simoncelli
Sebastian Weiss
tommaso.simoncelli@studenti.unitn.it
sebastian.weiss@studenti.unitn.it
University of Trento
Povo, Trento, Italy

ABSTRACT

The objective of this project is the detection of rain using Tiny Machine Learning (TinyML) on the Arduino Nano 33 BLE Sense platform, combined with the Edge Impulse framework. To address this problem, we developed a battery-less, ultra-low-power embedded system capable of real-time classification of two environmental conditions: Idle and Rain. The system is fully autonomous and self-powered, relying on a supercapacitor for energy storage and a solar panel for energy harvesting. Everything is managed by a dedicated Power Management Unit (PMU) that ensures efficient charging and stable operation even in variable lighting conditions.

We selected low-power components and integrated a Real-Time Clock (RTC) to precisely control the wake-up cycles of the device, reducing active time and consequently saving energy. The TinyML model was optimized for execution on a constrained microcontroller, with a focus on minimizing RAM and flash usage, as well as reducing inference time to lower energy cost per prediction.

Thanks to a carefully engineered feature extraction phase, the system prepares signal statistics in order to simplify classification. This preprocessing step significantly reduces the model's complexity, enabling the use of lightweight architectures with high accuracy and low computational overhead.

Power consumption is further minimized by enabling a deep sleep mode, where most of the systems components are switched off and only wake up upon external interrupts (e.g., from the RTC or manual triggers). The resulting architecture is ideal for Internet of Things (IoT) scenarios in which long-term deployment, energy independence, and environmental sensing are crucial.

1 MOTIVATION

Environmental monitoring plays a crucial role in modern smart systems, especially when it comes to the detection of weather phenomena such as rainfall. In many IoT scenarios—ranging from agriculture to environmental risk prevention—detecting rain accurately and in real time can enable prompt actions, such as closing automated irrigation systems or alerting about potential flooding events.

Traditional rain detection systems rely on meteorological sensors, which often involve bulky hardware, regular maintenance, and more importantly non-negligible energy consumption. These limitations make them unsuitable for deployment in remote, battery-powered, or maintenance-constrained environments.

In contrast, acoustic-based rain detection provides a lightweight and low-cost alternative. By recognizing the unique sound patterns of raindrops, it is possible to detect rainfall without the need for expensive and energy-intensive weather sensors. This approach is particularly advantageous in off-grid applications, where ultra-low-power consumption and autonomous operation are essential.

Our project explores the potential of sound-based rain classification using TinyML on the Arduino Nano 33 BLE Sense platform. By leveraging efficient models and energy-harvesting techniques, we aim to create a reliable, self-powered system for environmental monitoring with minimal resource usage.

2 LIMITATIONS OF THE STATE OF THE ART

2.1 Current Approaches

Several approaches for environmental sensing and rain detection have been explored in the literature. Traditional methods often rely on *humidity sensors*, *infrared reflectivity*, or *pressure sensors* to detect water presence on surfaces, as seen in commercial automotive systems. More recent approaches integrate *deep learning* models applied to *video data*, particularly in the automotive field using ResNet-based classifiers on embedded systems such as Jetson Nano and Raspberry Pi 4 [7].

In terms of energy autonomy, many low-power Wireless Smart Sensor Networks (WSSNs) adopt *solar energy harvesting systems* backed by *rechargeable batteries*, mostly LiPo types [9]. Some platforms [10] propose event-driven low-power IoT nodes with *microphone-based triggers*, but without full audio classification capabilities.

On the machine learning side, tools like *TensorFlow Lite for Microcontrollers* now enable real-time inference on tiny devices, allowing for simple classifiers to be deployed even on

MCUs with less than 256 kB RAM [4]. However, most literature focuses on vision-based applications or larger platforms, limiting their portability to microcontroller-scale systems.

2.2 Limitations

Despite recent advancements, several gaps remain between current state-of-the-art solutions and the needs for fully autonomous, low-power, low-cost rain detection systems:

- **Dependence on camera-based systems** limits applicability in low-cost, MCU-based solutions with constrained power budgets [7].
- **Battery-based solar systems** present issues of aging, cost, and maintenance in remote deployments [9]; very few works explore supercapacitor-based power architectures which offer longer lifecycles and robustness but require optimized energy management.
- Many sensor nodes lack **fine-grained energy-aware scheduling**; only a few works implement RTC-driven wake/sleep cycles or conditional sensing based on voltage thresholds.
- **Audio-based rain detection** remains underexplored: most works focus on visual or physical sensors, while sound-based classifiers, especially on low-power MCUs, are nearly absent [10].
- Large-scale deployments with **real-time LoRa communication** and fully off-grid operation are rare, especially when combined with embedded inference and energy harvesting [8].

3 KEY INSIGHTS

In this work we introduce a novel approach to environmental monitoring through the use of sound-based rain classification on ultra-low-power embedded hardware. Unlike traditional vision-based or sensor-based methods, our system leverages acoustic signatures of rainfall, enabling robust rain detection through a simple and lightweight machine learning classifier.

The first key insight is that audio signals—despite being rarely used in this context—can be effectively employed for rain classification. By training and deploying a sound-based model with TinyML techniques, we demonstrate that even constrained microcontrollers such as the Arduino Nano 33 BLE Sense [3] are capable of real-time audio inference without the need for cloud connectivity.

Secondly, we integrate this classifier into a fully autonomous system powered by a solar panel and a supercapacitor, completely eliminating the need for batteries. The inclusion of an RTC module allows the device to wake periodically and only perform inference when sufficient energy is available, thereby maximizing energy efficiency and ensuring long-term deployment without maintenance.

This work advances the state of the art by proving that it is possible to deploy a fully self-powered, intelligent edge device capable of environmental sensing, inference, and communication via LoRa, all within a minimal energy budget. In contrast to most prior solutions that either rely on external power or offload inference to more powerful devices, our system performs all operations locally, making it suitable for remote and infrastructure-free environments.

4 MAIN ARTIFACTS

4.1 Hardware Platform

In order to create a self-contained system, a complete hardware platform was designed. At the core of it lies a single PCB, housing the micro controller as well as all necessary peripherals. The components are chosen in order to support all required functions. Figure 1 and Figure 2 show a rendering of the populated PCB. As shown in Figure 3 the system is split into different power domains, governed by the power management unit (PMU) and the Real-time clock (RTC). This is done in order to minimize stand-by power consumption. The PMU handles power routing and conversion. At its *3V3_REG* output it provides regulated 3.3 V to the RTC, with in turn controls power with its integrated power switch (PS) to everything else through the 3V3_SW rail.

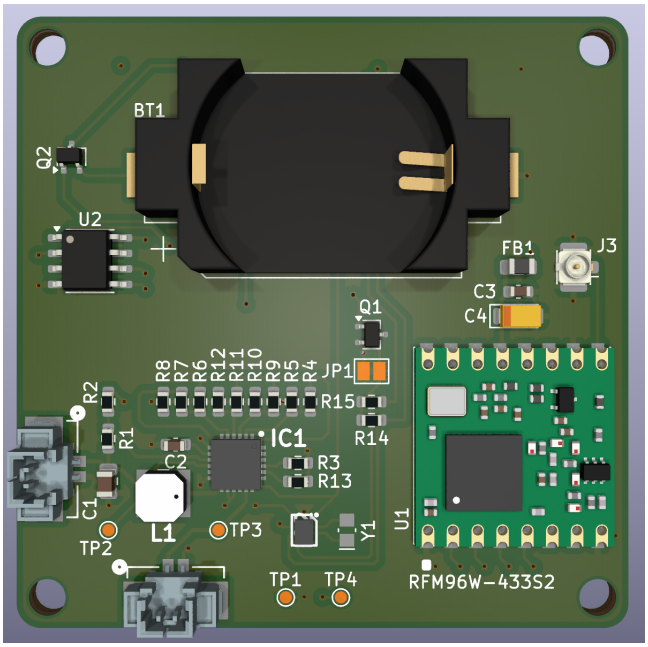

Figure 1: Front side of the PCB

This side of the PCB faces down to the ground, where the lid of the enclosure is located. By that, all connectors and the battery are accessible for easy service.

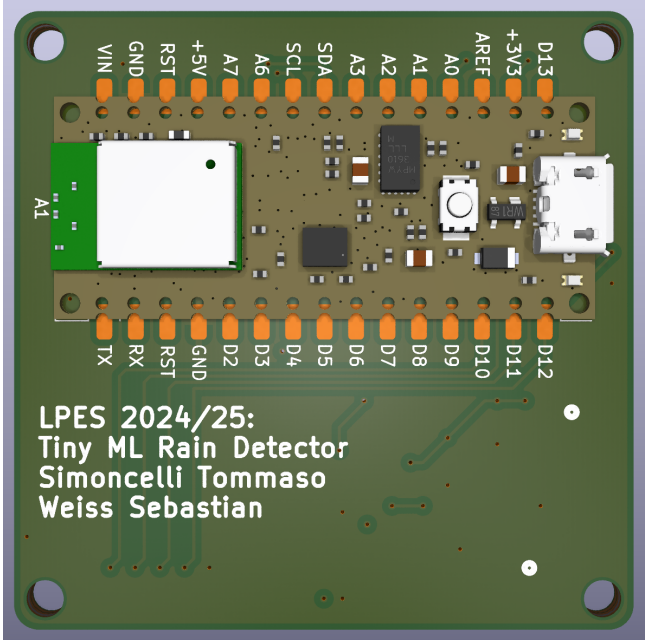

Figure 2: Back side of the PCB
This side faces towards the top of the enclosure, where the raindrops hit.

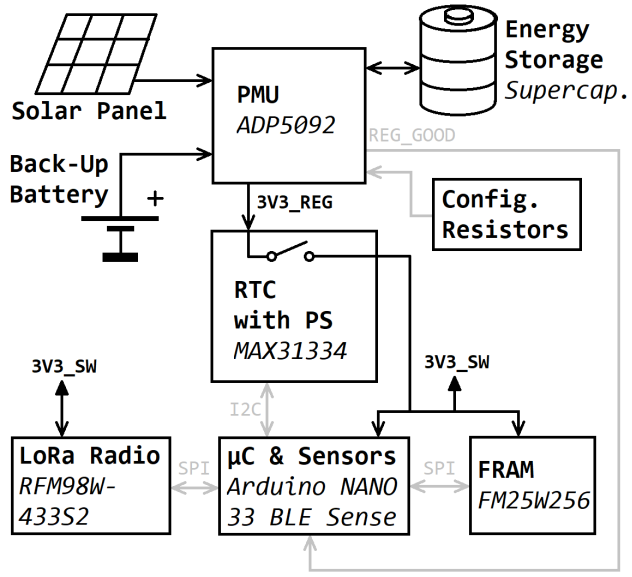

Figure 3: High level Block Diagram
Power-paths are drawn black, signal-paths are gray. Arrows indicate the flow direction of power or information respectively.

4.1.1 *Processing, Arduino NANO 33 BLE Sense:*

For the computing platform, an Arduino Nano 33 BLE Sense board is used. It features a Nordic nRF52840 with an Arm® Cortex®-M4F processor, 256 KB of RAM, and 1 MB of flash memory—sufficient to meet both the processing and memory requirements of the application. In addition, it includes all necessary sensors for rain detection (primarily the microphone) [3]. Although not the most energy-efficient option, the board was chosen due to its widespread use in Tiny Machine Learning applications and its ease of integration. This choice accelerates model development and allows the focus to shift toward optimizing the surrounding hardware components.

4.1.2 *Power Management, ADP5092:*

The ADP5092 is a popular choice for energy harvesting applications. Its extensive set of features and low power consumption [1] make it highly suitable for our use case. All important parameters, like for example maximum and minimum energy storage voltage and output voltage can be programmed by configuration resistors. The PCB includes two connectors: one for the power input, where a power source such as a solar panel can be connected, and another for energy storage. A major advantage of the ADP5092 is its broad compatibility with various energy storage devices. According to the datasheet, it supports both (super-)capacitors and a range of rechargeable battery chemistries. This flexibility led us to avoid integrating a fixed energy storage element directly onto the PCB. Instead, we routed the connection to an external port to facilitate easy prototyping and adaptability. A noteworthy feature is the integrated fixed voltage regulator along with its associated *REG_GOOD* pin. Since the peak current consumption of the entire system is not expected to exceed the regulator's 150 mA limit under any condition, all components can be powered directly from it. This approach prioritizes a more stable supply voltage at the cost of a slight reduction in efficiency. Furthermore, this design decision raises the maximum allowable voltage for the energy storage element from the 3.5 V limit—imposed by the 3.3 V components—to the ADP5092's upper limit of 5.2 V. This is particularly advantageous when using a supercapacitor, as the energy stored in a capacitor increases with the square of the voltage. If the voltage of the main energy storage becomes too low for the system to operate reliably, the PMU can be configured to automatically switch to the backup battery to maintain power supply. The state of the *REG_GOOD* pin indicates whether the system is currently powered by harvested energy or relying on the backup battery.

4.1.3 *Timekeeping and Scheduling, MAX31334:*

In order to keep track of the passed time while the system is at sleep, a real-time clock is needed. The MAX31334 is especially attractive due to its remarkable low current draw while in timekeeping mode of 70nA. In addition, the IC provides an integrated power switch for cutting power to the whole system and effectively putting it in a power-down state. For

waking up the system, alarm timestamps can be set by a micro-controller through the I^2C bus. For a more detailed description, see its datasheet [2]. Following the block design in Figure 3, the IC sits directly after the PMU and controls power to everything downstream.

4.1.4 *Connectivity, RFM96W-433S2:*

A RFM96W modem provides connectivity. It supports the LoRa protocol, which is ideally suited for reporting back system status and measurement results. It excels in terms of low power consumption and long communication range. An U.Fl connector on the PCB makes it possible to connect a variety of different antennas for the used 433 MHz ISM band.

4.1.5 *Data-Logging, FM25W256:*

For buffering and data logging purposes, a standard serial FRAM chip can be used. The SO-8 footprint and associated pinout is compatible with most common serial SPI memory devices. FRAM is preferred over EEPROM due to its significantly higher endurance in terms of write cycles. In this iteration of the circuit, the FM25W256 is selected. Thanks to the flexibility of the quasi-industry-standard footprint, many alternative variants can be used as well—for example, when a larger memory capacity is required.

4.1.6 *Energy Storage, Supercap:*

Harvested energy has to be stored somewhere. A typical choice for modern low power applications is the use of supercapacitors. These devices are generally more temperature tolerant and offer more charge/discharge cycles than the conventional rechargeable Li-Ion battery. The SCM series of supercapacitors offered by KYOCERA AVX is taken as an example. Following the datasheet [6] an expected life of 10 years can be obtained by keeping the operating temperatures under 45 °C and respecting a 80% to 90% de-rating of the operational voltage. Unbalanced types offer a lower voltage limit but show less leakage and therefore self-discharge can be minimized. A small collection of suitable 5.5 V types with varying capacity is presented in Table 1. All of them are rated for 5.5 V maximum voltage and have a diameter of 8 mm.

Table 1: Selected supercapacitors

Part number	Rated Capacitance [F]	Length [mm]	DCL Max @ 72 Hrs [µA]
SCMR14F474SRBA0	0.47	14	6
SCMR18F105SRBA0	1	18	9
SCMR22F155SRBA0	1.5	22	12

4.1.7 *Power Source, Solar:*

For a system like this, which should sense rain in semi-remote areas, it is a quite straight-forward choice to use solar as the power source. Instead of going the high efficiency, mono-crystalline route, the decision falls towards amorphous solar cells. This type of cell makes better use of scattered or indirect sunlight, such as during cloudy days. The final choice for the panel is the AM-5412CAR from Panasonic [11]. It is chosen on the basis of two criteria: first, the usable voltage range (i.e. the number of cells in series) has to be compatible with the PMU and second, the physical dimension should be compatible with the size of the system. The chosen panel has a dimention of 50 mm * 33 mm and its maximum open-circuit voltage should not exceed the 3.6 V maximum rating of the PMU. The electrical characteristics, power requirements and subsequent performance of the power source will be described in more detail in section 4.3.3.

4.1.8 *Backup power:*

A CR2032 primary cell is foreseen for backup power. It serves for supplying the RTC while no power is available and it is also connected to the backup pin of the PMU. In the case where the system should wake up while there is no harvested energy available, everything can still be powered by the battery. This is necessary to set a new wake-up alarm in the RTC. According to some rough estimates on power consumption, a single cell should last for more than 10^5 wake-up cycles.

4.2 Software: Model Development

The software component of the system was developed using the Edge Impulse platform [13]. Edge Impulse provides an end-to-end ecosystem for building, training, and deploying machine learning models on embedded devices. It was selected thanks to its integration with the Arduino Nano 33 BLE Sense, allowing for intuitive data collection, real-time visualization, and efficient model deployment. Moreover, the platform offers transparent control over each processing phase ranging from signal preprocessing to feature extraction and final model evaluation, enabling rapid interaction and optimization. Another key advantage of using Edge Impulse was the ability to test the model directly on real-world data collected from the device, before performing the actual deployment on the final hardware. This enabled validation of the model under real operating conditions and supported iterative refinement of the system to improve its robustness and performance.

4.2.1 *Training:*

To ensure real-world relevance and minimize domain mismatch, data collection was carried out using the final hardware setup enclosed in its dedicated case. This configuration reflects the actual deployment scenario, capturing audio under realistic operating conditions. A total of 13 minutes and 6 seconds of audio were recorded directly from the Arduino Nano 33 BLE Sense board, equally divided between the Rain and Idle classes.The dataset was then divided into 8 minutes and 23 seconds for training and 4 minutes and 43 seconds for testing.

4.2.2 *Preprocessing:*

In typical audio classification tasks common preprocessing techniques include the use of Mel-filterbank energy (MFE) features or spectrograms. These methods are particularly effective in speech-related tasks or when audio patterns exhibit clear frequency-based characteristics. However, in our specific scenario, detecting the sound of raindrops hitting the case, these techniques performed poorly. The reason lies in the nature of the signal: the acoustic response of a drop impact generates a broad-spectrum impulse, which is not easily isolated or enhanced through traditional frequency-domain filtering. (Fig: 4, 5)

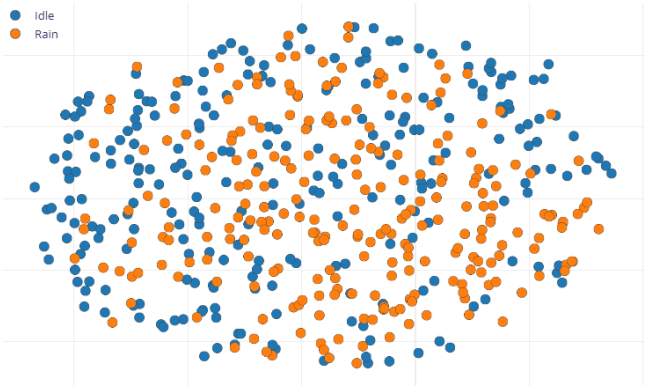

Figure 4: Features generated using the MFE.

Additionally, such preprocessing blocks generate a high number of features, increasing computational time during classification which is a major concern when working with resource constrained microcontrollers. To address both accuracy and resource limitations, we adopted a simpler and more efficient preprocessing strategy using the Flatten block in Edge Impulse. This block computes statistical summaries such as maximum value, standard deviation, and root-mean-square (RMS) over the raw audio signal. These features retain discriminative information (Fig: 6) while drastically reducing the dimensionality of the input, leading to faster inference and significantly lower memory usage (Tab: 2). Thanks to

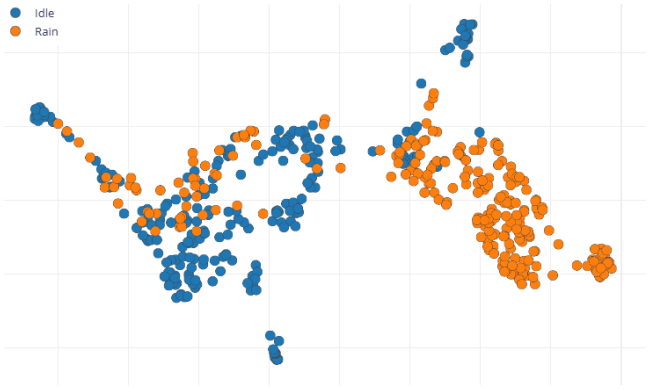

Figure 5: Features generated using the Spectrogram.

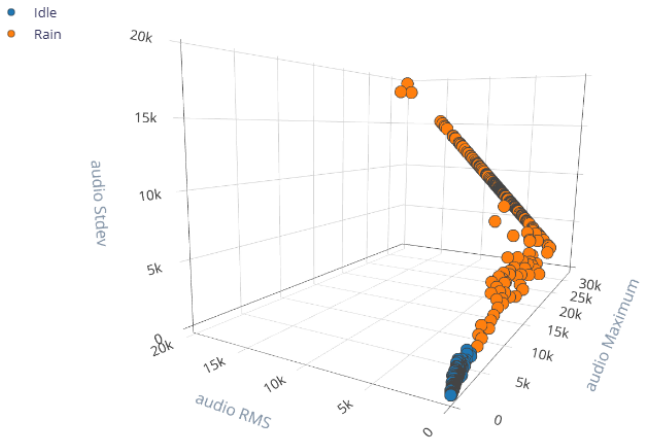

Figure 6: Features generated using the Flatten.

the adoption of the Flatten block, we were able to sample audio at 16 kHz and use a maximum listening window of 1 second enabling us to train our model on a dataset composed of 496 training samples and 283 testing samples, each corresponding to a 1-second audio window, without exceeding the memory limits of the device. This configuration would have required significant reduction in windows size if we had opted for more complex preprocessing techniques such as spectrograms or MFE. This trade-off allowed us to preserve useful acoustic features while remaining within the 256 KB SRAM limit of the Arduino Nano 33 BLE Sense.

Although the Flatten block shows a significantly lower processing time compared to more complex methods such as MFE and Spectrogram, it exhibits a higher peak RAM usage. This apparent contradiction is explained by the nature of the operations: Flatten computes simple statistics (e.g., max, standard deviation, RMS) over the entire 1-second window of raw samples, requiring the full buffer (16,000 samples at 16kHz) to be stored in memory. On the other hand, MFE

and Spectrogram rely on more computationally intensive steps (e.g., Mel-filterbanks and FFT), but they are typically implemented using smaller internal buffers and can reuse memory across processing stages, resulting in lower peak RAM usage despite longer execution times. This trade-off is summarized in Table 2.

Table 2: Comparison of DSP Blocks

DSP Block	Processing Time [ms]	Peak RAM Usage [KB]
Flatten	34	63
Spectrogram	115	28
MFE	258	15

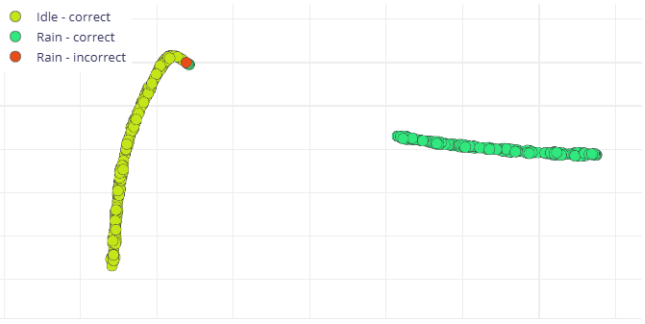

Figure 7: Data exploration after classification.

4.2.3 *Classifier:*

Thanks to the preprocessing done by the Flatten block, which produces highly distinguishable feature vectors for the two classes, the classification problem is significantly simplified. A lightweight neural network model was adopted, consisting of 2 dense layers (25 and 10 neurons respectively), trained with a learning rate of 0.005 for 2000 cycles. The model achieved excellent accuracy both on the validation set during training and on the test set. In terms of resource efficiency, the model reaches an inference time of 1ms, with a peak RAM usage of only 1.4KB and a flash usage of 15.5KB, making it highly suitable for ultra-low power, memory constrained embedded environments (Tab: 3).

Table 3: Summary of the Neural Network Classifier

Parameter	**Value**
Input Features	3 (max, stddev, RMS)
Hidden Layers	2 dense layers
Neurons per Layer	[3, 25 , 10, 2]
Output Neurons	2 (Idle, Rain)
Activation Function	ReLU (hidden), Softmax (output)
Training Epochs	2000
Learning Rate	0.005
Validation Accuracy	100%
Testing Accuracy	100%
Inference Time	1 ms
Peak RAM Usage	1.4 KB
Flash Usage	15.5 KB

4.2.4 *Software Strategy on Microcontroller:*

Given the excellent performance of the classifier in terms of both accuracy and efficiency, the implemented strategy on the Arduino device aimed to improve real-world robustness rather than classification performance itself. Specifically, a single 1-second inference may not always provide sufficient information to determine whether it is raining, especially in the presence of light or intermittent rainfall. To address this, the system performs a sequence of 11 consecutive inferences, each based on a 1-second audio recording. A majority voting mechanism is then applied to the results: the class with the highest number of votes (either *Idle* or *Rain*) is selected as the final output. This approach reduces the impact of occasional misclassifications and provides a more stable and realistic assessment of the ambient conditions. Furthermore, this voting mechanism lays the foundation for potential future improvements. By tracking the proportion of positive rain classifications within the 11 windows, the system could be extended to differentiate between light, moderate, or heavy rainfall, an enhancement that may be explored in subsequent iterations of the project.

4.3 Power Analysis

In order to sufficiently size the supercapacitor and solar panel, some estimations on power consumption have to be made. Table 4 summarizes current draw of the key components in different working states. The reported values for the Arduino are measured while the rest is taken from the respectable datasheets. Knowing the current consumption of the functional blocks permits to calculate the energy consumed for each action. The most expensive ones are displayed in Table 5. Energy consumption of other actions, such as writing a new timestamp to the RTC or logging to the FRAM chip are at least two orders of magnitude lower and therefore neglected in the following calculations.

While energy is expressed in joules, charge consumption is also provided in coulombs to simplify and more accurately represent calculations related to capacitor sizing. When the supercapacitor's voltage exceeds 3.3 V, the PMU's linear regulator activates and stabilizes the output voltage. In all cases, the *charge* consumed by the system is drawn directly from the supercapacitor, without any conversion gains or losses.

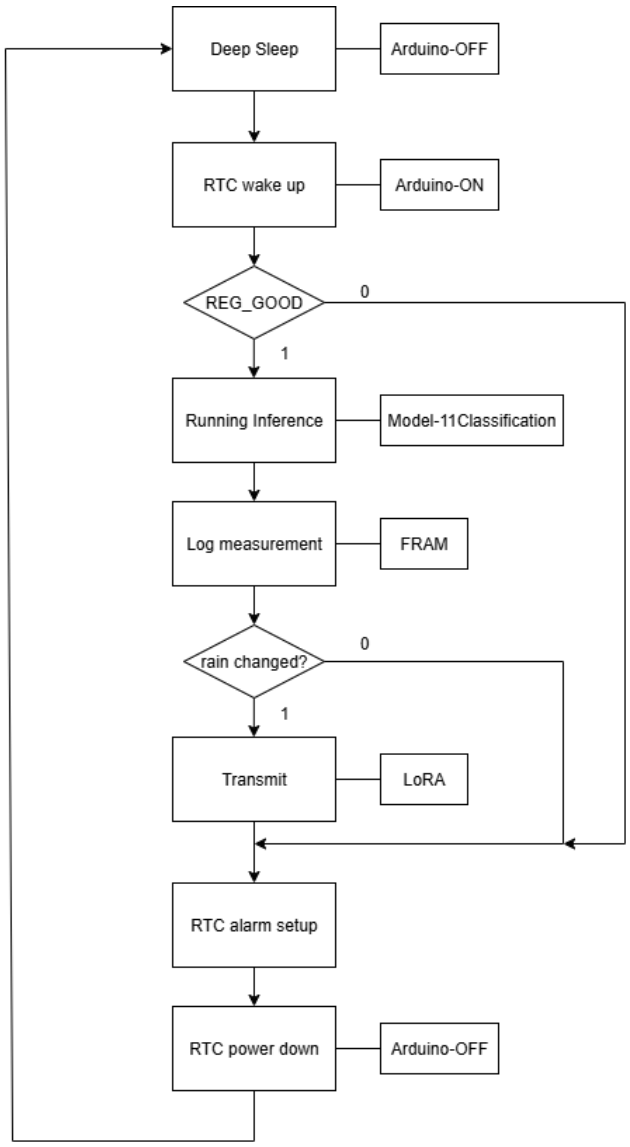

Figure 8: Flow chart.

In contrast, energy harvested from the solar panel is processed by a boost converter. As a result, the input charge does not directly correspond to the output charge. Instead, it is the input and output *power* that are approximately equal, accounting for conversion inefficiencies.

4.3.1 Energy consumption. For estimating the energy budged is to consider all different working modes of the system. Consulting the high-level flow chart in Figure 8 makes it easy to determine 4 different operational modes:

Full wake-up with measurement and transmission. This scenario describes on full cycle: As the RTC wakes up, the microcontroller reads the state of *REG_GOOD* and determines that there is enough energy avaliable. A measurement of rain is done, followed by a transmission of the event. Finally, the timestamp for the next measurement is set inside the RTC and the system goes back to sleep. Summing up one Boot of the system, one inference and one transmission, the resulting energy consumed equals 0.46 J or 138 mC if talking charge.

Table 4: Current Consumption

Component	State	Current [A]
Microcontroller with	running	6 m
Sensors *	idle	4.7 m
	RTC PS off	0
LoRa Radio *RFM98W*	transmitting	20 m
	sleep	1 µ
	RTC PS off	0
FRAM *FM25W256*	active	3 m
	idle	3 µ
	RTC PS off	0
RTC *MAX31334*	I^2C active	1.3 m
	RTC PS off	70 n
PMU *ADP5092*	operating	510 n
	sleep	390 n

*Measurements were taken using the Otii Arc Pro Power analyzer [12]

Table 5: Energy cost per action

Action	Time [s]	Total charge [mC]	Energy @ 3.3 V [J]
LoRa Transmission[1]	3.3	66	0.22
Inference	11	66	0.22
Boot	1	5.5	18 m

[1]LoRa transmissions are done with the following parameters: BW = 125 kHz; SF = 12; P = 7 dBm; Payload = 64 Byte

Partial wake-up without transmission. this case starts out in the same way as the full wake-up descibed before. The difference is that no change of state, i.e. it wasn't raining before and it isn't still raining, is detected. Therefore no transmission is scheduled. This measure reduces energy consumption by preventing redundant transmissions. For diagnostic purposes, at least once a day a transmission is forced, in which system parameters and system state is transmitted. this serves for knowing if the rain detector is working correctly or intervention is required. By omitting a LoRa transmission, needed energy is nearly cut in half in respect to a full cycle. Counting only one boot and one inference, 0.24 J or 71.5 mC are consumed.

Wake-up on back-up power without measurement. Right after wake-up, the microcontroller checks the state at the *REG_GOOD* pin of the PMU. If it reads a logic LOW, it knows that it's running from backup power because the energy harvesting storage is depleted. Therfore, a new wake-up timestamp is scheduled in the RTC and the system goes to sleep. No measurement or transmission is done. This functionality serves two main purposes: Firstly, after a waking up, a new timestamp has to be set before going to sleep, otherwise the system would stay powered down indefinitely. Secondly, by doing so, energy consumption from the back-up primary cell has to be minimized for ensuring a long life. The only significant action energy budged wise is the boot process with a charge consumption of 5.5 mC

Stand-by operation. In the absence of sunlight, the system is expected to draw a total of 460 nA while in deep sleep. In this state, only two components remain powered: the RTC, operating in timekeeping mode with a current consumption of 70 nA, and the PMU, which requires 390 nA in its sleep mode. This current is supplied by the supercapacitor. Only when its voltage drops below the backup threshold does the PMU switch to the primary cell.

4.3.2 Supercapacitor selection. As presented in table 1, three capacitors with different capacities are evaluated. Including the constrain that the usable voltage spans from 3 V to 5 V, gives values for effective energy and charge budgets in Joule and in Coulomb respectively. Table 6 summarizes the calculated values.

Table 6: Energy budget

Capacitor	Capacitance [F]	Charge budget [C]	Energy budged [J]
SCMR14F474SRBA0	0.47	0.94	3.76
SCMR18F105SRBA0	1	2	8
SCMR22F155SRBA0	1.5	3	12

Before calculating the required capacitance, some considerations regarding measurement frequency and required autonomous runtime have to be made. For all following calculations it is assumed that the system makes (or attempts) measurements at a constant interval of 30 min. This gives 48 measurements a day. Furthermore, it is assumed that at most 5 out of 48 measurements (roughly 10%) trigger a transmission. One full charge of the 1.5 F capacitor suffices for 34 measurements without transmission and 4 measurements with transmission if respecting the 90%/10% relationship. This means that one charge is easily enough for a single night and assuming the solar panel is powerful enough in order to fill the capacitor every day.

An additional thing to note is self discharge of supercapacitors. The systems stand-by current is more than an order of magnitude lower than the specified maximum direct current leakage (DCL) of the capacitors and thus can be neglected. Following the stated 12 µA for the biggest cap, the charge loss over one night is in the order of 0.5 C and therefore cuts the run time by one measurement with transmission and 5 measurements without transmission. Since the values are stated as the absolute maximum, general performance can be expected to be better.

4.3.3 Solar Panel evaluation. In the range of 1 klx to 100 klx the power provided by the solar panel can be approximated by the following linear relationship:

$$P_{solar} = 7.75 \times 10^{-7}\,\mathrm{W\,lx^{-1}} \cdot E_V \tag{1}$$

Where E_V is the radiant intensity in Lux and P is the output power of the cell at a working voltage of $V_{ope} = 2.2\,\mathrm{V}$. The coefficient $7.75 \times 10^{-7}\,\mathrm{W\,lx^{-1}}$ originates the datasheet of the panel [11] and was derived of the diagram for current vs illumination at the specified operational voltage.

The aforementioned case of 43 measurements without transmission and 5 measurements with transmission requires an average of E_{daily}12.6 J. In order to calculate the required energy from the solar panel, the average energy loss at the 3.3 V LDO has to be calculated. The working voltage of the storage capacitor ranges from 3 V to 5 V, which gives an average voltage of 4 V. This means that, on average, the LDO dissipates 0.7 V and an average estimate for conversion efficiency can be estimated

$$\eta_{LDO} = 3.3\,\mathrm{V}/4\,\mathrm{V} = 82.5\% \tag{2}$$

Adding to that the inefficiency of the converter at $\eta_{boost} = 70\%$, an approximate value for minimum daily energy production $E_{solar_{min}}$ can be calculated

$$\begin{aligned} E_{solar_{min}} &= E_{daily} \cdot \eta_{LDO}^{-1} \cdot \eta_{boost}^{-1} \\ &= 12.6\,\mathrm{J} \cdot 0.825^{-1} \cdot 0.7^{-1} \\ &= 22.2\,\mathrm{J} \end{aligned} \tag{3}$$

Now, all ingredients are gathered to finally calculate the required sun hours for sustained operation. A worst-case scenario is assumed for a cloudy, rainy winter day. By that, solar incidence is considered to be $E_V = 2\,\mathrm{klx}$. Consulting equation 1 the power produced under this condition is $P_{solar} = 1.55\,\mathrm{mW}$ and charge time for reaching the required $E_{solar_{min}} = 22.2\,\mathrm{J}$ can be calculated directly:

$$t = \frac{E_{solar_{min}}}{P_{solar}} = \frac{22.2\,\mathrm{J}}{1.55 \times 10^{-3}\,\mathrm{W} \cdot 3600\,\mathrm{s}} = 4\,\mathrm{h} \tag{4}$$

The obtained 4 h charge time is easily met, also during the shortest days in mid-European latitudes. If a the same calculations are repeated for solar irradiation of 100 klx, like during a clear, sunny day, the required energy budget for a whole day can be met within less than 3 minutes.

4.3.4 Back-up Battery life. In deep-sleep the current draw form the back-up battery equals rougly 460 nA. At these loads, the capacity of the example Lithium CR2032 [5] is 210 mA h when considering a cutoff voltage of 2.6 V. These values would equal an estimated battery life of more than 50 years. Therefore, deep-sleep consumption can be neglected and battery life in this state is solely limited by its shelf life. A more significant energy consumption arises from wake-ups with depleted harvesting storage. The example Lithium CR2032 has - in pulsed operation as in this special use case - a reduced capacity of 180 mA h at the same cutoff voltage as before. Given that 5.5 mC are consumed at each wake-up, an estimation on battery life can easily be calculated:

$$n = \frac{C_{Bat} \cdot 3600s}{Q} = \frac{0.18Ah \cdot 3600s}{5.5 \cdot 10^{-3}C} \approx 10^5 \tag{5}$$

Assuming 48 wake-ups a day, one each half an hour, 10^5 wake-ups are reached after 5 years. Following these insights, the life of one single battery reaches the expected lifetime of the whole system and can be seen as a non consumable.

5 KEY RESULTS AND CONTRIBUTIONS

Our experimental results demonstrate that it is possible to achieve reliable rain detection using a lightweight audio classifier deployed on an ultra-low-power, battery-free system. The classification accuracy is robust across multiple trials, and the system's energy consumption was kept extremely low thanks to a combination of efficient model design and hardware-level power management.

One key enabler of this result is the use of a compact machine learning model made possible by a prior feature extraction block, which reduces the dimensionality of the audio signal while preserving key information. This allowed us to deploy a simple yet effective classifier that fits well within the memory constraints of the Arduino Nano 33 BLE Sense. Furthermore, by evaluating 11 consecutive 1-second windows and applying a max-voting strategy, we improved temporal stability and reduced false positives, leading to more reliable event classification.

Additionally, we validated that a well-designed energy harvesting hardware platform—based on a solar panel, supercapacitor, and RTC can operate multiple classification cycles autonomously without the need for rechargeable batteries. This reinforces the feasibility of deploying scalable, low-maintenance environmental sensing nodes.

- We propose the first implementation of rain detection based solely on audio classification using TinyML on a batteryless, solar-powered device.
- We design and evaluate a feature extraction pipeline that enables the use of a minimal classifier, reducing memory footprint and inference latency.
- We introduce a voting mechanism over multiple inference windows to increase classification robustness against transient noise and variability in environmental conditions.
- We demonstrate a fully autonomous environmental sensing platform that combines energy harvesting, RTC-based scheduling, and edge ML to enable multiple daily inferences without the need for battery replacements.
- We provide empirical evidence that even under constrained power and memory budgets, real-time audio inference for rain detection is achievable, stable, and scalable.

Figure 9: Test setup used for evaluating the rain detection model.

6 FUTURE DIRECTIONS

While the present work addresses a binary classification task—detecting whether it is raining or not—there is potential to extend this approach to more complex problems, such as regression-based estimation of rainfall intensity. Specifically, by analyzing audio signal characteristics over time, it may be feasible to predict approximate rainfall levels in millimeters. This would offer a low-cost alternative to conventional pluviometers and weather stations, which are typically expensive and require regular maintenance.

A promising avenue would be to deploy a network of similar low-power devices arranged in a spatial matrix, allowing for the aggregation of localized rain data. Such a distributed system could generate real-time rainfall maps over large areas, enabling applications in agriculture, hydrology, and climate monitoring at a fraction of the cost of traditional infrastructure.

Moreover, to further improve energy efficiency, future iterations of the system could implement microphone-based wake-up triggers, similar to the SamurAI [10] implementation. This would allow the device to remain in deep sleep until a rain-like acoustic event is detected. However, this strategy requires careful design to avoid false activations due to unrelated noises (e.g., wind, vehicles, human activity), which could lead to premature energy depletion and reduced system reliability.

From a hardware perspective, it is clear that lower power alternatives to the current computing platform exist. The chosen Arduino board excels in rapid development and its integrated sensors enable fast and flexible prototyping. However, in a future revision of the system, selecting a more suitable microcontroller in combination with a low-power microphone could significantly reduce overall current consumption.

These enhancements would expand the applicability of our platform, reinforcing its potential as a general-purpose, energy-autonomous node for acoustic environmental sensing.

REFERENCES

[1] Analog Devices, Inc. ADP5091/ADP5092 ultralow power energy harvester pmus with mppt and charge management data sheet (rev.a). https://www.analog.com/media/en/technical-documentation/data-sheets/adp5091-5092.pdf, January 2016. Last accessed: May 16, 2025.

[2] Analog Devices, Inc. Ultra-low-power real time clock with integrated power switch data sheet. https://www.analog.com/media/en/technical-documentation/data-sheets/max31334.pdf, July 2024. Last accessed: Jun. 03, 2025.

[3] Arduino S.r.l. Arduino® nano 33 ble sense, user manual. https://docs.arduino.cc/resources/datasheets/ABX00031-datasheet.pdf, July 2025. Last accessed: Jul. 28, 2025.

[4] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, and Et Al. Tensorflow lite micro: Embedded machine learning on tinyml systems. *arXiv Preprint arXiv:2010.08678*, 2021.

[5] Energizer. Product data sheet energizer cr2032. https://data.energizer.com/pdfs/cr2032.pdf. Last accessed: Aug. 04, 2025.

[6] KYOCERA AVX. Scm series data sheet. https://web.archive.org/web/20250713075957/https://datasheets.kyocera-avx.com/AVX-SCM.pdf. Last accessed: Jul. 31, 2025.

[7] Chih-Hung G. Li, Kuei-Wen Chen, Chi-Cheng Lai, and Yu-Tang Hwang. Real-time rain detection and wiper control employing embedded deep learning. *IEEE Transactions on Vehicular Technology*, 2021.

[8] Dong Ma, Guohao Lan, Mahbub Hassan, Wen Hu, and Sajal K. Das. Sensing, computing, and communications for energy harvesting iots: A survey. *IEEE Communications Surveys and Tutorials*, 2019.

[9] Kaveh Malek, Edgardo Ortíz Rodríguez, Yi-Chen Lee, Joshua Murillo, Ali Mohammadkhorasani, and Et Al. Design and implementation of sustainable solar energy harvesting for low-cost remote sensors equipped with real-time monitoring systems. *Journal of Infrastructure Intelligence and Resilience*, 2023.

[10] Ivan Miro-Panades, Benoit Tain, Jean-Frédéric Christmann, David Coriat, Romain Lemaire, and Et Al. Samurai: A versatile iot node with event-driven wake-up and embedded ml acceleration. *IEEE Journal of Solid-State Circuits*, 2023.

[11] Panasonic Eco Solutions Amortron Co., Ltd. Am-5412car: Amrophous silicon solar cells speccification. https://www.mouser.com/catalog/specsheets/panasonic_AM-5412CAR.PDF, October 2014. Last accessed: Jun. 07, 2025.

[12] Qoitech AB. Otii arc pro by qoitech. https://www.qoitech.com/otii-arc-pro/. Last accessed: Aug. 01, 2025.

[13] Qualcomm Technologies, Inc. Edge impulse. https://edgeimpulse.com/, December 2019. Last accessed: Jul. 28, 2025.