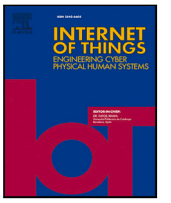
Internet of Things 25 (2024) 101109

Contents lists available at ScienceDirect

Internet of Things

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



Research article

# Machine learning-based prediction model for battery levels in IoT devices using meteorological variables

Juan Emilio Zurita Macias [a], Sergio Trilles [b,c,*]

[a] Satellite Applications Catapult, Electron Building, Fermi Ave, OX11 0QR, Didcot, Oxfordshire, United Kingdom
[b] Institute of New Imaging Technologies, Universitat Jaume I, Avd. Sos Baynat, s/n, 12071, Castelló de la Plana, Spain
[c] Facultad de Ciencias y Tecnología, Universidad Isabel I, Calle Fernán González, 76, 09003, Burgos, Spain

ARTICLE INFO

*Keywords:*
Internet of Things
Machine learning
Battery level prediction
Solar energy harvesting

ABSTRACT

Efficient energy management is vital for the sustainability of IoT devices employing solar harvesting systems, particularly to circumvent battery depletion during periods of diminished solar incidence. Embracing the structured methodology of CRISP-DM, this study introduces machine learning (ML) models that utilise meteorological data to predict battery charge levels in solar-powered IoT devices. These models enable proactive adjustments to the devices' data sampling frequencies, ensuring effective energy utilisation. The proposed ML models were evaluated using authentic battery charge data and weather forecast records. The empirical results of this study corroborate the predictive prowess of the models, with an average accuracy reaching as high as 94.09% in specific test cases. This substantiates the potential of the developed methodology to significantly enhance the energy autonomy of IoT devices through predictive analytics.

## 1. Introduction

The Internet of Things (IoT) allows real-time data collection, improving the efficiency of processes and decision-making [1]. IoT systems can collect data from everyday real-world objects, process it, derive actionable knowledge, and act on it, making it an enormous asset. By 2025, more than 16.44 billion IoT devices are expected to be connected [2], with mobile connections surpassing 30.9 billion. This has led to the emergence of next-generation applications in various domains, including smart cities [3,4], smart homes [5], healthcare [6], agriculture [7–9], smart factories, and Industry 4.0 [10].

IoT devices have evolved rapidly, enabling them to perform complex computational operations [11]. This has allowed for the generation of advanced analysis algorithms at the edge computing layer [12], which can improve network connections by reducing data latency and processing time. By processing data on-site, the need for a constant Internet connection is reduced, allowing for the utilisation of IoT technology even in areas with inadequate network infrastructure [8]. This also makes it possible to use Machine Learning (ML) analysis in the lower layers of an IoT architecture, eliminating the need to transfer data to higher layers, which significantly speeds up actions and reduces latency and computing loads that are common issues in cloud computing [13]. IoT and ML are a perfect symbiosis, with IoT providing the necessary data to feed ML models and techniques, making IoT smarter and generating decisions based on the data provided by devices. The use of ML models can be applied in a wide range of IoT domains, such as environmental monitoring [14], hydrologic monitoring [15], water quality monitoring [16], industrial applications [17] or parcel monitoring in agriculture [18].

* Corresponding author at: Institute of New Imaging Technologies, Universitat Jaume I, Avd. Sos Baynat, s/n, 12071, Castelló de la Plana, Spain.
*E-mail addresses:* juan.macias@sa.catapult.org.uk (J.E.Z. Macias), strilles@uji.es (S. Trilles).

https://doi.org/10.1016/j.iot.2024.101109
Received 3 December 2023; Received in revised form 28 January 2024; Accepted 1 February 2024
Available online 13 February 2024

The traditional approach to ML analytics involves cloud computing [19]. However, in an IoT architecture, cloud computing is located on the far end of the physical devices, which may not offer the desired Quality of Service (QoS) properties due to high latencies generated [20]. This is mainly due to the long path that large data flows must follow. These devices often need to take action based on the analysis results. In this scenario, the connection is not only the path to the cloud but also the path back to the device. This situation becomes even more complicated when the IoT application requires real-time analytical capabilities, especially when handling critical data such as emergency response, health monitoring, or smart assistants [21]. Besides these challenges, cloud-based analysis poses other problems, such as high energy consumption, privacy, and reliability issues [22].

In recent years, the emergence of edge computing has presented a solution to the issues caused by cloud computing [8]. Edge computing involves bringing data processing and storage closer to the point of data generation [23]. This approach has several benefits, such as reducing latency, improving security, and reducing bandwidth costs [24]. By processing and storing data closer to the source, edge computing reduces the amount of data that must be transmitted across the network, resulting in faster response times and better performance. Moreover, edge computing enhances security by keeping sensitive data closer to the source, reducing the risk of data breaches [25]. Lastly, edge computing reduces bandwidth costs by reducing the data transmitted over the network, resulting in lower business expenses.

Energy efficiency is a significant challenge that needs to be addressed in the IoT, as stated in [26]. Various solutions are proposed to tackle this problem, from energy-efficient hardware and software solutions to energy harvesting through renewable sources such as solar energy. Solar panel energy harvesting in wireless devices enables perpetual operation, creating a sustainable and maintenance-free IoT while eliminating battery switching, as mentioned in [27]. However, managing the energy intake and usage of energy-restrained IoT nodes can be difficult, especially when the energy harvesting is inherently unpredictable. IoT devices need accurate solar energy predictions to plan energy in volatile weather conditions. According to [28,29], these predictions must consider weather forecasts. With the availability of ML methods and computational power, it is possible to improve the accuracy of solar energy intake predictions. Although ML has been used in the renewable power sector for this purpose, as reported in [30], it has received less attention in the context of IoT nodes, which require medium-term solar predictions to plan energy.

To resolve the issue of limited energy in IoT nodes, this research proposes a methodology that leverages ML to manage and balance available energy. The potential impact of this model is considerable, particularly in terms of energy management and operational efficiency. IoT devices equipped with this model can continuously monitor and respond to environmental changes, utilising renewable energy sources such as solar power more effectively. This capability not only ensures uninterrupted operation but also significantly reduces dependence on traditional power sources. It also minimises the need for manual maintenance and recharging during periods when solar power is less available. This reduction in hands-on maintenance not only lowers direct labour costs but also extends the lifespan of the devices, leading to substantial long-term savings in operational expenses. The main contributions of the study are as follows: (a) proposing a set of ML models to predict battery levels on IoT nodes, (b) defining and developing necessary tools to retrieve weather forecasting, (c) introducing an edge computing architecture to deploy and run the ML models, and (d) real data collected from IoT nodes installed in farmers' fields has been utilised.

The paper is organised into several sections. Related works and main concepts are presented in Section 2. The methodology applied in this study, which includes detailed descriptions of the data and models used to achieve the objectives, is detailed in Section 3. Experiments and results are presented and discussed in Section 4. Finally, Section 5 summarises the main achievements and highlights future work.

## 2. Background and state of the art

### 2.1. Related concepts

There has been a lot of research interest in IoT devices in recent years due to their applications in digital agriculture, mobile health, or environmental monitoring [4,8]. However, the energy limitations and the need for frequent recharging remain obstacles to the widespread adoption of IoT devices [31]. To overcome this issue, researchers have proposed harvesting energy from ambient sources, such as light, motion, and electromagnetic waves [32,33]. Solar energy is the most efficient among these sources due to its ubiquitous presence and high energy density.

To harvest energy from ambient sources, algorithms are required that can manage the harvested energy efficiently [34,35]. These algorithms predict future energy availability to inform decisions on energy consumption. Thus, developing models that can accurately predict future energy availability is critical. Several approaches have been proposed to forecast the future availability of solar energy [34,36]. Among the various methods for developing these algorithms, notable ones include the Exponentially Weighted Moving Average (EWMA) and those employing ML techniques [37,38]

EWMA [34] uses past observations to predict the energy for a finite set of intervals. EWMA performs poorly when the weather conditions frequently change, such as when there are alternating sunny and cloudy days. To overcome this limitation, the EWMA [36] introduces a weather factor that indicates the change in weather compared to the previous days. Another commonly used algorithm is Profile-energy (Pro-energy) [39], which uses a pool of stored energy profiles to predict future energy. Pro-energy first finds the most similar profile and uses it to predict the energy.

Profile-based approaches, such as EWMA, exhibit high errors when they lack a stored profile for a specific weather condition. To address this limitation, ML-based prediction models have been proposed in recent times [37,38]. For instance, the Neural Network (NN) employed in [37] incorporates various environmental parameters, including wind speed, temperature, and pressure, to forecast future solar energy. In contrast, the study presented in [38] utilises a Reinforcement Learning-based (RL) algorithm for predicting

future energy levels amid diverse weather conditions. These ML approaches have been shown to outperform conventional models in terms of accuracy. However, they come with a high computational cost that may not be suitable for low-power IoT devices. Additionally, previous approaches usually train energy prediction models for a single location, and their accuracy in locations with different climates is not verified.

In data science and Edge Computing, a trend can be observed towards implementing continuous learning systems that allow the adaptation and constant improvement of the models used for decision-making. In this sense, the framework proposed in the article mentioned above [40] offers a complete methodology that includes from the implementation of ML models to the validation and monitoring of their performance. This approach can be particularly beneficial for predicting battery levels in edge computing models, as it allows for enhanced efficiency and accuracy in battery monitoring with the increasing amount of data collected and processed. In addition, continuous data cleaning and feature development can help identify patterns and trends that might otherwise go unnoticed.

### 2.2. Related works

In this section, the analysis focuses on research works utilising ML models to enhance the energy performance of IoT nodes, identifying several approaches akin to the proposed methodology. All selected research works depend on batteries to power IoT devices.

- In [41], the authors demonstrate a prototype that uses ML algorithms to forecast solar energy allocation for commercial sensor nodes. The k-Nearest Neighbours (k-NN) algorithm exhibits higher accuracy fluctuation than other algorithms tested.
- The research described in [42] employs a ML model that utilises the Principal Component Analysis (PCA)-based Random Forest (RF) regression algorithm to forecast the battery life of IoT devices. The accuracy of the model was enhanced by applying various pre-processing techniques, such as normalisation, transformation, and dimensionality reduction.
- In their study, Alzahrani et al. [43] explore the possibility of enhancing the efficiency of solar energy systems by applying ML techniques to environmental (historical data and weather forecasts) data for predicting future energy availability.
- Authors in [44] created a hybrid Long Short-Term Memory (LSTM) neural network-based battery prediction method to provide accurate information on the battery's state.
- [45] analyses weather forecast, charge and usage battery to see if there is any correlation between the behaviour of the nodes batteries, how the solar energy charges them, and how they use that power. After the behaviour is analysed, the goal is to see if ML can be deployed to predict the future behaviour of batteries.
- The authors of the article [46] suggest a novel interface that can transform non-energy-aware IoT devices into energy-aware ones. The interface employs ARIMA-based short-term energy forecasting and was tested using the OnePlanet sensor box. The authors' experiments showed that the proposed solution's dynamically optimised transmission rate outperformed the constant transmission rate-based solution.
- The researchers in [47] have proposed a new hierarchical ML framework capable of predicting solar energy harvest under different weather and environmental conditions. This approach allows for accurate predictions based on the time of the day while accounting for potential weather changes.
- Stricker et al. proposed a RF-based energy predictor for indoor energy harvesting systems, as described in [48]. The authors introduced an on-device online learning method to maintain high accuracy while reducing resource requirements.
- [49] presents three RL-based methods for addressing user access control and battery prediction challenges in a multiuser energy harvesting-based communication system. These methods, utilising LSTM-Deep Qlearning network (DQN) and deep LSTM algorithms, optimise scheduling, battery state prediction, and joint long-term sum rate and battery prediction.
- [50] utilises a dataset from a real-time IoT network at six beach locations to predict sensor battery life using a DNN-based model. The proposed model outperforms other Ml models by 12%, employing blockchain for backend storage to ensure tamper-proof data, presenting potential applications in fields such as supply chain management, while acknowledging scalability and transaction processing delays as unresolved challenges for future research.

In order to compare the formerly reviewed works, Table 1 has a comparison between the detailed works. The following features to characterise each one have been proposed:

- ML algorithms: used to predict battery usage.
- Dataset: with the variables used in the model.
- Metrics: used to evaluate the ML models.
- Architecture: shows if the work defines an IoT architecture to manage the proposed harvesting solution. Scale: Yes/No.
- Harvesting energy: reveals whether a system that captures ambient energy is used to harness it. Scale: Yes/No.
- Domain: implies the use cases selected where the research work is applied.

Predicting battery levels is a critical challenge in the era of mobile devices. ML provides powerful tools to address this issue. Among the most commonly used ML algorithms for battery level prediction, supervised regression techniques such as Regression Transformer (RT), Support Vector Regression (SVR), Kernel Ridge Regression (KRR), or Dynamic treatment regime (DTR) stand out after analysing all the works. Time series analysis methods, including ARIMA and RNNs, are also popular choices. Additionally, ensemble methods like RF and XGBoost can combine the strengths of multiple algorithms to improve prediction accuracy by

**Table 1**
Extracted features from selected papers.

| Work reference | ML algorithm | Dataset | Metrics | Architecture | Harvesting energy | Domain |
|---|---|---|---|---|---|---|
| [41] | KNN, SVM, ANN and RT | Energy data (battery voltage, solar charge current, ...), weather data (forecast, temperature, wind, humidity, rain, ...) and sun position (zenith and azimuth) | RMSE | ✓ | ✓ | N/A |
| [42] | LR, RF and XGBoost | Battery life, water and waves conditions | MAE, RMSE and R2 | ✗ | ✗ | Environ. |
| [43] | KNN, SVM and ANN | Historical and weather forecast, photovoltaic data, sun position and battery data | ✗ | ✗ | ✓ | Environ. |
| [44] | hybrid LSTM-PCA | Temperature and battery life | RMSE and MAE | ✗ | ✗ | Environ. |
| [45] | LR, SVR, KRR, KNN, DTR, MLP | Weather forecast, charging of the battery and battery usage | Confidence Scores | ✗ | ✗ | Environ. |
| [51] | ELM-based | Weather forecast, charging of the battery and battery usage | RMSE and MAE | ✗ | ✗ | Environ. |
| [46] | ARIMA | Solar irradiance and power sensor | RMSE | ✗ | ✓ | Environ. |
| [47] | NN | Solar irradiation and solar energy | MAE | ✗ | ✓ | Environ. |
| [48] | RF | Energy usage | MADPE | ✗ | ✓ | N/A |
| [49] | LSTM | Energy usage | Own | ✗ | ✓ | N/A |
| [50] | DNN | Water temp., turbidity, transducer depth, period and height of wave and energy usage | MAE, MSE, RMSE, TVS | ✗ | ✓ | Smart cities |
| Our | LSTM and GRU | Weather forecast and energy consump. | MAE | ✓ | ✓ | Agriculture |

accounting for complex interactions between different variables. In this kind of scenario, supervised classification algorithms such as Support Vector Machine (SVM), Linear Regression (LR), Extreme Learning Machine (ELM), and k-Nearest Neighbour (kNN) are also used to predict energy. Finally, dimensionality reduction using PCA is also applied to these types of problems. The created models' performance evaluation metrics are Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), or R squared (R2).

Most works that create models for battery level prediction use meteorological data such as temperature, wind, humidity, rain, or solar irradiation. Some studies also use meteorological prediction, as seen in [43,45,51]. In most research works, energy harvesting techniques are used, mainly solar. In addition, two works, [41,43], use the sun's position. Out of all the analysed works, only [41,46,47], and our study modify their behaviour based on the energy prediction obtained.

This research work introduces a new ML approach that accurately predicts solar energy availability. LSTM and Gated Recurrent Units (GRU) models are proposed that can adapt to seasonal and environmental changes. Experiments were conducted using real battery data and weather forecasts to demonstrate the approach's effectiveness when used in an energy management algorithm. Real datasets from the agriculture field were used; the ultimate beneficiaries of this solution are farmers, who can achieve more robust monitoring in terms of energy, resulting in direct benefits for their field practices.

## 3. Methodology

The proposed research work is focused on accurately predicting IoT devices' battery levels through ML algorithms informed by meteorological variables. The predictive model is designed to adjust the sampling rate of the nodes, and these adjustments are implemented on the IoT platform, which communicates back with the devices through the Gateway using the downlink channel [52].

In a previous project that was part of a study on smart farming, IoT nodes were developed and deployed to monitor vineyards across various locations in the Castellón province of Spain [9]. These nodes, known as SEnviro, were designed to collect environmental sensor data such as temperature, air humidity, soil moisture, atmospheric pressure, rain, and wind speed/direction. Powered by solar panels, the nodes were energy-autonomous [53]. The battery levels of the nodes were monitored and recorded. The aim was to detect vineyard diseases such as Black rot, Botrytis, Powdery mildew, or Downy mildew, based on meteorological conditions [54]. Sensor data was collected from April 1, 2018, to October 31, 2018, and the data made publicly available on the Zenodo data repository [55].

The work follows the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology proposed by Chapman [56]. This methodology is widely recognised in data science and provides a well-defined framework to ensure a systematic approach at every stage of the data mining process. The process is divided into six steps: business understanding, data understanding, data preparation, modelling, evaluation, and deployment.

A crucial initial step is defining the study's primary objective within the business context. The goal is to utilise data acquired by IoT nodes in vineyard plots, complemented with third-party services data that provides the daily number of sunlight minutes for a specific geolocation. Regular analysis and visualisation of this data are essential to ensuring correct interpretation.

Moving on to the data preparation stage, the data is treated to be ready for the modelling phase. Tasks in this stage include cleaning and processing the data for subsequent use in modelling. A key task is normalising and sequencing data from various sources, preparing it as input for the prediction model. The data are then divided into different subsets depending on the type of model to be applied. The modelling and evaluation stages involve creating models through training and subsequent metrics

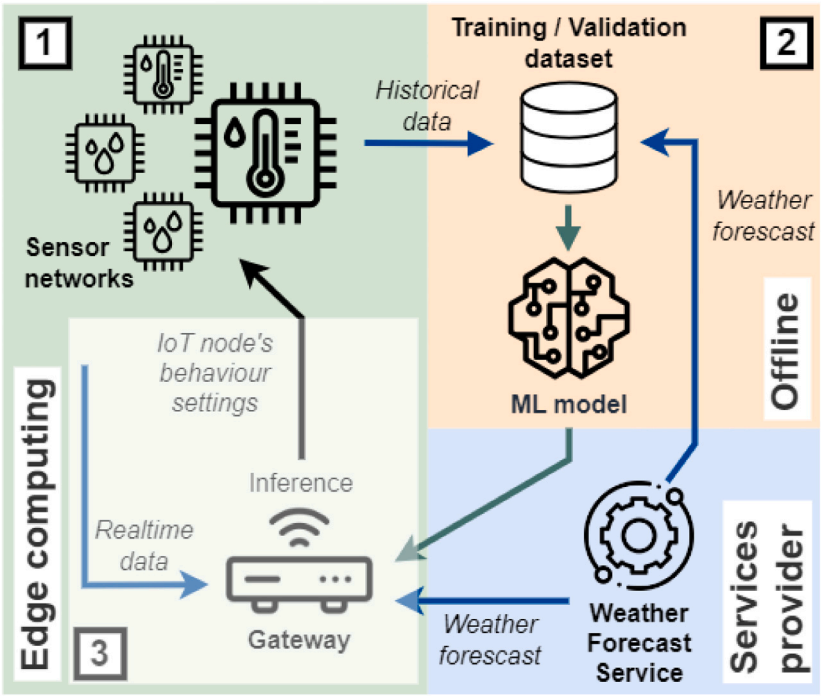



**Fig. 1.** System Design in Edge Computing.

validation. These stages are conducted offline. Finally, the deployment stage involves implementing the model on an IoT gateway for inference to obtain predictions.

Inference is performed using real-time data. After obtaining predictions from the data, it is essential to make further adjustments. This process involves selecting a suitable sampling rate based on the gathered prediction data and issuing appropriate commands to the nodes from the IoT gateway. The system is designed to be part of a node in Edge Computing, which includes the components shown in Fig. 1. Each component's functionality, as identified by its figure number (Fig. 1), is explained:

1. **Data collection/preparation** (num. 1): The system utilises two primary sources of data: IoT devices and a third-party service. These data are gathered to generate historical records that include meteorological information, such as temperature, air and soil humidity, precipitation, wind speed, barometric pressure, and battery level. Additionally, data from third-party services are used to collect information on the number of minutes of sunlight per day for a specific geolocation.
2. **Data modelling** (num. 2): During this stage, the focus is on understanding the data, preparing it for analysis and modelling, and evaluating the results. This stage is conducted offline and is iterative. The data from previous steps is analysed and integrated to form different sets, which will be used to generate ML models. Various algorithms, such as GRU and LSTM will be used to create different scenarios for predicting battery performance.
3. **Model deployment** (num. 3): This block contains an instance of a ML model that has been trained. Once implemented, the model can handle incoming data and generate predictions for battery levels. These predictions are based on data from forecast meteorological variables and historical data. The edge computing layer includes an IoT Gateway equipped with advanced computational capabilities. This device is connected to various nodes deployed in the field through low-power and wide-area networks. This network is used to receive new real-time data, perform inference, and reconfigure the nodes with different settings. Consequently, this setup allows for real-time adjustments to the sampling frequency of the IoT nodes based on the model predictions.

The current research centres on developing and evaluating ML models (num. 1 and 2) for accurately predicting IoT device battery levels based on meteorological variables. Notably, the deployment of these models on hardware infrastructure (num. 3) is intentionally excluded as it is considered less research-intensive compared to the intricate modelling processes. The decision to defer model deployment to future work underscores the primary focus on modelling intricacies, while acknowledging the importance of seamlessly integrating the models into the operational IoT framework in subsequent research and deployments.

In summary, the system is created to use meteorological variables through a ML model. This model assists in accurately predicting the battery levels of IoT devices, allowing for careful adjustment of data sampling rate, which in turn leads to better energy management. Our work focuses on stages 1 and 2, which involve generating the ML models and require in-depth research.

**Table 2**
Results of the normality test (Lilliefors test).

| Variable | Lilliefors statistic | Lilliefors $p$-value |
|---|---|---|
| Barometric Pressure | 0.026 | 0.001 |
| Battery | 0.119 | 0.001 |
| Humidity | 0.060 | 0.001 |
| Precipitation | 0.506 | 0.001 |
| Soil Humidity | 0.357 | 0.001 |
| Temperature | 0.052 | 0.001 |
| Wind Speed | 0.223 | 0.001 |
| Day Length | 0.125 | 0.001 |

**Table 3**
Example of data at the end of the preparation process.

| Barometric pressure | Battery (%) | Humidity (%) | Temperature (°C) | Wind speed (m/s) | Day length (min) | Hour |
|---|---|---|---|---|---|---|
| 987.628 | 87.636 | 49.686 | 23.712 | 2.412 | 846.0 | 15 |
| 990.870 | 86.220 | 46.395 | 21.543 | 2.410 | 846.0 | 18 |
| 991.743 | 77.947 | 55.387 | 15.701 | 0.535 | 846.0 | 21 |
| 991.772 | 69.660 | 61.112 | 12.890 | 0.160 | 848.0 | 0 |
| 991.103 | 61.120 | 65.922 | 12.018 | 0.141 | 848.0 | 3 |
| 991.265 | 51.054 | 65.933 | 13.695 | 0.267 | 848.0 | 6 |
| 991.297 | 63.023 | 38.357 | 28.445 | 2.544 | 848.0 | 9 |

### 3.1. Data preparation

The dataset used in this study was collected by Trilles et al. [57]. This dataset covers the time period from April 1, 2018, to October 31, 2018, and was gathered through a network of IoT nodes placed in different outdoor environments, including four nodes in vineyard plots. These IoT nodes continuously monitor environmental variables such as temperature, air humidity, soil moisture, atmospheric pressure, rainfall, and wind speed/direction. Opting for these variables, instead of solely focusing on solar radiation data, was a strategic decision influenced by the high costs associated with solar irradiation sensors. While these sensors provide precise measurements, their expense can be prohibitive, especially when deploying a large number of nodes. Each of the seven nodes is uniquely identified, ensuring that the data collected can be traced back to the specific node. The nodes operate on a 10-minute monitoring interval, generating a wealth of raw data, which may contain invalid or missing entries. Therefore, a robust data preparation process is necessary before engaging in modelling activities to address and mitigate any discrepancies in the dataset. The dataset has been enriched by adding a new variable called "day length". This variable represents the duration of sunlight in minutes for each day, and it has been obtained from the Sunrise Sunset API [58]. Its inclusion is crucial for increasing the model's adaptability across different geographical regions and providing valuable insights. Given the potential impact of solar radiation on the battery charge of IoT nodes, this variable is essential and offers significant information to improve the model's understanding and predictions.

To determine whether the data is normally distributed or not, the Lilliefors test [59] is performed on each variable. Table 2 shows that the $p$-values of all variables are significantly lower than the threshold value of $\alpha = 0.05$. Therefore, it is appropriate to reject the null hypothesis and conclude that the variables do not adhere to a normal distribution.

Afterwards, the analysis focuses on the relationships between variables. Nonparametric methods are particularly useful for datasets that have a large number of samples which do not follow a normal distribution. These methods are not constrained by predefined data distributions and are robust against such variations [60]. To investigate the complex interplay between variables, a correlation matrix is calculated using Spearman's correlation coefficient. This coefficient is effective at identifying both linear and non-linear relationships, without relying on the assumption of data normality. Fig. 2 presents a heatmap that visually displays the correlation results among the variables.

After completing the process of data cleaning, the next step is to sequence the historical data for model training. Sequencing involves arranging the data points in chronological order to help the model identify patterns over time. The first step is to align the data points based on their timestamps, ensuring a consistent interval between each point. Once the data points are aligned, data windows are created. These windows contain a specific number of sequential points that serve as input features for the model. These windows allow the model to consider not only the current state of the variables but also their temporal progression, which is critical for capturing temporal dynamics. After sequencing the data, the original timestamps associated with each observation are removed. However, to maintain temporal information, such as the hour of the day, this data is encoded and introduced as a categorical variable. This helps the model to retain its temporal context, which is essential for predicting patterns influenced by the time of day. By incorporating the hour of the day, the model benefits from seven distinct features during training. Table 3 provides a visual representation of the data inputs before the sequencing process.

Before delving into the architecture and functionality of the model, it is important to clarify how forecast data will be used. Once the model is trained and deployed, it uses the most recent 120 samples of cleaned historical data, including battery levels, to predict the battery level in the forthcoming 1-hour interval, relative to the latest data point. To obtain the future battery values, the model

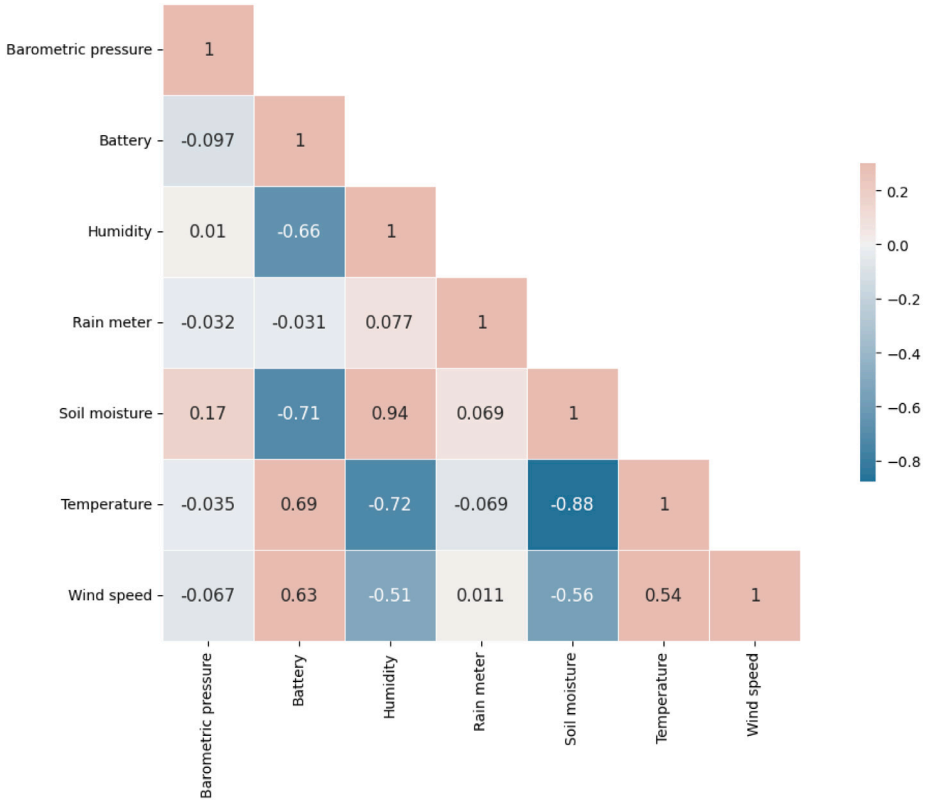



**Fig. 2.** Heatmap representation of variable correlations.

**Table 4**
Example of data sourced from OpenWeather.

| Date | Humidity (%) | Temperat. (° C) | Wind speed (m/s) | Day length (min) |
|---|---|---|---|---|
| 2023-05-30 12:00:00 | 57 | 21.10 | 4.43 | 887 |
| 2023-05-30 15:00:00 | 57 | 21.62 | 4.04 | 887 |
| 2023-05-30 18:00:00 | 61 | 20.97 | 2.35 | 887 |
| 2023-05-30 21:00:00 | 74 | 18.24 | 1.83 | 887 |
| ... | ... | ... | ... | ... |
| 2023-06-04 03:00:00 | 80 | 17.41 | 1.29 | 887 |
| 2023-06-04 06:00:00 | 79 | 17.94 | 1.44 | 887 |
| 2023-06-04 09:00:00 | 72 | 19.47 | 0.86 | 887 |

integrates forecast data from the OpenWeather API. This service provides a series of 40 predictions for weather conditions over a 5-day span, spaced at 3-hour intervals. The choice of a 120-sample window for data sequencing covers a full 1-hour interval over a 5-day cycle. This duration is specifically selected to align with the OpenWeather API's 5-day forecast. The forecasted weather data must be aligned with a 1-hour resolution, ensuring consistency with the historical data. An excerpt from the downloaded dataset, before interpolation, is shown in Table 4.

### 3.2. Data normalisation

Data normalisation is an essential preprocessing step that ensures uniformity by bringing all data to a consistent scale. Among the different normalisation methods, min–max normalisation is widely used, which transforms feature values to a standardised range of 0 to 1. The min–max scaling formula is given by:

$$X\_std = \frac{X - X.\min(axis = 0)}{X.\max(axis = 0) - X.\min(axis = 0)}$$

$$X\_scaled = X\_std \times (\max - \min) + \min$$

In this equation, the variable $X$ stands for the original feature, and $X.\min(axis = 0)$ represents the feature's minimum value, while $X.\max(axis = 0)$ represents the feature's maximum value. The parameters max and min denote the maximum and minimum

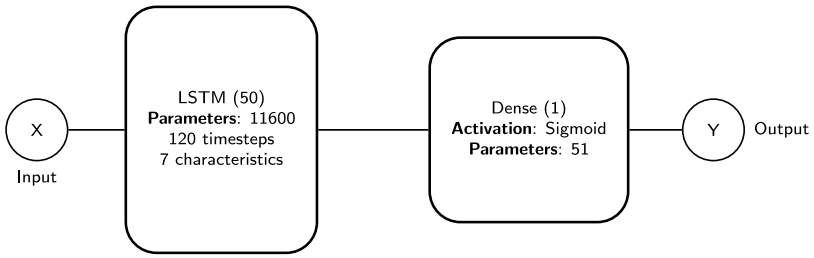



**Fig. 3.** Simple LSTM model diagram.

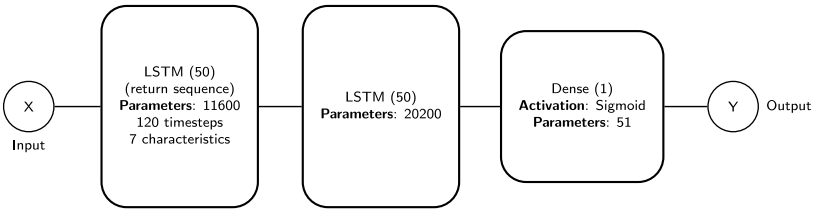



**Fig. 4.** Double LSTM Model Architecture.

values of the scaling range, which are set to default values of 1 and 0, respectively. The feature normalised to a range of 0 to 1 is denoted as $X\_std$, while the feature normalised to a specified range that is determined by min and max is represented as $X\_scaled$.

Using normalisation to transform data is a useful technique because it restricts the data to a specific range, making it less vulnerable to outliers than standardisation. This method is especially helpful when ML algorithms require input features to be within the same range, preventing any one feature from dominating the learning process.

### 3.3. Data modelling

This phase is critically important as it focuses on developing models to predict battery usage. The main goal is to enhance the data transmission rate between the node and the gateway, which is key to achieving better energy efficiency and extending battery life. The selection of meteorological variables, as shown by the correlation analysis in Fig. 2, is based on the acknowledgement that weather conditions significantly affect the battery level, our target variable. This decision is informed by the understanding that climatic factors are crucial in affecting the performance of IoT nodes, especially regarding their energy production and consumption.

In predictive modelling, selecting the right model architecture is crucial for generating precise and dependable forecasts. When all four models are trained using the same datasets, it establishes a consistent basis for evaluating them. This makes it easier to compare their strengths and weaknesses in the specific context in which they are used. Before diving into the specifics of each model's architecture, it is important to understand the overall prediction process. To make predictions, the model uses a portion of historical data that is aligned with its training sequence length of 120, taken from the test dataset. The model predicts one step into the future at a time using a multi-step forecasting approach. After each prediction, the input data is updated to reflect the latest predicted value. To start the process, the most recent 120 data points from the dataset are used as the initial input sequence. Once these data are normalised and arranged, they are fed into the model for prediction. The model's output represents the predicted battery value and is then converted back to its original scale and combined with the weather forecast data obtained from the OpenWeather API for the next prediction step. This cycle is repeated until a complete set of 120 future predictions is generated, mirroring the time span of 5 days covered by the OpenWeather forecasts.

#### 3.3.1. LSTM

The initial model employs the LSTM algorithm and comprises a layer with 50 units, followed by a dense layer. The LSTM layer processes input data of shape 120 (time steps) by seven features, as outlined in Table 3. The model concludes with a dense layer activated by a sigmoid function, ensuring output values within the range of 0 to 1. Notably, this model encompasses 11,651 trainable parameters, distributed as 11,600 in the LSTM layer and 51 in the dense layer. The architectural representation is visualised in Fig. 3.

The second model utilises a two-layer LSTM architecture. Each layer consists of 50 units and is followed by a dense layer. In this configuration, the first LSTM layer processes input data of shape 120 (time steps) with seven features, which are the same as those used in the previous model. It generates a complete sequence of outputs, which is then input to the subsequent LSTM layer. The model concludes with a dense layer activated by a sigmoid function, ensuring output values within the range of 0 to 1. Notably, this model contains 31,851 trainable parameters, with 11,600 in the first LSTM layer and 20,200 in the second (see Fig. 4).

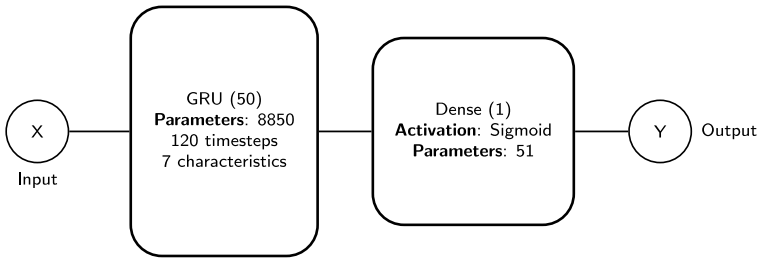



**Fig. 5.** Single Layer GRU Model Architecture.

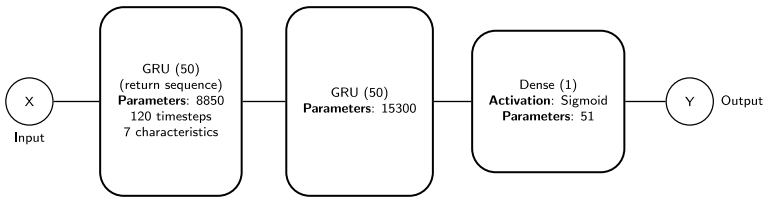



**Fig. 6.** Double Layer GRU Model Architecture.

### *3.3.2. GRU*

Two models that use the GRU architecture are described in this paper. The first model consists of a single GRU layer with 50 units, designed to handle input data of dimensions 120 by seven specified features. The architecture of the model is shown in Fig. 5 and comprises a total of 8,901 trainable parameters, with 8,850 of these in the GRU layer and the remaining 51 in the subsequent dense layer.

The second model that uses GRU has a dual-layer architecture, which includes two GRU layers. Each layer has 50 units and is followed by a dense layer. The first GRU layer produces a complete sequence of outputs. It is important to note that this model has 24,201 trainable parameters, which is a reduction from the 31,851 parameters in the double LSTM model. A visual representation of the architecture is shown in Fig. 6.

## *3.4. Evaluation metrics*

It is crucial to define key metrics before beginning model training. These metrics will serve as indicators of the model's predictive performance within the specific problem domain. For time-series models like LSTM and GRU, the focus is on assessing the forecast accuracy of the model in comparison to the actual data. To evaluate the current research problem, the Mean Absolute Error (MAE) is chosen as the metric. This metric calculates the average error and maintains the same scale as the battery values, which range from 0 to 100. MAE treats positive and negative errors equally, making it a well-suited option. Adopting MAE as the evaluation metric enables an effective comparison between different models to identify the model that exhibits the lowest MAE. A lower MAE indicates more precise forecasts of battery levels. To translate the MAE into accuracy, the percentage accuracy is calculated using the following formula:

$$\text{Accuracy } (\%) = \left(1 - \frac{\text{MAE}}{\text{Range of Battery Levels}}\right) \times 100 \tag{1}$$

In this context, the *Range of Battery Levels* refers to the difference between the maximum and minimum levels, which in the case of a scale from 0 to 100 simplifies to:

$$\text{Accuracy } (\%) = (1 - \text{MAE}/100) \times 100 \tag{2}$$

This calculation generates a percentage that reflects how close the model's predictions are to the actual values. Higher percentages indicate greater accuracy.

## 4. Results

After training the models, a test prediction is executed using the test dataset. A subset of historical data from the test set, equivalent to a training sequence length of 120, is selected to serve as the model input. The model then begins forecasting for the subsequent 120 time steps, using a multi-step prediction method. This involves predicting the next step and updating the input data with the most recently predicted value.

The process begins with the first 120 entries from the dataset, forming the input sequence to estimate the initial battery level. These data are then normalised, and sequences are constructed to feed into the model for generating predictions. After obtaining

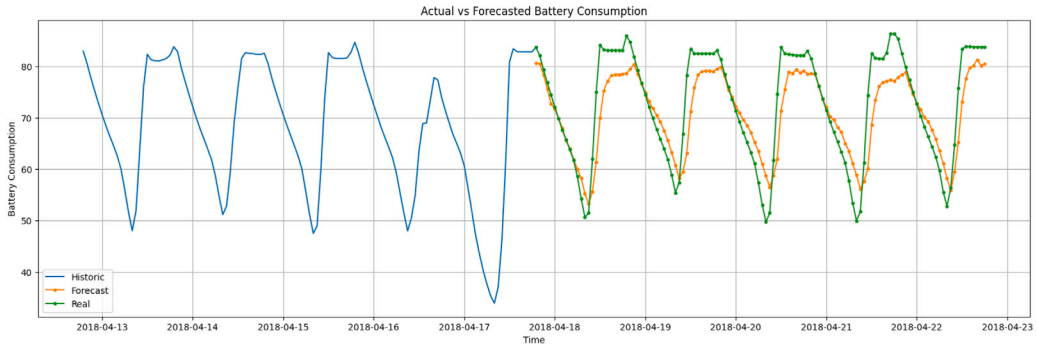



**Fig. 7.** Future value predictions using a simple LSTM.

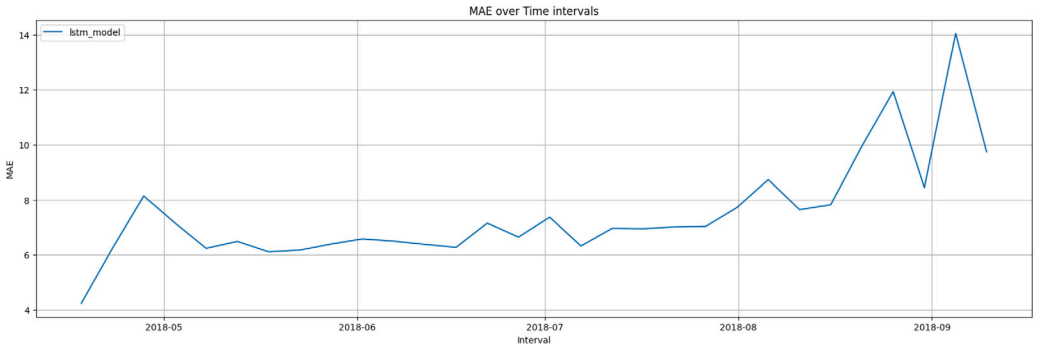



**Fig. 8.** MAE across the entire dataset using a simple LSTM.

the predicted battery value, it is converted back to its original scale. This predicted value is then integrated with upcoming weather forecast data to prepare the input for the next prediction step.

This iterative approach continues until the model completes all 120 future predictions. The final step is to visualise the predictions alongside the historical data and actual observations. Using Scenario 1 as an illustrative example (see Section 4.1.1), a graph (Fig. 7) is generated to illustrate the trends in battery level over time. The graph distinguishes between historical data (depicted in blue), model forecasts (depicted in orange), and real observations (depicted in green).

In this scenario, the MAE metric registers at 4.24, this value indicates an average error of 4.24% relative to the full range. Attaining such a metric across 120 predictions underscores noteworthy model accuracy. Nonetheless, it is crucial to acknowledge that any error persists with each subsequent prediction step, potentially magnifying the overall error as the predictions unfold. Additionally, it is important to note that this MAE value reflects performance within a specific subset of the data. A comprehensive evaluation across the entire test dataset is imperative to fully comprehend the model's effectiveness in diverse conditions and to identify any consistent error patterns that may emerge. Fig. 8 illustrates the fluctuation of this metric across different segments of the dataset.

### *4.1. Scenarios*

In order to fully evaluate the effectiveness and flexibility of our predictive model, we created a range of different scenarios by combining various datasets. Each scenario has its own specific configuration, featuring different combinations of datasets from different IoT nodes. By merging these diverse datasets, we are able to test the model's ability to adapt to different conditions, including changes in weather, location, and battery types. These scenarios represent a broad range of real-world situations, providing a solid foundation for testing the model's predictive capabilities. This approach not only enhances the model's reliability, but also gives us a better understanding of its strengths and limitations in different circumstances.

An analysis was carried out to determine how much the values of the target variable varied across datasets generated by individual nodes. The insights gained from this examination could help us understand how differences in dataset values may impact the future performance of a model trained on data that exhibits significant differences. As mentioned in Section 3.1, the Lilliefors test results suggested that the variables did not follow a normal distribution, making the assumption of normality and homogeneity of variance in the ANOVA test inappropriate. Therefore, the Kruskal–Wallis test, which is a non-parametric alternative to ANOVA designed for datasets that are not presumed to be normally distributed, is considered more appropriate. This test compares medians across groups and posits the null hypothesis that all group medians are equal, while the alternative hypothesis suggests that at least

**Table 5**
Simplified comparison of dataset pairs using the Kruskal–Wallis test.

| Dataset A | Dataset B | Statistic | *p*-value | Differences |
|---|---|---|---|---|
| Dataset 1 | Dataset 3 | 1.910 | 0.167 | Not significant |
| Dataset 0 | Dataset 6 | 5.818 | 0.016 | Significant |
| Dataset 3 | Dataset 4 | 49.083 | 0.000 | Significant |
| Dataset 0 | Dataset 2 | 85.921 | 0.000 | Significant |
| Dataset 3 | Dataset 6 | 105.278 | 0.000 | Significant |

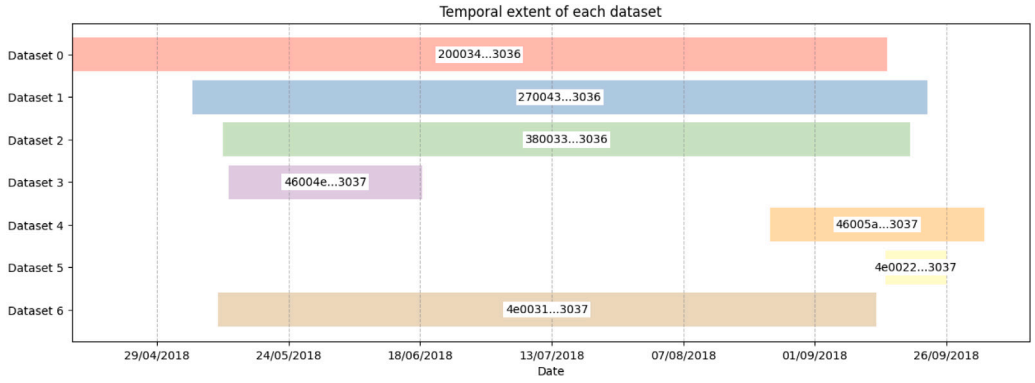



**Fig. 9.** Comparison of the temporal extent of datasets.

one group median differs. Table 5 concentrates on instances with non-significant differences. Additionally, the identifiers for each node will be replaced with dataset indices to facilitate more straightforward reference.

Comparing the time span of each dataset is crucial, as it informs decisions related to the selection of datasets for model training and testing. Fig. 9 provides a visual comparison of the temporal extents of the datasets.

Table 5 shows that Dataset 5 should potentially be excluded, primarily due to its marked discrepancies compared to others. Additionally, the apparent strong correlation between Datasets 1 and 3 prompts an initial recommendation to use Dataset 1 for training and Dataset 3 for testing purposes. It should be emphasised that this is an initial recommendation, and further confirmation is necessary during the upcoming modelling phase.

Building upon the analysis of dataset variability and the Kruskal–Wallis test results, the next step involves assessing the temporal duration of each dataset. This assessment is critical for guiding the decision-making process in the modelling phase, particularly in choosing which datasets are best suited for training and testing. The visual representation in Fig. 9 provides a comparative view of each dataset's duration, offering insights into their respective coverage periods. This analysis, coupled with the simplified dataset comparisons in Table 5, lays the groundwork for devising various modelling scenarios. These scenarios, each with a distinct approach to dataset utilisation, aim to explore and validate the model's predictive capabilities under different conditions.

The ensuing sections delve into three distinct scenarios employing the two algorithms (LSTM and GRU) outlined earlier: training with separate datasets, employing multiple datasets for training, and choosing datasets with minor variations. These scenarios are crafted to assess the model's effectiveness and precision under varied conditions.

#### *4.1.1. Scenario 1: Separate training and test datasets*

In this first scenario, the model is trained using *dataset 1*, keeping 20% of the data for validation. The testing is done using *dataset 0*. This method checks the model's capability when trained and tested on different datasets. Fig. 10 and Table 6 show the results for this model.

The results in Fig. 10 and Table 6 suggest that the model's ability to adapt to data not seen during the training phase can be gauged effectively. The low MAE scores suggest that the model, trained on Dataset 1, has demonstrated commendable performance on Dataset 0. This is a favourable indicator of the model's potential effectiveness in real-world settings where it would be exposed to diverse data variations. The best average accuracy in this scenario is achieved by the double LSTM model, with an accuracy of up to 94.09%.

#### *4.1.2. Scenario 2: Training with multiple datasets*

In the second scenario, *datasets 0 and 4* are used for training, while *dataset 1* is used for testing. The goal is to improve prediction accuracy. However, the MAE values increased for all models compared to Scenario 1. Refer to Table 6 and Fig. 11 for results.

The intention behind using multiple datasets for training in the second scenario was to expose the model to a diverse range of data, aiming to enhance its predictive accuracy. However, the increase in MAE values, as documented in Fig. 11, indicates that a greater quantity of data does not necessarily equate to improved model performance. This suggests that the added data may bring additional complexity or elements of inconsistency that challenge the model's predictive capabilities. The best average accuracy in this scenario is achieved by the double LSTM model, with an accuracy of up to 91.14%.

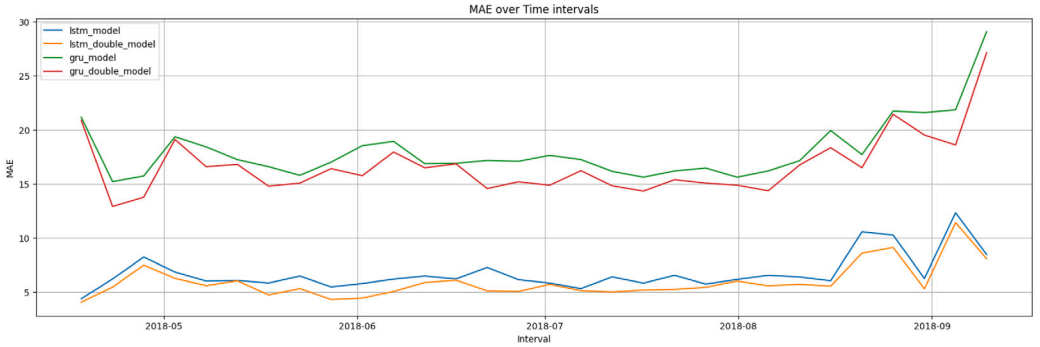



**Fig. 10.** MAE comparison between LSTM and GRU models.

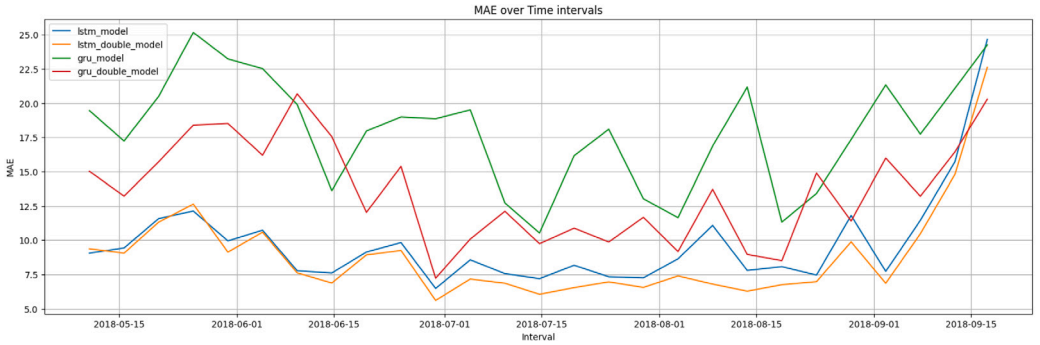



**Fig. 11.** MAE comparison using datasets 0 and 4 for training and dataset 1 for testing.

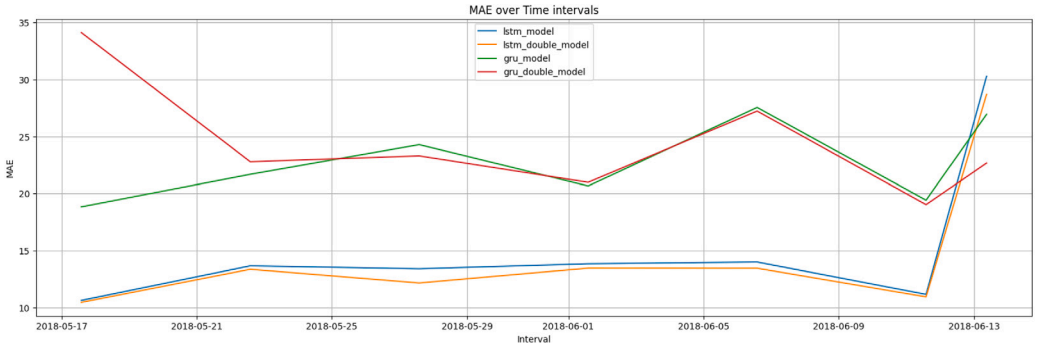



**Fig. 12.** MAE comparison using dataset 1 for training and dataset 3 for testing.

#### *4.1.3. Scenario 3: Datasets with small variations*

In this scenario, datasets with small changes in battery levels are chosen. Training is done using *dataset 1*, and *dataset 3* is used for testing. The MAE values increased across all models compared to previous scenarios. Table 6 and Fig. 12 summarise the results.

The selection of datasets with slight variations in battery levels was presumed to simplify the model's task, potentially leading to improved accuracy. Contrary to expectations, the observed increase in MAE values across all models, as reported in Table 6 and illustrated in Fig. 12. It appears that even slight changes in the battery data have a substantial impact on the model's ability to predict accurately. This reaction highlights the model's responsiveness to specific data attributes and emphasises the critical role of selecting the most suitable datasets for training and validation to ensure optimal performance. The best average accuracy in this scenario is achieved by the double LSTM model, with an accuracy of up to 85.36%.

### *4.2. Comparison*

To offer a comprehensive perspective on the performance of each model across diverse scenarios, Table 6 has been compiled. This table incorporates an efficiency index for each model, computed by dividing the average MAE by the total count of model

**Table 6**
Model comparison in different scenarios.

| Used datasets | Model | MAE min | MAE max | MAE avg (STD) | MAE/Parameters |
|---|---|---|---|---|---|
| Dataset 1 (train) Dataset 4 (test) | simple LSTM | 4.37 | 12.31 | 6.73 (1.64) | $5.77 \times 10^{-4}$ |
| | double LSTM | 4.04 | 11.41 | 5.91 (1.54) | $1.86 \times 10^{-4}$ |
| | simple GRU | 15.21 | 29.08 | 18.08 (2.78) | $2.03 \times 10^{-3}$ |
| | double GRU | 12.90 | 27.14 | 16.71 (2.79) | $6.90 \times 10^{-4}$ |
| Dataset 0/4 (train) Dataset 1 (test) | simple LSTM | 6.47 | 24.65 | 9.78 (3.56) | $8.39 \times 10^{-4}$ |
| | double LSTM | 5.60 | 22.61 | 8.86 (3.46) | $2.78 \times 10^{-4}$ |
| | simple GRU | 10.52 | 25.16 | 17.91 (3.96) | $2.01 \times 10^{-3}$ |
| | double GRU | 7.22 | 20.68 | 13.59 (3.65) | $5.61 \times 10^{-4}$ |
| Dataset 1 (train) Dataset 3 (test) | simple LSTM | 10.62 | 30.28 | 15.27(6.25) | $1.31 \times 10^{-3}$ |
| | double LSTM | 10.45 | 28.70 | 14.64 (5.85) | $4.60 \times 10^{-4}$ |
| | simple GRU | 18.82 | 27.55 | 22.76 (3.27) | $2.55 \times 10^{-3}$ |
| | double GRU | 19.02 | 34.12 | 24.30 (4.63) | $1.00 \times 10^{-3}$ |

parameters. This index functions as a gauge of performance efficiency, with lower values being desirable, indicating a reduced error rate relative to the model's complexity.

Across all scenarios, the Double LSTM model consistently outperformed other models, recording the lowest minimum, maximum, and average MAE across the board. It demonstrated superior performance, notably in terms of efficiency, as evidenced by the lowest MAE/Parameters index in the first two scenarios. While the Single LSTM model also achieved reasonable results, it fell short of the efficiency benchmark set by its Double counterpart. The GRU models, both Single and Double, trailed behind the LSTM models in performance, potentially attributed to their fewer trainable parameters. These findings suggest that LSTM architectures, and the Double LSTM model in particular, may offer greater efficiency for predictive tasks, effectively balancing error rates.

## 5. Discussion

The presented results underscore the robustness and adaptability of the ML models in predicting IoT device battery levels under various scenarios. Notably, the Double LSTM model consistently outperformed other models across different datasets, exhibiting superior accuracy and efficiency, surpassing both the simple LSTM and both GRU variants. Despite its higher parameter count of 31,851, the double LSTM model exhibits superior precision in predicting actual values as evidenced by the lowest MAE/Parameters index observed in the initial scenarios. This finding aligns with the model's architecture, which allows for capturing more intricate temporal dependencies within the data. The Single LSTM model also demonstrated commendable performance, while both GRU models trailed slightly, potentially due to their reduced parameter count.

The success of the Double LSTM model in Scenario 1, where the model is trained on one dataset and tested on another, indicates its ability to generalise well to unseen data. This adaptability is a crucial attribute for real-world applications, where IoT devices may encounter diverse environmental conditions. In Scenario 2, where training involved multiple datasets, the increase in MAE values suggests that a larger quantity of training data does not necessarily lead to improved model performance. This result prompts a deeper investigation into the complexities introduced by combining diverse datasets and their potential impact on predictive accuracy. Scenario 3, focusing on datasets with small variations, revealed unexpected challenges. The models struggled to maintain accuracy even when presented with subtle changes in battery levels. This sensitivity highlights the importance of careful dataset curation and the need for models that can discern meaningful patterns amid nuanced variations.

The efficiency index, calculated by dividing the average MAE by the total count of model parameters, provides insights into the trade-off between model complexity and performance. The lower values for the Double LSTM model indicate a more efficient use of parameters in reducing prediction errors. This suggests that, in certain scenarios, a more complex model architecture may indeed be beneficial for enhancing predictive capabilities.

### 5.1. Limitations

One notable limitation lies in the variability across datasets, as evidenced by the Kruskal–Wallis test results. While efforts were made to select datasets that represent distinct conditions, the inherent diversity among IoT nodes may introduce challenges in achieving a uniform model performance. Addressing this limitation may require further investigation into the impact of dataset heterogeneity on predictive accuracy.

While the models showcased strong performance within the defined scenarios, the translation of these findings to real-world applications warrants careful consideration. Factors such as network connectivity, hardware constraints, and unforeseen environmental anomalies could influence model efficacy in practical settings. Future work should involve field testing to validate the models under authentic IoT deployment conditions.

The models' performance is contingent on hyperparameter tuning, and the presented results are based on optimal configurations determined during experimentation. Sensitivity to hyperparameter changes may influence the models' generalisation capabilities. A more exhaustive exploration of hyperparameter space could unveil additional nuances in model behaviour.

## 6. Conclusion

In addressing the challenge of constrained energy resources in IoT nodes, this study introduced an innovative edge computing architecture empowered by ML for the purpose of effective energy management. The potential ramifications of this proposed model are extensive, particularly with regards to energy optimisation and operational efficiency. IoT devices integrated with this advanced model exhibit continuous monitoring and adaptive responses to environmental dynamics, thereby enhancing the utilisation of renewable energy sources, such as solar power. This not only ensures uninterrupted device functionality but also substantially diminishes reliance on conventional power grids. Moreover, the model's capacity to minimise manual maintenance and recharging requirements during periods of limited solar power availability contributes to significant operational benefits. The resultant reduction in hands-on interventions not only lowers direct labour costs but also prolongs the operational lifespan of the devices, translating into substantial long-term savings.

The proposed predictive models, particularly the Double LSTM architecture, showcase promising results in forecasting battery levels across diverse scenarios. The study highlights the importance of careful dataset selection, considering variations, and the potential limitations associated with training on multiple datasets. The efficiency index, incorporating model complexity, indicates that the Double LSTM strikes an effective balance between accuracy and efficiency. The model achieves an outstanding accuracy rate of up to 94.09% in the initial scenario, making it ideal for efficient energy resource management and guarding against operational disruptions during periods of reduced solar activity.

The model's predictive capability is due to its integration of meteorological data from the OpenWeather API, which can be used to adjust the sampling rates of IoT nodes. To ensure adaptability to diverse daylight hour fluctuations, the model also considers geographical location and seasonal variations. This approach enhances the model's ability to generalise across different times and locations. Designed for deployment on an Edge Computing platform, the model is ideal for implementation in environments utilising star topology networks with gateway technologies for IoT communication.

Exploration into distributed deployment directly on nodes for real-time data processing and immediate response to battery level predictions is suggested for future work. This approach allows for a comparative analysis between centralised and decentralised models, shedding light on efficiency, scalability, and potential trade-offs in prediction accuracy due to edge device computational constraints. Additionally, a distributed model could pave the way for federated learning systems, promoting collaborative learning among nodes to enhance prediction accuracy and generalisation without compromising data privacy and security.

Considering the model's potential evolution, incorporating additional meteorological variables such as solar radiation or cloud cover could enhance prediction accuracy. An experiment investigating the impact of IoT device location on model accuracy is also proposed, recognising that different regions may exhibit distinct weather patterns.

## CRediT authorship contribution statement

**Juan Emilio Zurita Macias:** Writing – original draft, Validation, Software, Data curation. **Sergio Trilles:** Writing – review & editing, Supervision, Methodology, Investigation, Funding acquisition, Conceptualization.

## Declaration of competing interest

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

## Data availability

Datasets are published in Zenodo.

## Acknowledgments

Grant PID2022-141813OB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF, a way of making Europe", by the European Union - Next GenerationEU/ PRTR.

## References

[1] C. Granell, A. Kamilaris, A. Kotsev, F.O. Ostermann, S. Trilles, Internet of things, Man. Digit. Earth (2020) 387–423.
[2] IHS Statista, Internet of things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions), 2018, URL: https://www.statista.com/statistics/471264/iot-number-of-connecteddevicesworldwide/(Consult\'e17/05/2020).
[3] H. Arasteh, V. Hosseinnezhad, V. Loia, A. Tommasetti, O. Troisi, M. Shafie-khah, P. Siano, IoT-based smart cities: A survey, in: 2016 IEEE 16th International Conference on Environment and Electrical Engineering, EEEIC, IEEE, 2016, pp. 1–6.
[4] S. Trilles, A. Calia, Ó. Belmonte, J. Torres-Sospedra, R. Montoliu, J. Huerta, Deployment of an open sensorized platform in a smart city context, Future Gener. Comput. Syst. 76 (2017) 221–233.
[5] M. Wang, G. Zhang, C. Zhang, J. Zhang, C. Li, An IoT-based appliance control system for smart homes, in: 2013 Fourth International Conference on Intelligent Control and Information Processing, ICICIP, IEEE, 2013, pp. 744–747.
[6] M.H. Kashani, M. Madanipour, M. Nikravan, P. Asghari, E. Mahdipour, A systematic review of IoT in healthcare: Applications, techniques, and trends, J. Netw. Comput. Appl. 192 (2021) 103164.

[7] M. Abbasi, M.H. Yaghmaee, F. Rahnama, Internet of things in agriculture: A survey, in: 2019 3rd International Conference on Internet of Things and Applications, IoT, IEEE, 2019, pp. 1–12.
[8] S. Trilles, J. Torres-Sospedra, Ó. Belmonte, F.J. Zarazaga-Soria, A. González-Pérez, J. Huerta, Development of an open sensorized platform in a smart agriculture context: A vineyard support system for monitoring mildew disease, Sustain. Comput. Inform. Syst. 28 (2020) 100309.
[9] S. Trilles, A. González-Pérez, J. Huerta, An IoT platform based on microservices and serverless paradigms for smart farming purposes, Sensors 20 (8) (2020) 2418.
[10] P.K. Malik, R. Sharma, R. Singh, A. Gehlot, S.C. Satapathy, W.S. Alnumay, D. Pelusi, U. Ghosh, J. Nayak, Industrial Internet of Things and its applications in industry 4.0: State of the art, Comput. Commun. 166 (2021) 125–139.
[11] H. El-Sayed, S. Sankar, M. Prasad, D. Puthal, A. Gupta, M. Mohanty, C.-T. Lin, Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment, IEEE Access 6 (2017) 1706–1717.
[12] W. Shi, S. Dustdar, The promise of edge computing, Computer 49 (5) (2016) 78–81.
[13] K. Bajaj, B. Sharma, R. Singh, Implementation analysis of IoT-based offloading frameworks on cloud/edge computing for sensor generated big data, Complex Intell. Syst. 8 (5) (2022) 3641–3658.
[14] M. Kashid, K. Karande, A. Mulani, IoT-based environmental parameter monitoring using machine learning approach, in: Proceedings of the International Conference on Cognitive and Intelligent Computing: ICCIC 2021, Vol. 1, Springer, 2022, pp. 43–51.
[15] P.P. Ariza-Colpas, C.E. Ayala-Mantilla, Q. Shaheen, M.A. Piñeres-Melo, D.A. Villate-Daza, R.C. Morales-Ortega, E. De-la Hoz-Franco, H. Sanchez-Moreno, B.S. Aziz, M. Afzal, SISME, estuarine monitoring system based on IoT and machine learning for the detection of salt wedge in aquifers: case study of the Magdalena River estuary, Sensors 21 (7) (2021) 2374.
[16] N.K. Koditala, P.S. Pandey, Water quality monitoring system using IoT and machine learning, in: 2018 International Conference on Research in Intelligent and Computing in Engineering, RICE, IEEE, 2018, pp. 1–5.
[17] A. Kanawaday, A. Sane, Machine learning for predictive maintenance of industrial machines using IoT sensor data, in: 2017 8th IEEE International Conference on Software Engineering and Service Science, ICSESS, IEEE, 2017, pp. 87–90.
[18] R. Akhter, S.A. Sofi, Precision agriculture using IoT data analytics and machine learning, J. King Saud Univ.-Comput. Inf. Sci. 34 (8) (2022) 5602–5618.
[19] J. Gao, H. Wang, H. Shen, Machine learning based workload prediction in cloud computing, in: 2020 29th International Conference on Computer Communications and Networks, ICCCN, IEEE, 2020, pp. 1–9.
[20] L. Cui, S. Yang, F. Chen, Z. Ming, N. Lu, J. Qin, A survey on application of machine learning for Internet of Things, Int. J. Mach. Learn. Cybern. 9 (2018) 1399–1417.
[21] L.S. Kondaka, M. Thenmozhi, K. Vijayakumar, R. Kohli, An intensive healthcare monitoring paradigm by using IoT based machine learning strategies, Multimedia Tools Appl. 81 (26) (2022) 36891–36905.
[22] R.P. Padhy, M.R. Patra, S.C. Satapathy, Cloud computing: security issues and research challenges, Int. J. Comput. Sci. Inf. Technol. Secur. (IJCSITS) 1 (2) (2011) 136–146.
[23] Ó. Belmonte-Fernández, E. Sansano-Sansano, S. Trilles, A. Caballer-Miedes, A reactive architectural proposal for fog/edge computing in the internet of things paradigm with application in deep learning, in: Artificial Intelligence, Machine Learning, and Optimization Tools for Smart Cities: Designing for Sustainability, Springer, 2022, pp. 155–175.
[24] W.Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, A. Ahmed, Edge computing: A survey, Future Gener. Comput. Syst. 97 (2019) 219–235.
[25] L. Kong, J. Tan, J. Huang, G. Chen, S. Wang, X. Jin, P. Zeng, M. Khan, S.K. Das, Edge-computing-driven Internet of Things: A survey, ACM Comput. Surv. 55 (8) (2022) 1–41.
[26] H. Jayakumar, A. Raha, Y. Kim, S. Sutar, W.S. Lee, V. Raghunathan, Energy-efficient system design for IoT devices, in: 2016 21st Asia and South Pacific Design Automation Conference, ASP-DAC, IEEE, 2016, pp. 298–301.
[27] F.K. Shaikh, S. Zeadally, Energy harvesting in wireless sensor networks: A comprehensive review, Renew. Sustain. Energy Rev. 55 (2016) 1041–1054.
[28] N. Sharma, J. Gummeson, D. Irwin, P. Shenoy, Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems, in: 2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON, IEEE, 2010, pp. 1–9.
[29] C. Renner, S. Unterschütz, V. Turau, K. Römer, Perpetual data collection with energy-harvesting sensor networks, ACM Trans. Sensor Netw. 11 (1) (2014) 1–45.
[30] M. Sharifzadeh, A. Sikinioti-Lock, N. Shah, Machine-learning methods for integrated renewable power generation: A comparative study of artificial neural networks, support vector regression, and Gaussian Process Regression, Renew. Sustain. Energy Rev. 108 (2019) 513–538.
[31] Y.-K. Chen, Challenges and opportunities of internet of things, in: 17th Asia and South Pacific Design Automation Conference, IEEE, 2012, pp. 383–388.
[32] M. Geisler, S. Boisseau, M. Perez, P. Gasnier, J. Willemin, I. Ait-Ali, S. Perraud, Human-motion energy harvester for autonomous body area sensors, Smart Mater. Struct. 26 (3) (2017) 035028.
[33] A. Valenzuela, Energy harvesting for no-power embedded systems, Tex. Instrum. Oct. 28 (2008).
[34] A. Kansal, J. Hsu, S. Zahedi, M.B. Srivastava, Power management in energy harvesting sensor networks, ACM Trans. Embed. Comput. Syst. (TECS) 6 (4) (2007) 32–es.
[35] C.M. Vigorito, D. Ganesan, A.G. Barto, Adaptive control of duty cycling in energy-harvesting wireless sensor networks, in: 2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, IEEE, 2007, pp. 21–30.
[36] J.R. Piorno, C. Bergonzini, D. Atienza, T.S. Rosing, Prediction and management in energy harvested wireless sensor nodes, in: 2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, IEEE, 2009, pp. 6–10.
[37] S. Dhillon, C. Madhu, D. Kaur, S. Singh, A solar energy forecast model using neural networks: Application for prediction of power for wireless sensor networks in precision agriculture, Wirel. Pers. Commun. 112 (2020) 2741–2760.
[38] S. Kosunalp, A new energy prediction algorithm for energy-harvesting wireless sensor networks with Q-learning, IEEE Access 4 (2016) 5755–5763.
[39] A. Cammarano, C. Petrioli, D. Spenza, Pro-energy: A novel energy prediction model for solar and wind energy-harvesting wireless sensor networks, in: 2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems, MASS 2012, IEEE, 2012, pp. 75–83.
[40] P.P. Hanzelik, A. Kummer, J. Abonyi, Edge-computing and machine-learning-based framework for software sensor development, Sensors 22 (11) (2022) http://dx.doi.org/10.3390/s22114268, URL https://www.mdpi.com/1424-8220/22/11/4268.
[41] F.A. Kraemer, D. Ammar, A.E. Braten, N. Tamkittikhun, D. Palma, Solar energy prediction for constrained IoT nodes based on public weather forecasts, in: Proceedings of the Seventh International Conference on the Internet of Things, 2017, pp. 1–8.
[42] P.K. Reddy Maddikunta, G. Srivastava, T. Reddy Gadekallu, N. Deepa, P. Boopathy, Predictive model for battery life in IoT networks, IET Intell. Transp. Syst. 14 (11) (2020) 1388–1395.
[43] M. Alzahrani, A.S. Weddell, W. Gary, Using environmental data for IoT device energy harvesting prediction, 2022.
[44] T. Siva, A. Beno, B. Lanitha, V. Yogalakshmi, M. Manikandan, S.S. Kumar, V. Peroumal, A. Darwin Nesakumar, V.R.R. Prasad, Hybrid LSTM-PCA-powered renewable energy-based battery life prediction and management for IoT applications, J. Nanomater. 2022 (2022).
[45] K.B.L. Fjærestad, Analyzing the Correlation in the Behavior of Batteries in IoT Nodes, Powered by Solar Energy (Master's thesis), NTNU, 2018.
[46] A.J. Rajappa, A. Sabovic, B. Celikkol, M. Aernouts, P. Reiter, S. Mercelis, P. Hellinckx, J. Famaey, An energy management unit for predictive solar energy harvesting IoT, 2023.

[47] N. Yamin, G. Bhat, Online solar energy prediction for energy-harvesting internet of things devices, in: 2021 IEEE/ACM International Symposium on Low Power Electronics and Design, ISLPED, IEEE, 2021, pp. 1–6.
[48] N. Stricker, L. Thiele, Accurate onboard predictions for indoor energy harvesting using random forests, in: 2022 11th Mediterranean Conference on Embedded Computing, MECO, IEEE, 2022, pp. 1–6.
[49] M. Chu, H. Li, X. Liao, S. Cui, Reinforcement learning-based multiaccess control and battery prediction with energy harvesting in IoT systems, IEEE Internet Things J. 6 (2) (2018) 2009–2020.
[50] S.R.K. Somayaji, M. Alazab, M. Manoj, A. Bucchiarone, C.L. Chowdhary, T.R. Gadekallu, A framework for prediction and storage of battery life in IoT devices using DNN and blockchain, in: 2020 IEEE Globecom Workshops, GC Wkshps, IEEE, 2020, pp. 1–6.
[51] A. Sinha, D. Das, V. Udutalapally, S.P. Mohanty, Ithing: Designing next-generation things with battery health self-monitoring capabilities for sustainable IIoT, IEEE Trans. Instrum. Meas. 71 (2022) 1–9.
[52] LoRa Alliance, LoRaWAN specification v1.0.3, 2015, URL https://lora-alliance.org/resource_hub/lorawan-specification-v1-0-3/.
[53] S. Trilles, A. González-Pérez, J. Huerta, A comprehensive IoT node proposal using open hardware. A smart farming use case to monitor vineyards, Electronics 7 (12) (2018) 419.
[54] S. Trilles Oliver, A. González-Pérez, J. Huerta Guijarro, Adapting models to warn fungal diseases in vineyards using in-field Internet of Things (IoT) nodes, Sustainability 11 (2) (2019) 416.
[55] S. Trilles, A. González-Pérez, B. Zaragozí, J. Huerta, Data on records of environmental phenomena using low-cost sensors in vineyard smallholdings, Data Brief 33 (2020) 106524.
[56] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth, The CRISP-DM user guide, in: 4th CRISP-DM SIG Workshop in Brussels in March, Vol. 1999, sn, 1999.
[57] S. Trilles, Enviromental sensor data collected using low-cost IoT nodes (SEnviro) from vineyard smallholdings (Season 2018), 2020, http://dx.doi.org/10.5281/zenodo.3727310.
[58] Sunrise sunset API, 2023, https://sunrise-sunset.org/api. (Accedido: 20 de mayo de 2023).
[59] H.W. Lilliefors, On the Kolmogorov-Smirnov test for normality with mean and variance unknown, J. Amer. Statist. Assoc. 62 (318) (1967) 399–402, http://dx.doi.org/10.1080/01621459.1967.10482916.
[60] D.G. Bonett, T.A. Wright, Sample size requirements for estimating pearson, Kendall and spearman correlations, Psychometrika 65 (1) (2000) 23–28, http://dx.doi.org/10.1007/BF02294183.