big data and cognitive computing

Article

A Guide to Data Collection for Computation and Monitoring of Node Energy Consumption

Alberto del Rio [1,*], Giuseppe Conti [1], Sandra Castano-Solis [2], Javier Serrano [1], David Jimenez [1,3,*] and Jesus Fraile-Ardanuy [3]

[1] GATV Research Group, Signals, Systems and Radiocommunications Department, Universidad Politécnica de Madrid, 28040 Madrid, Spain; gic@gatv.ssr.upm.es (G.C.); jsr@gatv.ssr.upm.es (J.S.)

[2] Escuela Técnica Superior de Ingeniería y Diseño Industrial (ETSIDI), Universidad Politécnica de Madrid, 28040 Madrid, Spain; sp.castano@upm.es

[3] Information Processing and Telecommunications Center (IP&T Center), Universidad Politécnica de Madrid, 28040 Madrid, Spain; jesus.fraile.ardanuy@upm.es

* Correspondence: arp@gatv.ssr.upm.es (A.d.R.); djb@gatv.ssr.upm.es (D.J.)

Abstract: The digital transition that drives the new industrial revolution is largely driven by the application of intelligence and data. This boost leads to an increase in energy consumption, much of it associated with computing in data centers. This fact clashes with the growing need to save and improve energy efficiency and requires a more optimized use of resources. The deployment of new services in edge and cloud computing, virtualization, and software-defined networks requires a better understanding of consumption patterns aimed at more efficient and sustainable models and a reduction in carbon footprints. These patterns are suitable to be exploited by machine, deep, and reinforced learning techniques in pursuit of energy consumption optimization, which can ideally improve the energy efficiency of data centers and big computing servers providing these kinds of services. For the application of these techniques, it is essential to investigate data collection processes to create initial information points. Datasets also need to be created to analyze how to diagnose systems and sort out new ways of optimization. This work describes a data collection methodology used to create datasets that collect consumption data from a real-world work environment dedicated to data centers, server farms, or similar architectures. Specifically, it covers the entire process of energy stimuli generation, data extraction, and data preprocessing. The evaluation and reproduction of this method is offered to the scientific community through an online repository created for this work, which hosts all the code available for its download.

Keywords: dataset; energy dataset; energy measurement; IoT; resource utilization; stress tests; energy features; system power; energy behavior

check for updates

Citation: del Rio, A.; Conti, G.; Castano-Solis, S.; Serrano, J.; Jimenez, D.; Fraile-Ardanuy, J. A Guide to Data Collection for Computation and Monitoring of Node Energy Consumption. *Big Data Cogn. Comput.* **2023**, 7, 130. https://doi.org/10.3390/bdcc7030130

Academic Editors: Anupam Kumar Bairagi, Md. Golam Rabiul Alam, Anselme Ndikumana and Md. Shirajum Munir

Received: 30 May 2023
Revised: 26 June 2023
Accepted: 5 July 2023
Published: 11 July 2023

1. Introduction

New technologies and their multidimensional applications using the Internet and cloud servers are growing steadily [1]. Social networks, mail services, video streaming, and games, all kinds of applications that use data centers, have one common factor: the need for electric power. In fact, the constant growth of these applications means that a larger number of data centers is required. As a consequence, more energy is used to operate these data centers, demanding new techniques to reduce their electric consumption.

The application of artificial intelligence (AI) to server management can lead to increased energy efficiency. By analyzing the intrinsic information patterns in service usage data, it is possible to anticipate both the behavior of individual users and the general stress of the servers, allowing for optimization of the charge over them. These improved efficiencies are achieved thanks to AI techniques but require a set of training data for the implementation of these tools. This represents the condition for the application of machine learning (ML) techniques to a previously organized dataset.

There are many open repositories to find suitable datasets for different areas and tasks [2]. The increase in demand for services and application development drives growth for a larger community and expands the dataset market [3]. Sometimes, even the data are provided openly in order to support new initiatives for developers to create services for specific tasks. It is not always intended for commercial purposes, but to create competition to drive research even further. This is beneficial for a whole stack of sectors: students starting to learn about AI, developers expanding their knowledge and driving research, and private organizations that would benefit from the creativity of the created solutions.

However, energy datasets are not open and freely shared [4] with the community. Security issues are the main concern, as any uncontrolled access to data centers represents a risk for companies because they store private data of great value. Furthermore, energy consumption patterns can provide information about the services and applications that are held in a data center. This leads to the unavailability of energy datasets, which prevents the development of ML algorithms for energy efficiency optimization.

Additionally, the requirement for data acquisition, from a technical point of view, involves the coordination about a set of technologies and qualified staff, i.e., physical layer availability (infrastructure, hardware, sensors, etc.); software layer development, including process monitoring and flexible and adaptable configuration (software developer); interconnection between operating systems and data management services (DevOps); and data processing and analysis (data scientists).

Most of the procedures proposed to date are focused on a specific applications; however, the generation pf energy consumption data could represent a scalable solution. The method presented in this work is useful in more general cases because it allows for the extraction of energy consumption data from one device or from a complete set of devices used in a data center.

The goal of this research is to offer a public method for the creation of datasets focused on server processing stimuli, electrical consumption, and data collection. An open repository offers the opportunity to reproduce and customize this approach for the community in other environments, which is made possible by taking advantage of the open-source tools used by this technique. The following sections explain how to implement and create an energy-specific dataset.

According to the proposed goal, the idea is to create an energy consumption dataset as a starting point for research on applications oriented to optimize energy requirements in data centers and server farms. The combination of several machines enables the generation of data for different purposes, which can be analyzed and used as aggregated data or fractioned by type of application.

Therefore, this article presents a complete method that integrates different components for the creation of a dataset. This approach enables monitoring of energy consumption in real time. The contributions of this work are summarized as follows:

- A brief description of the software and hardware architectures for energy measurements;
- System stimuli through benchmarks used as charge tests;
- Extraction and processing of the obtained data;
- Analysis and results on the dataset itself;
- Analysis of possible future applications through energy efficiency optimization services, leveraging concepts like network function virtualization (NFV), software-defined networking (SDN), and dynamic resource allocation.

This paper is organized as follows, starting with the related work in Section 2. Section 3 shows the methodology adopted to choose a measurement architecture (software/hardware); apply a stimulus to the servers; plan some charge tests to obtain and store the energy consumption data; and, finally, process all the acquired information to generate the dataset. In Section 4, the obtained results are described and presented graphically to improve comprehension. Sections 5 and 6 discuss and analyze the obtained results and presents some proposals for future work, providing final conclusions of the work.

2. Related Work

This section of the article reviews other work related to our proposed data collection methodology and dataset creation. The need for good datasets to obtain accurate analysis and AI applications has been proven to be of high importance. This means that several works have been conducted with respect to this task.

In [5], a complete review of real and synthetic non-intrusive load monitoring (NILM) of energy datasets was introduced. The authors described the strengths and weaknesses of both approaches and depicted a complete dataset list. The dataset scenarios were mainly residential and home appliances, while only a few datasets included office appliances. Furthermore, to improve the results, the authors encouraged the running of a real-world measurement campaign.

A tool capable of generating extended semisynthetic data in real time was developed and presented in [6]. The authors focused on household appliances to characterize their consumption behavior, generating synthetic data on major appliances, such as electrical space heaters (ESHs) and electrical water heaters (EWHs). A statistical analysis of real-world data was used to create probabilistic models, taking into account the energy needs of the main appliances of a smart home. Furthermore, a simulation structure was developed to characterize the power consumption profiles and generate semisynthetic data.

In [7], a study was presented with a focus on the strengths and weaknesses of existing solar radiation datasets. The authors highlighted the importance of these data from a financial point of view in studying the performance of a solar project and in understanding the percentage of projects that can be financed, considering the uncertainty of solar resources. The development of a bankable dataset was proposed to understand the importance of resource variability and the nature of uncertainties in the various constituents of the data. Various datasets were also analyzed, introducing a methodology to validate them either through the use of concurrent ground-measured data or correlation with a longer-term ground-based dataset.

A very complete work was presented in [8], where the authors reviewed recent issues in dataset development, focusing on computer vision and natural language processing. In fact, datasets have been mythologized but could be impacted by errors and limitations. Errors such as incorrect content, haphazard collection, incorrect annotation, etc., may arise from data collection and annotation. Another problem is related to dataset maintenance is the difficulty of researchers in controlling dataset repositories if they are not closely supervised by an institution. Other one is the potential aspect about legal restrictions, as all images may not be protected by license or copyright. Although the author found critiques of insufficiently careful data annotation and documentation practice, which reduce the reliability of scientific inquiries of datasets, they hoped that their optimization and improvement would distribute data in a better way, supporting accuracy and respecting intellectual properties and privacy.

An overview of research on sustainable data modeling was presented in [9]. The authors discussed model energy efficiency, computational requirements in learning, and new algorithm approaches that use fewer resources in terms of memory requirements and computational processing. In addition, some interesting classifications were analyzed. A type of data classification (environmental, health biomedical field, universe, and space) was introduced by analyzing the amount of memory used to store the dataset. Further classification analysis of the modeling methodologies was reported. The main focus was on data modeling issues, including the relation between data models and characteristics. Some recommendations were provided, with the goal of increasing the accuracy of learning, reducing computational costs, and improving the large amount of data processing in a rapid and efficient way.

A complete pilot to generate data sets was described in [10]. Pereira et al. developed a system based on economic hardware to perform NILM over a typical building to generate a dataset. In addition, a software implementation was developed to read the measures by event from the sensors and save the data in the dataset. Furthermore, the authors evaluated the performance of the system using precision, recall, and F1 indicators that form the contingency matrix.

A ML approach was presented to detect home absence based on the electrical usage of appliances in [11]. The authors modified the UK-DALE dataset, including artificial outing events, generating a NILM dataset. Many ML algorithms were tested, and the obtained results prove that it is possible to use the power consumption of appliances to detect absence from the home.

In the same way, the authors of [12] presented the Reference Energy Disaggregation Data Set (REDD) to monitor the power consumption of several homes. The main objective of this dataset is the energy disaggregation of devices in different homes. To accomplish this task, a specific circuit device was implemented to monitor the energy consumption of the houses. The basic algorithms and data are available on a website.

In [13], a generic system identification tool was developed. The authors used a data-based modeling approach to optimize electrical power and resource consumption for a given load. The developed identification method focuses on a machine-independent approach to various types of machines and systems. Automated feature engineering and network configuration introduce the advantage of reduced effort and time to model a real system compared to physical modeling. The tool is capable of mapping the behavior of a system based on the measured input and output data of the system. Highly accurate data-based models were created for the energy and resource consumption of a machine or a system.

In [14], the authors investigated a model to optimize energy efficiency in multithread architectures. Some benchmarks were performed to generate datasets and analyze multithread behavior. This allowed the authors to develop a technique that automatically determines the best configuration for any application, with the goal of optimizing efficiency and energy consumption using ML algorithms. The proposed approach uses hardware performance information during the multithread runtime of applications and predicts and proposes the best energy-efficient configuration, setting the number of threads and the operating frequency.

Power measurement was introduced in [15]. The authors used the Chauffeur framework to create stimuli. This Java multiplatform benchmark is used to create a load on the server system while a power analyzer is examining the power consumption. The core of this work is represented by PTDaemon, a special program that is loaded in the system and is able to launch the Chauffeur framework and communicate with the power analyzer to examine the server power need as a function of the provided stimuli.

The application of datasets for the analysis of data-driven power systems is explained in [16], where an algorithm based on the separation of hyperplanes was used. According to the authors, this method allows for management of high input dimensionality with an itemized description of the security boundary. The dataset was used to perform optimal power flows of power systems with more than 125 control variables.

There are some works where a review of datasets, tools, models, and metrics were introduced. First, a work [17] provides a detailed overview of the open-source dataset model and tools that can be used in real-world energy communities oriented toward local energy communities (LECs). In other hand [18], this work is focused on NILM, which suggests not only datasets and benchmarking frameworks but also performance metrics for a complete evaluation of energy consumption estimation for NILM systems.

A novel regression-based modeling method was proposed in [19] to characterize the I-V characteristic curves of PV modules under different ambient conditions. The model uses a single hidden-layer feedforward neural network trained by a machine learning algorithm based on large datasets of I-V curves measured under operating conditions. The input of the model is meteorological data and the voltage, while the output is the current. The overall experimental results on six large datasets where the model was tested and optimized were reported.

In [20], the authors provide a complete overview of the landscape, current approaches, core applications, and challenges associated with dataset creation, in addition to providing recommendations. The authors also analyzed ML applications, techniques, and datasets and provided a list of known low-voltage-level open datasets, addressing and encouraging the reader to follow further research and development.

A method of forecasting power consumption using high-dimensional datasets was presented in [21]. The work shows that the ML models outperformed the benchmark models when regularized coefficients were adopted using high-dimensional datasets. In fact, the results obtained in this work show more accurate forecasts for all horizons.

The authors proposed a virtual sensing environment for smart building energy systems in [22]. The virtual sensing method includes an autoencoder, a model that introduces a correspondence with the real environment. The virtual applications supported by this method close to a real sensing system are diverse, such as monitoring, control, automation, fault detection and diagnosis, in situ sensor calibration, and prognostics. Furthermore, this method can be applied to prepare reliable and informative datasets, for example, reconstructing missing values in operational datasets based on the autoencoder reconstruction capability. Another important feature is represented by the calibration of the dataset from based on data from physical and virtual sensors, using the output layer of system-level virtual sensing. Finally, this method can provide extended diagnostic information for fault detection and diagnosis of building energy systems, using the differences between the input and output layers of the virtual sensing structure for residual-based fault detection.

A work focusing on imputation methods was presented in [23]. The authors evaluated the performance of five methods and imputation methods (two statistical and three ML) for the processing of electric power data over a two-year period. The experiment showed that the imputation methods generate higher error rates during the summer season compared to other periods. Furthermore, the test evidenced that ML methods represent a better choice for imputation of missing data during peak times, while statistical methods are more efficient during off-peak and semipeak times.

Other applications in which a dataset may be needed were presented in [24]. This study evaluated the potential of transfer learning in response to the problem of real-time electricity price forecasting. A set of hybrid datasets of the New York Independent System Operator real-time electricity prices was developed to train the models. The authors carried out all three aspects of modeling, helping to improve accuracy and introducing the use of hybrid datasets to add meaningful variance and reduce and prevent overfitting of the networks.

Open-StreetMap datasets were used in [25] to model and simulate energy systems of city district buildings. Using this GIS-based approach and the Python tool TEASER, an automated urban energy system model was generated. Building models include geometries, physical properties, and space-heating demand. The main objective of this method is the simplification of building data extraction and the rapid generation of city district models to perform simulations of space-heating loads on an urban scale.

Karantoumanis and Ploskas presented a study [26] on the estimation of power consumption in data centers utilizing machine learning techniques. The authors explored the use of machine learning models to predict power consumption based on various factors and parameters. They discussed the process of collecting data and feature selection, as well as the application of eight regression methods to estimate power consumption. This research provides valuable information on the use of machine learning for energy efficiency and optimization in data centers, comparing various adopted methods to obtain accurate predictions using the smallest number of parameters.

In [27], Uchechukwu et al. investigated energy consumption needs in cloud data centers. The authors provided formulas for calculating the total energy consumption of cloud environments, validating the models in six scenarios shown as examples. This research had the goal of designing energy-efficient infrastructure for data centers and server farms.

Another work focusing on a neural model for estimating the power consumption of virtual machines was presented in [28]. The authors proposed a deep neural network model to estimate power consumption of virtual machines, considering the impact of CPU, memory, disk, and network resource utilization. They evaluated the performance of their model, called Lapem, using real-world data collected from typical cloud services and comparing the model with other power estimation approaches. Lapem achieves accurate predictions of the power consumption of virtual machines. It also provides insights into energy-efficient resource allocation and management in data centers.

Previous work shows the utility of using datasets to improve energy system efficiency. However, in most cases, the methodology used to create the datasets is not fully explained. This fact could reduce the reproducibility of the procedures described above. The main goal of this work is to present a method for the creation of an energy consumption dataset that is easy to reproduce and implement. Moreover, scripts and code are available online [29–31] for download and testing. By launching the code mentioned above, it is possible to collect data for the creation of a new dataset, with the improvement of energy variations directly from the computing resources, enabling the analysis of the deployment of services. It is especially useful for virtualized environments, where everything is based on software instantiation and resource allocation.

3. Data Acquisition, Annotation, and Organization

3.1. Measurement System

To collect data on energy consumption, a system capable of monitoring the power demand of the equipment is needed. Many solutions are available to track and sense computer hardware. The software solution proposed in [32] allows energy consumption data from motherboard sensors to be read. This system has the advantage that the data are ready to be recorded and written in the dataset just by calling the API or interconnecting with the sensor driver. On the other hand, even if computer motherboards have sensors, not every system has drivers to collect data from the software layer. Furthermore, the power consumption information could be peripherally specific and not applicable to the global system. In order to ensure an exact measure, a system similar to that proposed in [33] with a hardware implementation could be recommended. This consists of a physical sensor introduced in the power line that measures the total power needs of the server.

Figure 1 shows a simple schematic in which a sensor is connected to the line that detects power consumption across the line. The physical probe represents a flexible solution on the Internet of Things (IoT) for the servers to be measured, enabling more accurate measures, even if it needs a connection to a device that can interpret the power consumption data , transmit them to a server to record the data, and save them in the dataset. Both solutions are valid to create a power consumption dataset; the software version is easy to develop, while the hardware version requires more effort in terms of time and economic resources to implement it. Depending on the accuracy needed, the two solutions allow for the creation of a dataset and permit eventual ML algorithm training, providing a power consumption estimate.

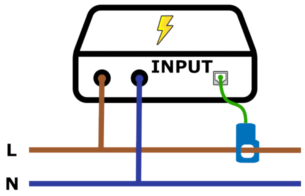


Figure 1. Sensor connection to the embedded system as input.

To create a coherent dataset, some electrical measurements are necessary, the most important of which are voltage, current, active power, and power factor. These metrics represent the energy information that is needed as a training requirement for the generation of optimization algorithms. It is important to provide this information to achieve a system that can monitor energy consumption and act according to the situation.

In this work, a proprietary IoT hardware solution was used to acquire the measurement data. In fact, a custom multichannel measurement system was developed and is presented in [34]. The device is connected through a power cable that feeds the server. Internally, a set of current transformer sensors acquire the servers' power consumption, and a measurement chip is responsible for interpreting the analog waveform from the sensors mentioned above. Once these instant factors are obtained, the measurement system sends them to the next stage over a network connection, where they are saved in the dataset. A brief scheme that summarizes the hardware measurement system is shown in Figure 2.

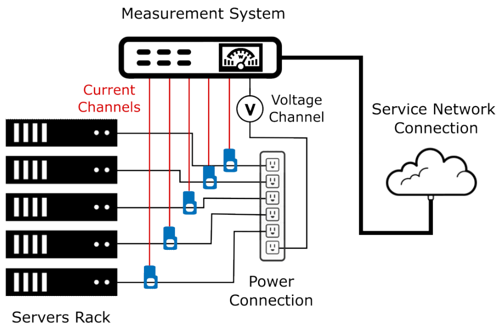


Figure 2. Architecture connection diagram. The measurement system is connected to the servers' power input to measure their energy consumption and send the information over a computer network.

3.2. Stress Tests

The measurement system described in the previous section has the ability to detect energy variations because it has been previously calibrated. Therefore, these measurement probes can be connected to the equipment, allowing for measurement of its energy consumption while running. However, it is necessary to create the dataset considering the need to save energy consumption information of different applications running on the servers. The proposal is to apply various stress tests to the servers in order to identify different state conditions.

Phoronix [35] is a software suite that provides a set of tests and benchmarks to evaluate the performance of servers, imposing stress on the machine at the desired level. The results of the tests are also entered into a ranking that permits a comparison of the different hardware architectures in terms of performance (CPU, disks, video cards, RAM, etc.). The Phoronix suite is easy to use; in fact, its architecture allows for management and installation of the stress tests. There are over 400 tests available—from CPU and memory to graphics—all based on simple command-line tools. In this specific case, since the servers do not have graphics cards with GPUs, tests have been chosen to increase the use of the CPU and memory.

Additionally, this software allows for automation of the scheduling and execution of the tests. Furthermore, it provides information logs on each test carried out, enabling verification of correct operation, as well as the performance of several of these tests in parallel.

Therefore, this test suite is responsible for pushing specific components of the PC to the limit, such as the system itself, the processor, or disk writing. The Phoronix suite can be automated through a mode called batch mode. The idea was to carry out different tests to vary the type of stress on the servers, analyzing the energy consumption, and to be able to simulate real situations as completely as possible.

3.3. Test Cases

The tests used and the types of stimuli that allow for simulation of real application consumption requirements are based on three Phoronix benchmarks (AsmFish, Radiance, and Sysbench). The following bullet list is organized in terms of use cases. Each use case describes the type of test, the corresponding benchmark used, and the reason for its selection. To facilitate understanding, a brief explanation or example is provided for each use case.

- No charge. A script is launched that does not run any charge tests to monitor the consumption of operating system processes. These data are the base case, allowing for a comparison between the normal or idle system behavior and the rest of the charging scripts.
- AsmFish. This is an advanced chess benchmark written in Assembler. This test spends the largest amount of resources, since its consumption is maximized for duration of the test. AsmFish simulates intensive applications that constantly request many resources, for example, multimedia processing in servers (YouTube, Twitch, etc.).
- Radiance. This is a benchmark for NREL Radiance, an open-source synthetic imaging system developed by the Lawrence Berkeley National Laboratory in California. This test allows for alternation of high stress with a low stress, simulating applications that are related to hourly consumption, that is, applications for which, at certain times, few users connect and spend few resources, whereas at other times, many users connect and spend many resources, for example, video or file compression.
- Sysbench. This is a Sysbench benchmark with built-in memory and CPU subtests. Sysbench is a scriptable multithreaded benchmark tool based on LuaJIT. This test places server resources at their maximum stress for a short time, allowing for the simulation of punctual charge applications, for example, Apache Server.
- Hybrids. Since it is possible to run different types of applications on a server, further tests are proposed that combine the previous tests in an effort to simulate days with more punctual charge needs, days with more intense charge, and days where the charge is predominantly alternated.

3.4. System/CPU Information

Once the stimulus is explained, it is important to gather the resource usage information to be saved and to create a log file to extract the data of interest. Energy and physical resource consumption are closely related, especially for virtualized environments, where deployment and execution of services are preassigned to virtual resources built upon physical resources. Therefore, the way the virtual resources are assigned and their use during the execution define the final energy consumption. The method adopted in this work is to start the Phoronix tests, monitor resource usage, and log into a file in order to associate the server's resource utilization and energy consumption and generate the dataset. Many monitoring tools are available; in this case, our work is based on a Linux system, and two tools (htop and CPU stat) were chosen. However, the idea could be used in other operating systems just by finding equivalent tools.

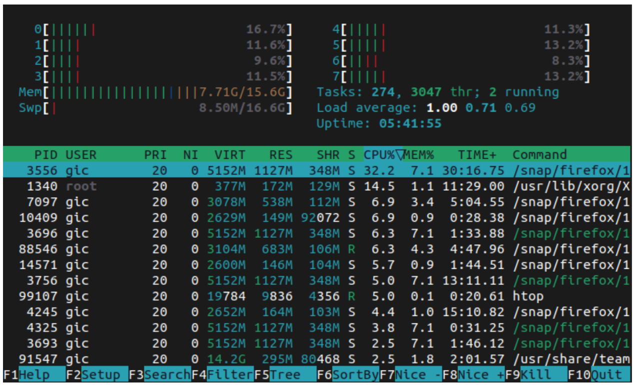
3.4.1. Htop

Htop is a terminal-based system monitoring application that shows data on various system processes and their resource consumption in a structured and human-readable format. Figure 3 shows a screen capture of the htop utility.

The information is organized, and a brief explanation of the data considered for this work is provided as follows.

- PID represents the ID of the process;
- CPU% represents the use of CPU;
- MEM% is the use of memory;
- Command is the name of the program that uses the resources.

The information presented by htop enable the collection of the relevant data for the creation of the dataset. The complete htop guide is available in [36].



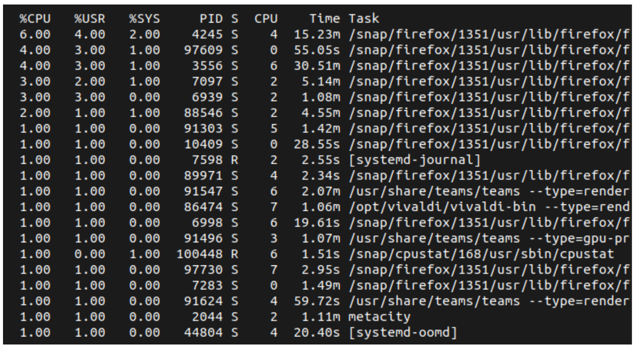
```
    0[||||||                  16.7%]     4[|||||                   11.3%]
    1[||||                    11.6%]     5[|||||                   13.2%]
    2[||||                     9.6%]     6[||||                     8.3%]
    3[||||                    11.5%]     7[|||||                   13.2%]
  Mem[|||||||||||||||||||7.71G/15.6G]   Tasks: 274, 3047 thr; 2 running
  Swp[|                 8.50M/16.6G]    Load average: 1.00 0.71 0.69
                                        Uptime: 05:41:55

    PID USER       PRI  NI  VIRT   RES    SHR S CPU%▽MEM%   TIME+  Command
   3556 gic         20   0 5152M 1127M   348M S 32.2  7.1 30:16.75 /snap/firefox/1
   1340 root        20   0  377M  172M   129M S 14.5  1.1 11:29.00 /usr/lib/xorg/X
   7097 gic         20   0 3078M  538M   112M S  6.9  3.4  5:04.55 /snap/firefox/1
  10409 gic         20   0 2629M  149M  92072 S  6.9  0.9  0:28.38 /snap/firefox/1
   3696 gic         20   0 5152M 1127M   348M S  6.3  7.1  1:33.88 /snap/firefox/1
  88546 gic         20   0 3104M  683M   106M R  6.3  4.3  4:47.96 /snap/firefox/1
  14571 gic         20   0 2600M  146M   104M S  5.7  0.9  1:44.51 /snap/firefox/1
   3756 gic         20   0 5152M 1127M   348M S  5.0  7.1 13:11.11 /snap/firefox/1
  99107 gic         20   0 19784  9836   4356 R  5.0  0.1  0:20.61 htop
   4245 gic         20   0 2652M  164M   103M S  4.4  1.0 15:10.82 /snap/firefox/1
   4325 gic         20   0 5152M 1127M   348M S  3.8  7.1  0:31.25 /snap/firefox/1
   3693 gic         20   0 5152M 1127M   348M S  2.5  7.1  1:46.12 /snap/firefox/1
  91547 gic         20   0 14.2G  295M  80468 S  2.5  1.8  2:01.57 /usr/share/team
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Figure 3. Htop Linux command example.

3.4.2. CPU Stat

Another utility that could be useful to extract resource utilization is represented by the command CPU stat shown in Figure 4.



```
  %CPU    %USR    %SYS     PID S   CPU     Time Task
  6.00    4.00    2.00    4245 S     4  15.23m /snap/firefox/1351/usr/lib/firefox/f
  4.00    3.00    1.00   97609 S     0  55.05s /snap/firefox/1351/usr/lib/firefox/f
  4.00    3.00    1.00    3556 S     6  30.51m /snap/firefox/1351/usr/lib/firefox/f
  3.00    2.00    1.00    7097 S     2   5.14m /snap/firefox/1351/usr/lib/firefox/f
  3.00    3.00    0.00    6939 S     2   1.08m /snap/firefox/1351/usr/lib/firefox/f
  2.00    1.00    1.00   88546 S     2   4.55m /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00   91303 S     5   1.42m /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00   10409 S     0  28.55s /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00    7598 R     2   2.55s [systemd-journal]
  1.00    1.00    0.00   89971 S     4   2.34s /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00   91547 S     6   2.07m /usr/share/teams/teams --type=render
  1.00    1.00    0.00   86474 S     7   1.06m /opt/vivaldi/vivaldi-bin --type=rend
  1.00    1.00    0.00    6998 S     6  19.61s /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00   91496 S     3   1.07m /usr/share/teams/teams --type=gpu-pr
  1.00    0.00    1.00  100448 R     6   1.51s /snap/cpustat/168/usr/sbin/cpustat
  1.00    1.00    0.00   97730 S     7   2.95s /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00    7283 S     0   1.49m /snap/firefox/1351/usr/lib/firefox/f
  1.00    1.00    0.00   91624 S     4  59.72s /usr/share/teams/teams --type=render
  1.00    1.00    0.00    2044 S     2   1.11m metacity
  1.00    1.00    0.00   44804 S     4  20.40s [systemd-oomd]
```

Figure 4. CPU stat Linux command example.

Similar to htop but more simple, this tool introduces the consumption of resources as follows (only the features relevant for this work are described):

- %CPU represents the use of CPU;
- PID represents the process ID;
- CPU is number of threads used by the task;
- Task is the name of the program that uses the resources.

Consult [37] to access the complete CPU stat guide.

3.4.3. Bash Script

The Phoronix suite can run in batch mode, even if, once the tests are finished, they need to be restarted. To generate a complete dataset, the execution time should be set according to the needed analysis. Furthermore, the information provided by the Linux commands that monitor resource consumption presented in Sections 3.4.1 and 3.4.2 must be stored and associated with the information about power consumption acquired with the hardware/software energy measurement system described in Section 3.1.

To automate these tasks, some Linux Bash scripts were generated that allow for a relaunch of the test once it has finished for the entire established analysis execution time. Additionally, to store server resource consumption data, the scripts save the output of the htop/CPU stat commands as a log file, creating an archive with the consumption of each system process. This solution allows for automation of the tests for the time desired to create a dataset.

The script structure shown in Figure 5 consists of a general loop that operates for the duration of the analysis time (T_{text}). In the time loop, the scripts launch the desired Phoronix test and check its execution. If the Phoronix test process stops, it is relaunched.

This is possible because the scripts are able to identify the process of the Phoronix test; if the process is not present, it means that the test is not running and should be relaunched. Furthermore, the scripts launch the htop/CPU stat, saving their output in a log file and creating the resource power archive as explained in the previous section. The scripts used in this work are provided and can be downloaded from [29].

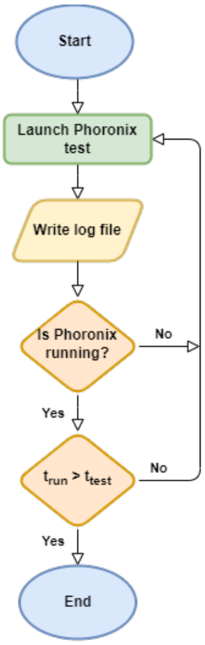


Figure 5. Script structure explanation.

With these scripts, it is possible to schedule a set of tests to analyze power consumption by simulating the power needs of various applications in a data center. Table 1 shows an example of experiment planning using five servers with a different test day that simulates various power needs.

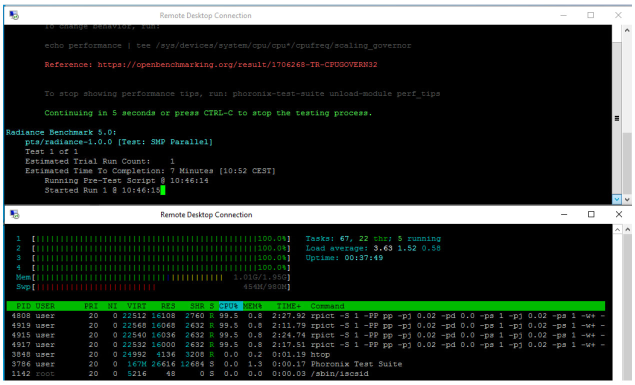
- Test 1: the script does not run any tests and creates a log file with the consumption of server operating system resources, which is very useful to compare power needs with the other tests.
- Test 2: the script simulates very intense applications that use 100% of the server resources.
- Test 3: this configuration simulates applications with punctual charge needs.
- Test 4: the test simulates applications that perform an intense task for a medium period of time and stop for a while.
- Test 5: hybrid test with high resource use predominance.
- Test 6: hybrid test in which medium-intensity tasks are launched.
- Test 7: hybrid test with applications that make a punctual use of resources.

The previous example shows a possible solution to schedule a set of tests, but is possible to adapt the set of tests to a different situation, number of servers, etc. Furthermore, it is possible to change Phoronix tests by installing a new test and modifying the scripts, maintaining the script structure shown in Figure 5 but changing the name of the test (command to be launched).

Figure 6 shows the Radiance test running. It is made up of two windows; in the upper part, it is possible to see the Radiance Phoronix test launched in the terminal and running, while in the lower part, the htop command is shown with the rpic processes that represent the Radiance test running on four threads (multithreading execution).

Table 1. Charge test planning.

Test ID	Server 1	Server 2	Server 3	Server 4	Server 5
1	No Charge	No Charge	No Charge	No Charge	No Charge
2	Asmfish	Asmfish	Asmfish	Asmfish	Asmfish
3	Sysbench	Sysbench	Sysbench	Sysbench	Sysbench
4	Radiance	Radiance	Radiance	Radiance	Radiance
5	Asmfish	Radiance	Asmfish	Sysbench	Asmfish
6	Radiance	Sysbench	Radiance	Asmfish	Radiance
7	Sysbench	Asmfish	Sysbench	Radiance	Sysbench



```
        To change behavior, run:

        echo performance | tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor

        Reference: https://openbenchmarking.org/result/1706268-TR-CPUGOVERN32


        To stop showing performance tips, run: phoronix-test-suite unload-module perf_tips

        Continuing in 5 seconds or press CTRL-C to stop the testing process.

Radiance Benchmark 5.0:
    pts/radiance-1.0.0 [Test: SMP Parallel]
    Test 1 of 1
    Estimated Trial Run Count:    1
    Estimated Time To Completion: 7 Minutes [10:52 CEST]
        Running Pre-Test Script @ 10:46:14
        Started Run 1 @ 10:46:15
```

```
  1  [||||||||||||||||||||||||||||||||||||||||||||||100.0%]   Tasks: 67, 22 thr; 5 running
  2  [||||||||||||||||||||||||||||||||||||||||||||||100.0%]   Load average: 3.63 1.52 0.58
  3  [||||||||||||||||||||||||||||||||||||||||||||||100.0%]   Uptime: 00:37:49
  4  [||||||||||||||||||||||||||||||||||||||||||||||100.0%]
  Mem[||||||||||||||||||||||||||||||||||||||||  1.01G/1.95G]
  Swp[||||||||||||||||||||||||||||                454M/980M]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 4808 user       20   0 22512 16108  2760 R 99.5  0.8  2:27.92 rpict -S 1 -PP pp -pj 0.02 -pd 0.0 -ps 1 -pj 0.02 -ps 1 -w+ -
 4919 user       20   0 22568 16068  2632 R 99.5  0.8  2:11.79 rpict -S 1 -PP pp -pj 0.02 -pd 0.0 -ps 1 -pj 0.02 -ps 1 -w+ -
 4915 user       20   0 22540 16036  2632 R 99.5  0.8  2:24.74 rpict -S 1 -PP pp -pj 0.02 -pd 0.0 -ps 1 -pj 0.02 -ps 1 -w+ -
 4917 user       20   0 22532 16000  2632 R 99.5  0.8  2:17.51 rpict -S 1 -PP pp -pj 0.02 -pd 0.0 -ps 1 -pj 0.02 -ps 1 -w+ -
 3848 user       20   0 24992  4136  3208 R  0.0  0.2  0:01.19 htop
 3786 user       20   0  167M 26616 12684 S  0.0  1.3  0:00.17 Phoronix Test Suite
 1142 root       20   0  5216    48     0 S  0.0  0.0  0:00.03 /sbin/iscsid
```

Figure 6. Example: Radiance test running with htop showing the consumption of process resources rpic.

3.5. Dataset Creation

In certain fields, the low availability of datasets causes a decrease in future analysis and development. Specifically, in the energetic discipline, it is complex to find datasets that combine internal data from computers with energy data measured in real time. This work aims to offer a methodology and resources to build a dataset from scratch and to seek future energy solutions, taking the obtained datasets as a starting point.

The data generated in this study were collected in a university laboratory with specific resources dedicated to internal tasks and developments. It is not adequate to compare with larger data centers, such as the most-used cloud providers (Amazon Web Services, Google Cloud, Microsoft Azure, etc.) [38], where a cloud structure has been created to offer services to a large community. The purpose of this research article is to provide a starting point and guidance for energy datasets. Table 2 shows statistical measurements to help understand the scope and limits of the system.

The dataset creation process presented in this work follows the best practices presented in [39–41] and the recommendations evaluated in [8]. These steps are general in every field (e.g., [42,43]), such as data creation and extraction, data processing, and analysis and visualization to corroborate the obtained results. The data were obtained in various rounds of tests, and, thanks to this approach, the tool was refined. This helped to understand how to manage the benchmarks to obtain a diverse and complete dataset, using different stimuli to obtain various energy parameters.

In addition to the complete dataset, the information logs containing the raw data on the performed stress tests are also available as individual machine and test log files. Maintaining complete data accessibility is important because it allows the community to check the reliability of the dataset, in addition to supporting further development and knowledge about energy consumption, including a possible starting point for future AI projects. The code and individual datasets can be obtained from the project's main repository [30] or on Kaggle [31]. The whole integration process is shown in Figure 7, with the steps explained in the following points.

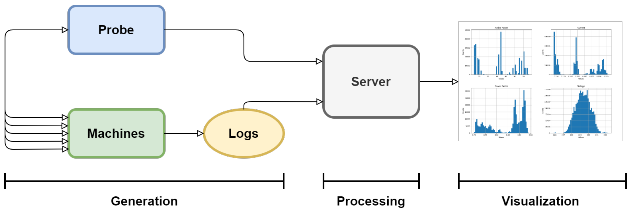


Figure 7. Dataset creation process.

Table 2. Energy measurement statistical operators for data acquisition in one probe channel.

	Active Power (W)	Current (A)	Power Factor	Voltage (V)
mean	0.819	40.829	0.221	229.364
std	0.031	10.684	0.065	0.868
median	0.832	42.000	0.220	229.360

3.5.1. Data Generation

The data used in the analysis were obtained from two sources. First, values were obtained from the physical probe connected to the equipment. These measurements are periodically sent to a REST API server, as described in Section 3.1. Second, the consumption data are generated by the server, where the Phoronix suite stress test is run, as described in Section 3.2. Specifically, the data are obtained by launching the htop system command and saving its output process as log files.

Data are extracted in parallel at the same time, allowing for synchronization with data sources and the creation of a data record filtered by timestamp. This practice permits the CPU values of the systems to be obtained and correlated with the energy parameters measured at the same time.

For every test conducted (as mentioned in Section 3.4.3), individual datasets are formed test by test. The data files from each source are synchronized with the probe's measurements in the following step. This allows the dataset to be filtered by test type and source number.

3.5.2. Data Processing

The initial stage consists of processing the data, analyzing the output log files, extracting relevant information, and merging them suitably. As discussed in the preceding section, data extraction was performed in parallel, integrating the data into a single record.

The probe sends raw data to the system, focusing on four parameters that characterize the energy measurements of servers in a data center: voltage, current, active power, and power factor. However, the system logs generated by the script described in Section 3.4.3 contain additional information that includes all operation system processes (Figures 3 and 4), which require data curation.

Analyzing the information from the htop and CPU stat tools contained in the log files, the CPU values used by the charge processes are needed to correlate with the values measured by the probe. The goal consists of extracting useful information from the raw data, creating a set of registers that can be varied, and analyzing the effects produced by the different charges on the systems.

3.5.3. Data Analysis and Visualization

The dataset is obtained by combining the two data sources. It is classified by probe measurement channel, where each channel corresponds to a server connection. The analysis focuses on power consumption and the types of tests performed. The seven tests carried out are described in Section 3.4.3 and Table 1. This categorization enables a detailed impact analysis of the system charging tests according to energy consumption and system process usage.

The no charge test was developed as a baseline for future analysis and comparison. Using this approach, it is possible to isolate the system processes from the benchmark techniques applied to stimulate the machines. Therefore, the no charge test served as a benchmark for comparison of the energy consumption of the other tests.

Graphical analysis helps to understand the relationship between the different identified characteristics. The conclusions drawn thanks to the evaluation enable the creation of a correct and reliable dataset. Data are analyzed with a special focus on finding possible information patterns of interest. Additionally, the evolution of measurements throughout the benchmarks is considered, taking into account the CPU progression for each test.

The dataset obtained by processing the power consumption information provides valuable insights into the impact of different stress tests on the system (statistics of a specific channel are presented in Table 2).

The dataset consists of 29 columns, including timestamps, CPU usage for various tests, energy measurements, and the source of the energy. With a total of 294.910 rows, the data provide a comprehensive view of the energy consumption and performance of the system under various stress tests. Data can be analyzed to observe the effects of charging tests on the energy consumption of the system and the usage of internal resources. Additionally, the dataset is grouped by test case and source, and individual datasets are also provided for each test case, allowing for a more in-depth analysis of the results. A summary of this information is organized in Table 3.

Table 3. Complete summary of the dataset.

Item	Value
# of Features	29
# of Entries	294.910
# of Sources	5
# of Tests Cases	7
Time Per Test	4 h
# of Files	8 csv (raw data considered separately)

4. Results

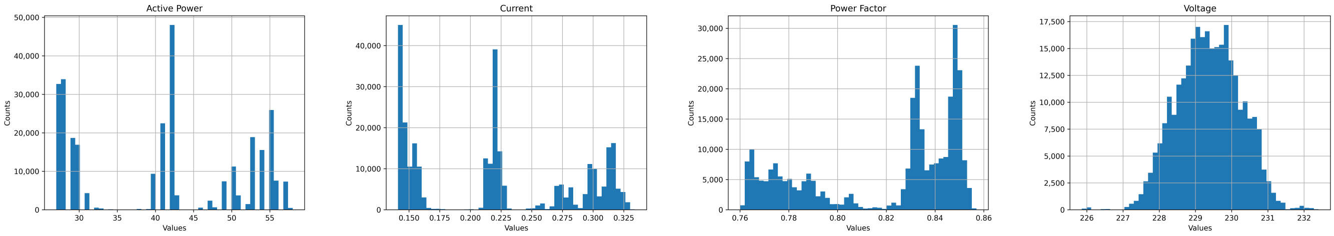
As an initial draft of the results, the histograms of the four primary energy features in the dataset are presented. The histograms presented in Figure 8 provide a visual representation of the distribution of values for active power, current, power factor, and voltage. The graphical representation of these data helps to improve understanding of the energy parameter behavior. The complete dataset obtained in the previous sections was used to generate these histograms.

In detail, in Figure 9, the distribution of energy characteristics is shown, corresponding to all the benchmark iterations. The distribution values support analysis of the behavior of the energy parameters according to the stress tests applied to the servers. In Figure 9, a single test is considered that shows how the methodology can be applied, setting a starting point for further analysis.

By analyzing the distribution of values, behavioral knowledge is acquired according to each energy characteristic as it evolves over time, considering the stress from the benchmark techniques. This methodology is demonstrated using a single test case, which serves as an initial demonstration and provides a starting point for further analysis.

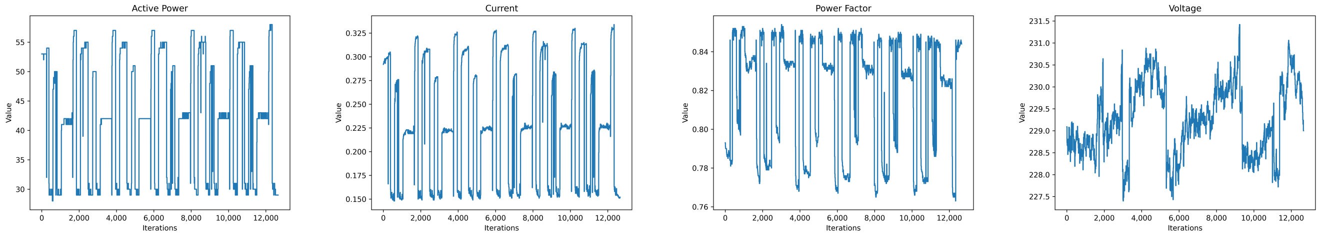
To evaluate the consumption of energy over a period of time, the CPU usage of the servers is considered. These values are acquired from the htop tool, where the range of 0–400 represents the usage of CPU threads (100 for each thread).

The energy consumption varies according to the stress test applied. Figure 10 shows an example in which the stress tests are activated from the scripts according to the flow chart presented in Figure 5.



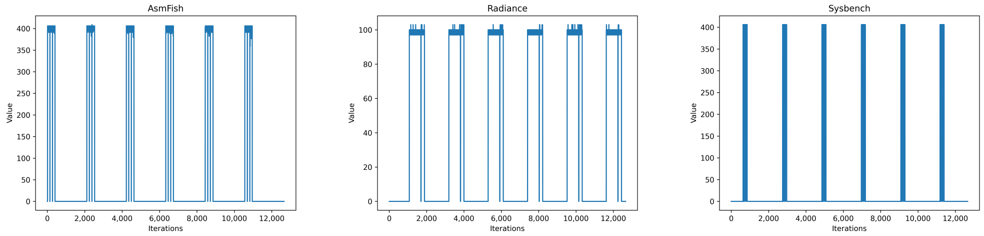
(a) Active Power. (b) Current. (c) Power Factor. (d) Voltage.

Figure 8. Histogram representation of energy measurements.



(a) Active Power. (b) Current. (c) Power Factor. (d) Voltage.

Figure 9. Example of energy measurement in stress tests.



(a) AsmFish. (b) Radiance. (c) Sysbench.

Figure 10. Example CPU usage measurements in tests.

When working with the complete dataset, it is crucial to identify the relationships between the variables. To ensure that the energy dataset accurately reflects the underlying energy characteristics, it is important to establish the degree of interdependence between the different features. One way to achieve this is to develop a correlation matrix by plotting the values on a heat map to examine the relationships between pairs of variables. The numerical values on the heat map represent the correlation coefficient, which measures the strength and direction of the relationship between the features. These values are presented in the range of −1.0 to 1.0 using the following interpretation:

- A positive value represents a direct correlation between the characteristics. In fact, both characteristics grow in the same direction. This is expressed by positive numerical values (where 1.0 is the maximum obtainable value) in the correlation matrix;
- A negative value represents an inverse relationship between the characteristics. This means, for example, that one characteristic is growing while the other is decreasing. This is expressed by negative numerical values, where −1.0 is the minimum obtainable value;
- Values close to zero show a weaker correlation, where no trend is present between the variables.

The aim of this research was to gather the results accurately. This was achieved through a dedicated testing and data collection process, using server infrastructure exclusively for these purposes. The use of a private structure for this experiment allowed for the appropriate time conditions for data collection (estimated to be four hours per day). Following the plan shown in Table 1, the results of each test case are presented as follows:

1. No charge. The purpose of the no charge test is to compare the operating system resource usage with other test cases and to examine the low correlation between parameters, as the benchmarks do not consume server resources. The energy behavior recorded by the probe during this test indicates the power consumption of the servers when they are in idle mode. Figure 11 shows the results for the mentioned test case.

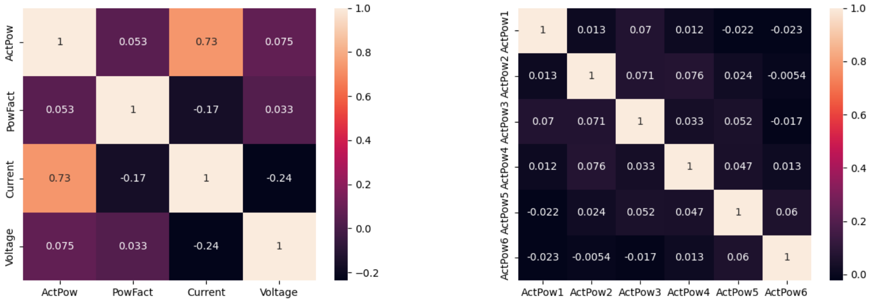


Figure 11. No charge test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

2. Radiance. This test shows a better relationship between the energy features, even though the correlation value is not particularly high. It is worth mentioning that all energy values have a complete correlation. Figure 12 exposes the correlation between measurements.

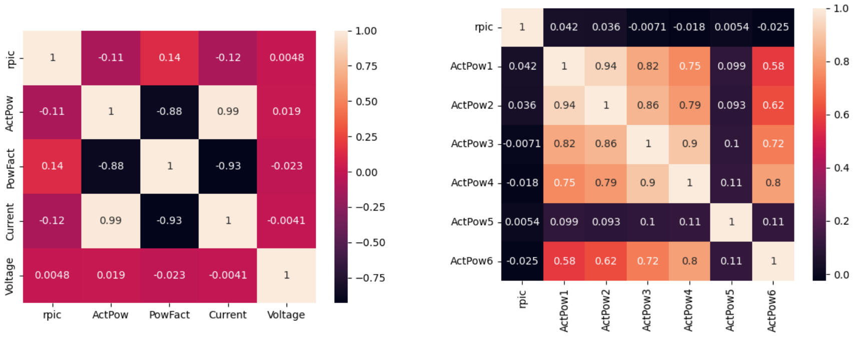


Figure 12. Radiance test correlations: measurements correlation (**left figure**); active power correlation (**right figure**).

3. Sysbench. The Sysbench test shows a noticeable improvement compared to the previous test. A stronger correlation is noticed between the system and the active power, with all energy values still fully correlated, except for voltage, which is expected to stay relatively stable. In Figure 13 are presented the correlations between measurements.

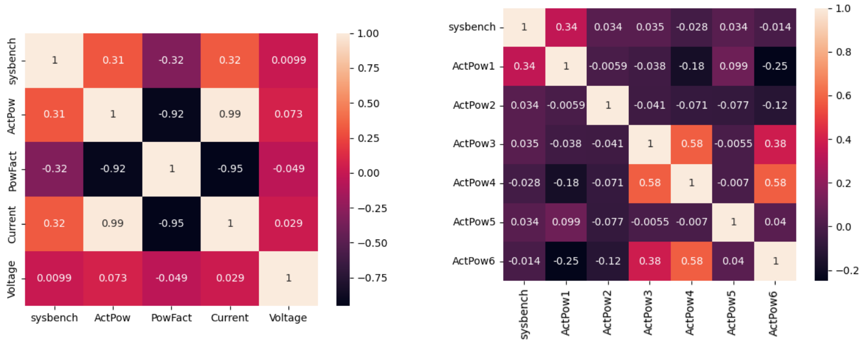


Figure 13. Sysbench test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

4. AsmFish. The enhanced correlation achieved in these tests, in comparison to the Sysbench test, is evident through the stronger correlation observed in the energy values. AsmFish test correlation values are expressed in Figure 14.

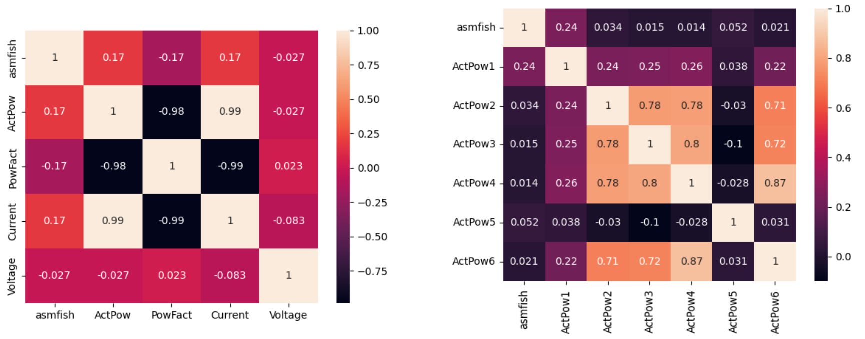


Figure 14. AsmFish test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

5. Hybrid 1. The hybrid tests of this methodology represent a modality that combines the execution of the Radiance, Sysbench, and AsmFish tests. Hybrid 1 simulates the prevalence of high-intensity activities. The correlation between the energy values is strong, except for the results from the Sysbench process. First hybrid test correlation values are shown in Figure 15.

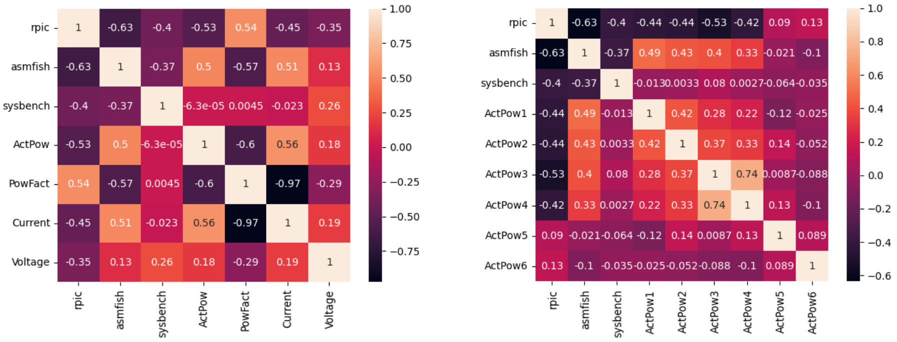


Figure 15. Hybrid 1 test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

6. Hybrid 2. This test simulates the prevalence of medium-intensity activities. The correlation between the outcomes improves for the AsmFish process; however, the

correlation for the Sysbench process is not complete. Correlation values for the hybrid 2 test is presented in Figure 16.

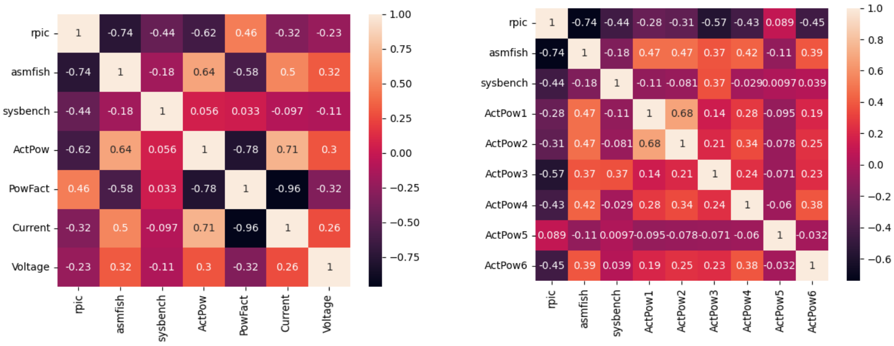


Figure 16. Hybrid 2 test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

7. Hybrid 3. The final test features the prevalence of punctual resource utilization. The correlation between the results shows a decline in the Sysbench process compared to the other high-intensity tests, while Asmfish and Radiance still exhibit a strong correlation. Final correlation values are in Figure 17.

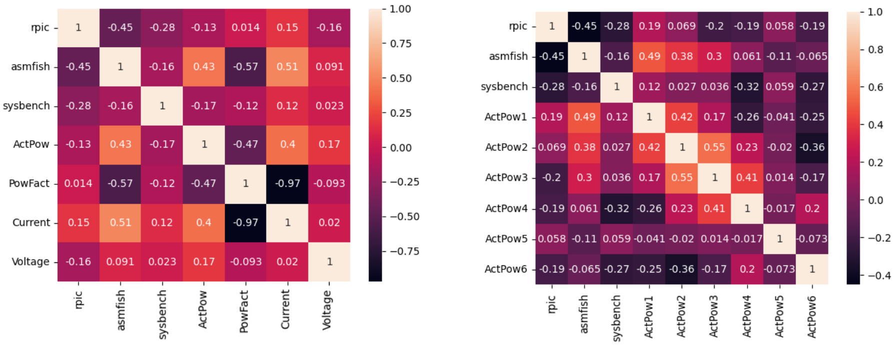


Figure 17. Hybrid 3 test correlations: measurement correlation (**left figure**); active power correlation (**right figure**).

5. Discussion

When considering the measurement system, two alternatives were initially proposed: a software version and a hardware version based on an IoT energy probe. For this particular study, the hardware version was selected due to its superior precision in recording data, despite requiring a more substantial effort to develop the hardware measurement system. This decision was made because the hardware measurement system can be customized to meet specific power consumption requirements, ensuring accurate and precise data acquisition. It was deemed that the benefits of the customized hardware system outweighed the additional effort involved in its creation.

The stress tests conducted in this study used the Phoronix benchmark suite, which specifically targets CPU and memory usage to simulate data centers for Internet applications, e-commerce usage, and more. It is worth noting that the Phoronix suite offers a diverse range of benchmarks, allowing for the simulation of various other data center applications.

The no charge test served as a reference point to evaluate the performance of other tests. From its correlation matrix, it was evident that all active power values were independent of each other.

In the Radiance test, the correlation between the active power values increased, while the relationship between the CPU usage of the test process and the active power was close to zero. This indicates that, on the basis of the measurement data, there was no significant correlation between the energy measurement values. These results may have been influenced by measurement lag, as launching the Radiance benchmark on multiple servers may cause variations in the start and end times, resulting in inaccurate measurements.

Similarly, in the Sysbench test, a correlation between the CPU usage of the test process and the energy measurements was observed, confirming the impact of the benchmark on the CPU. However, this benchmark generates punctual system stress, causing variations in the start and end times and potentially leading to measurement lag.

In the AsmFish test, an increase in the correlation between the active power values was observed, while there was a high correlation between the energy measurement values, albeit slightly lower than in the Sysbench test.

Hybrid tests, which involve a combination of stress tests, resolved the measurement lag issues existing in the previous tests. The continuous stress on the system over time due to a variety of benchmarks running concurrently reduced the lag in data collection and resulted in high correlation between the measurement values and active power. This can be seen in the high correlation in hybrid test 2, which is clearly depicted in its color map.

The active power measurement, which provided a more accurate assessment of energy consumption, is summarized in Table 4 for all the test cases and two sources. The table showcases the statistical representation of the essential values obtained from AP_1 (active power from source 1) and AP_2 (active power from source 2) measurements. The statistical analysis encompasses several metrics, such as mean, standard deviation, and interquartiles. Additionally, it incorporates three comparative indicators in relation to the no charge test: mean difference (±NC), percentage of mean increase (%Mean), and increase in standard deviation (%Std).

Table 4. Energy active power (AP) values of different test cases in watts.

Test	**No Charge**		**Radiance**		**Sysbench**		**AsmFish**		**Hybrid 1**		**Hybrid 2**		**Hybrid 3**	
Stats	**AP_1**	**AP_2**	**AP_1**	**AP_2**	**AP_1**	**AP_2**	**AP_1**	**AP_2**	**AP_1**	**AP_2**	**AP_1**	**AP_2**	**AP_1**	**AP_2**
Mean	27.40	29.30	44.10	47.90	49.60	50.30	54.00	49.90	45.70	49.90	46.20	50.40	42.20	46.10
Std	00.50	00.46	05.94	06.13	03.06	08.28	02.88	12.23	07.39	07.83	07.11	07.57	10.10	11.00
Min	27.00	29.00	27.00	29.00	29.00	31.00	28.00	29.00	28.00	29.00	28.00	29.00	28.00	29.00
25.00%	27.00	29.00	41.00	45.00	50.00	51.00	54.00	32.00	41.00	45.00	42.00	45.00	30.00	32.00
50.00%	27.00	29.00	42.00	46.00	50.00	55.00	54.00	58.00	47.00	51.00	47.00	50.00	42.00	46.00
75.00%	28.00	30.00	42.00	46.00	51.00	55.00	55.00	58.00	53.00	58.00	53.00	58.00	53.00	58.00
Max	28.00	30.00	57.00	61.00	51.00	56.00	55.00	59.00	56.00	60.00	56.00	60.00	58.00	62.00
±NC	NA	NA	17.00	19.00	22.00	21.00	27.00	21.00	18.00	21.00	19.00	21.00	15.00	17.00
%Mean	NA	NA	60.70	63.50	80.50	71.70	96.50	70.60	66.50	70.40	68.30	72.00	50.40	57.30
%Std	NA	NA	33.40	40.30	44.20	45.50	53.10	44.80	36.60	44.60	37.60	45.70	29.50	36.40

Upon examining Table 4, it becomes apparent that all the conducted tests exhibited a significant rise in active power. The minimum observed increase was 50% (hybrid 3), while the maximum reached nearly 96.5% in the case of the AsmFish test. Comparatively, the no charge test served as a reference point, yielding an average active power of 27.4 W. In stark contrast, the AsmFish test recorded an average of 54W, nearly doubling the baseline value. As evidenced by the findings, AsmFish exerted the most substantial influence on active power, despite not exhibiting the highest active power readings. Sysbench demonstrated the second-highest impact, while Radiance yielded the lowest impact among the three stress tests. The third hybrid test had the highest peak value for power source two, but its average was lower than the other two hybrid tests. The first and second hybrid tests showed results similar to the individual tests. Unlike the other tests, the hybrid tests did not reach the highest levels of active power, with AsmFish holding the record for the highest value.

6. Conclusions

This research aimed to address the scarcity of energy datasets that combine internal resource data with real-time energy measurements from an IoT module. By systematically collecting and analyzing energy data, the researchers aimed to fill this gap and provide valuable insights for energy consumption optimization in different systems and for different applications. The primary goal was to contribute to the understanding of energy consumption in real-world scenarios and establish a foundation for further research in this field.

The stress tests conducted in this study, as demonstrated in Table 4, effectively increased the energy values, generating diverse and valuable data for the creation of the dataset. These tests successfully stressed the systems, creating varied conditions that proved instrumental for analysis. The reliability of the tests was evident in their ability to significantly increase the active power measurements.

Analysis of the dataset revealed a correlation between the charging processes and the energy measurements, indicating that impairment processes influence the systems, resulting in varying outcomes depending on the specific test and the level of stress applied. These findings underscore the influence of impairment processes on energy consumption and highlight the importance of considering stress levels in optimizing energy efficiency.

The work presented in this article allows the community to continue to research, using these data as a starting point (available from the GitHub project [30] or from the Kaggle data repository [31]), whether as a source of data for an intelligence application or as a source of inspiration for the creation of other datasets, encouraging new ones to emerge. The evolution of dataset views over time in our own Kaggle repository is demonstrated in Figure 18, showing a stable interest in energy consumption and the performance of computing systems.

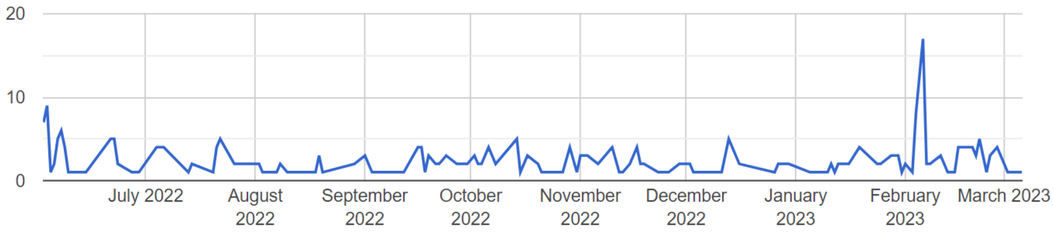


Figure 18. Evolution of views of the dataset on the Kaggle platform.

We have focused on the effects on CPU in particular, but the proposed method can be expanded to the rest of the components. As an example, in future work, it could be interesting to consider memory consumption, disk file read/write transactions, and possible uses of graphics cards (energy consumption, GPU usage, temperature, etc.). This can be applied with the same suite used in the present work, i.e., Phoronix, as it allows for the benchmarking of disks (for example, IOzone [44]) useful for the simulation of backup and database application data centers, graphics rendering (Yquake or Left 4 Dead 2 games [45,46]) for games and hosting services, CUDA tests (NVIDIA CUDA Blender [47]) for machine and deep learning training server farms, or other applications for which the power need can be simulated by installing the appropriate corresponding stress tests.

This implies that a more focused study of power consumption could be conducted based on a specific application feature or by selecting a group of different benchmarks to simulate the concurrent use of multiple applications in a data center. The variety of power consumption simulation scenarios proves the flexibility and scalability of the method presented in this work. In general, data that can be obtained from the systems can serve to increase the number of characteristics that add robustness to the data.

In addition to their utility in optimizing energy consumption, the collected data hold significant potential for AI applications. By leveraging these datasets, it becomes possible to develop models that can evaluate the achieved efficiency. Moreover, employing clustering techniques enables the differentiation of data points based on varying charge levels, providing valuable insights into the distribution patterns within the dataset.

The emergence of next-generation communication technologies, such as 5G and beyond, has introduced novel possibilities, such as service virtualization, dynamic scaling, dynamic orchestration, and service optimization. Optimization techniques in this context are typically focused on enhancing quality of experience (QoE) and quality of service (QoS). The data presented in this study offer a unique opportunity to optimize service energy consumption, leading to significant impacts not only from a sustainability perspective but also from an economic standpoint. By leveraging the insights gained from the data analysis, energy optimization measures can be implemented to achieve both environmental and cost-related benefits.

For these environments, optimization based on intelligent systems makes it possible to significantly improve the energy consumption derived from the use of physical resources associated with virtualization. In the ongoing effort to optimize energy consumption, a current preliminary development focuses on the softwarization of multiple domains and the increased use of virtualization. A paradigmatic case is the integration of multiaccess edge computing (MEC) capabilities in next-generation communication networks, where reducing latency and efficiently provisioning services are crucial. The configuration of virtualization parameters, resource allocation, and deployment of virtual network functions play a pivotal role in energy consumption. By carefully optimizing these factors, significant improvements in energy efficiency can be achieved, resulting in reduced environmental impact and enhanced operational cost-effectiveness. This ongoing work aims to develop novel approaches and techniques that intelligently manage virtualization in MEC environments, with the overarching goal of maximizing energy efficiency while maintaining optimal service performance. By addressing the intricate relationship between virtualization and energy consumption, this research endeavor aims to contribute to the development of sustainable and energy-efficient solutions for future communication networks.

Author Contributions: Conceptualization, A.d.R., G.C., S.C.-S., J.S. and D.J.; Formal analysis, A.d.R. and G.C.; Investigation, A.d.R., G.C., J.S. and D.J.; Methodology, A.d.R., G.C. and J.S.; Software, A.d.R. and G.C.; Supervision, D.J. and J.F.-A.; Validation, A.d.R. and D.J.; Writing—original draft, A.d.R. and G.C.; Writing—review and editing, S.C.-S., J.S. and D.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in Kaggle (at 10.34740/kaggle/ds/2233513) [31].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
ML	Machine Learning
NFV	Network Function Virtualization
SDN	Software-defined networking
NILM	Non Intrusive Load Monitoring
ESH	Electrical Space Heaters
EWH	Electrical Water Heaters
REDD	Reference Energy Disaggregation Data Set
LEC	Local energy communities

References

1. Miraz, M.H.; Ali, M.; Excell, P.S.; Picking, R. Internet of Nano-Things, Things and Everything: Future Growth Trends. *Future Internet* **2018**, *10*, 68. [CrossRef]
2. Tyagi, H. Data Repositories for Almost Every Type of Data Science Project. *Medium*. 2020. Available online: https://towardsdatascience.com/data-repositories-for-almost-every-type-of-data-science-project-7aa2f98128b (accessed on 14 March 2023).
3. Research, G.V. AI Training Dataset Market Size, Share & Trends Analysis Report By Type (Text, Image/Video, Audio), by Vertical (IT, Automotive, Government, Healthcare, BFSI), by Regions, and Segment Forecasts, 2022–2030. *Grand View Research*. 2020. Available online: https://www.grandviewresearch.com/industry-analysis/ai-training-dataset-market#:~:text=Report%20Overview,e%2Dcommerce%2C%20and%20healthcare (accessed on 14 March 2023).
4. Pfenninger, S.; DeCarolis, J.; Hirth, L.; Quoilin, S.; Staffell, I. The importance of open data and software: Is energy research lagging behind? *Energy Policy* **2017**, *101*, 211–215. [CrossRef]
5. Iqbal, H.K.; Malik, F.H.; Muhammad, A.; Qureshi, M.A.; Abbasi, M.N.; Chishti, A.R. A critical review of state-of-the-art non-intrusive load monitoring datasets. *Electr. Power Syst. Res.* **2021**, *192*, 106921. [CrossRef]
6. Hosseini, S.; Kelouwani, S.; Agbossou, K.; Cardenas, A.; Henao, N. A semi-synthetic dataset development tool for household energy consumption analysis. In Proceedings of the 2017 IEEE International Conference on Industrial Technology (ICIT), Toronto, ON, Canada, 22–25 March 2017; pp. 564–569. [CrossRef]
7. Vignola, F.; Grover, C.; Lemon, N.; McMahan, A. Building a bankable solar radiation dataset. *Solar Energy* **2012**, *86*, 2218–2229. [CrossRef]
8. Paullada, A.; Raji, I.D.; Bender, E.M.; Denton, E.; Hanna, A. Data and its (dis)contents: A survey of dataset development and use in machine learning research. *Patterns* **2021**, *2*, 100336. [CrossRef] [PubMed]
9. Al-Jarrah, O.Y.; Yoo, P.D.; Muhaidat, S.; Karagiannidis, G.K.; Taha, K. Efficient Machine Learning for Big Data: A Review. *Big Data Res.* **2015**, *2*, 87–93. [CrossRef]
10. Pereira, L.; Ribeiro, M.; Nunes, N. Engineering and deploying a hardware and software platform to collect and label non-intrusive load monitoring datasets. In Proceedings of the 2017 Sustainable Internet and ICT for Sustainability (SustainIT), Funchal, Portugal, 6–7 December 2017; pp. 1–9. [CrossRef]
11. Lentzas, A.; Vrakas, D. Machine learning approaches for non-intrusive home absence detection based on appliance electrical use. *Expert Syst. Appl.* **2022**, *210*, 118454. [CrossRef]
12. Kolter, J.Z.; Johnson, M.J. REDD: A public data set for energy disaggregation research. In Proceedings of the Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego CA, USA, 21 January 2011; Volume 25, pp. 59–62.
13. Weber, T.; Sossenheimer, J.; Schäfer, S.; Ott, M.; Walther, J.; Abele, E. Machine Learning based System Identification Tool for data-based Energy and Resource Modeling and Simulation. *Procedia CIRP* **2019**, *80*, 683–688. [CrossRef]
14. Sayadi, H.; Patel, N.; Sasan, A.; Homayoun, H. Machine Learning-Based Approaches for Energy-Efficiency Prediction and Scheduling in Composite Cores Architectures. In Proceedings of the 2017 IEEE International Conference on Computer Design (ICCD), Boston, MA, USA, 5–8 November 2017; pp. 129–136. [CrossRef]
15. von Kistowski, J.; Lange, K.D.; Arnold, J.A.; Sharma, S.; Pais, J.; Block, H. Measuring and Benchmarking Power Consumption and Energy Efficiency. In Proceedings of the Companion of the 2018 ACM/SPEC International Conference on Performance Engineering, Berlin, Germany, 9–13 April 2018; ICPE ’18; Association for Computing Machinery: New York, NY, USA, 2018; pp. 57–65. [CrossRef]
16. Venzke, A.; Molzahn, D.K.; Chatzivasileiadis, S. Efficient creation of datasets for data-driven power system applications. *Electr. Power Syst. Res.* **2021**, *190*, 106614. [CrossRef]
17. Kazmi, H.; Munné-Collado, Í.; Mehmood, F.; Syed, T.A.; Driesen, J. Towards data-driven energy communities: A review of open-source datasets, models and tools. *Renew. Sustain. Energy Rev.* **2021**, *148*, 111290. [CrossRef]
18. Pereira, L.; Nunes, N. Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools—A review. *WIREs Data Min. Knowl. Discov.* **2018**, *8*, e1265.
19. Chen, Z.; Yu, H.; Luo, L.; Wu, L.; Zheng, Q.; Wu, Z.; Cheng, S.; Lin, P. Rapid and accurate modeling of PV modules based on extreme learning machine and large datasets of I–V curves. *Appl. Energy* **2021**, *292*, 116929. [CrossRef]
20. Haben, S.; Arora, S.; Giasemidis, G.; Voss, M.; Vukadinović Greetham, D. Review of low voltage load forecasting: Methods, applications, and recommendations. *Appl. Energy* **2021**, *304*, 117798. [CrossRef]
21. Albuquerque, P.C.; Cajueiro, D.O.; Rossi, M.D. Machine learning models for forecasting power electricity consumption using a high dimensional dataset. *Expert Syst. Appl.* **2022**, *187*, 115917. [CrossRef]
22. Hong, Y.; Yoon, S.; Kim, Y.S.; Jang, H. System-level virtual sensing method in building energy systems using autoencoder: Under the limited sensors and operational datasets. *Appl. Energy* **2021**, *301*, 117458. [CrossRef]
23. Wang, M.C.; Tsai, C.F.; Lin, W.C. Towards missing electric power data imputation for energy management systems. *Expert Syst. Appl.* **2021**, *174*, 114743. [CrossRef]
24. Yang, H.; Schell, K.R. Real-time electricity price forecasting of wind farms with deep neural network transfer learning and hybrid datasets. *Appl. Energy* **2021**, *299*, 117242. [CrossRef]
25. Schiefelbein, J.; Rudnick, J.; Scholl, A.; Remmen, P.; Fuchs, M.; Müller, D. Automated urban energy system modeling and thermal building simulation based on OpenStreetMap data sets. *Build. Environ.* **2019**, *149*, 630–639. [CrossRef]

26. Karantoumanis, E.; Ploskas, N. Power Consumption Estimation in Data Centers Using Machine Learning Techniques. In Proceedings of the Learning and Intelligent Optimization: 14th International Conference, LION 14, Athens, Greece, 24–28 May 2020; Revised Selected Papers; Springer: Berlin/Heidelberg, Germany, 2020; pp. 195–200. [CrossRef]
27. Uchechukwu, A.; Li, K.; Shen, Y. Energy Consumption in Cloud Computing Data Centers. In Proceedings of the International Conference on Cloud Computing, Anchorage, AK, USA, 27 June–2 July 2014. [CrossRef]
28. Yu, X.; Zhang, G.; Li, Z.; Liangs, W.; Xie, G. Toward Generalized Neural Model for VMs Power Consumption Estimation in Data Centers. In Proceedings of the ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7. [CrossRef]
29. Conti, G. GitHub. Online Charge Script Repository. 2022. Available online: https://github.com/gic81/charge_script_dataset (accessed on 2 June 2022).
30. del Rio, A. GitHub. Project Repository Energy-Dataset-Creation. 2022. Available online: https://github.com/Kaiser-14/energy-dataset-creation (accessed on 10 June 2022).
31. del Rio, A.; Conti, G.; Serrano, J.; Jimenez, D. Energy-Dataset. Kaggle. 2022. Available online: https://doi.org/10.34740/KAGGLE/DS/2233513 (accessed on 10 June 2022).
32. Fieni, G.; Rouvoy, R.; Seinturier, L. SmartWatts: Self-Calibrating Software-Defined Power Meter for Containers. In Proceedings of the 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), Melbourne, Australia, 11–14 May 2020; pp. 479–488. [CrossRef]
33. Bedard, D.; Lim, M.Y.; Fowler, R.; Porterfield, A. PowerMon: Fine-grained and integrated power monitoring for commodity computer systems. In Proceedings of the Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon), Concord, NC, USA, 18–21 March 2010; pp. 479–484. [CrossRef]
34. Conti, G.; Jimenez, D.; del Rio, A.; Castano-Solis, S.; Serrano, J.; Fraile-Ardanuy, J. A Multi-Port Hardware Energy Meter System for Data Centers and Server Farms Monitoring. *Sensors* **2023**, *23*, 119. [CrossRef] [PubMed]
35. Phoronix. Phoronix Benchmarking and Test Suite. 2022. Available online: https://www.phoronix-test-suite.com/ (accessed on 15 February 2023).
36. Htop. Htop Manual. Available online: https://man7.org/linux/man-pages/man1/htop.1.html (accessed on 14 June 2020).
37. CPU Stat. Cpustat Manual. Available online: https://github.com/uber-archive/cpustat (accessed on 14 June 2020).
38. Gupta, B.; Mittal, P.; Mufti, T. A Review on Amazon Web Service (AWS), Microsoft Azure & Google Cloud Platform (GCP) Services. In Proceedings of the 2nd International Conference on ICT for Digital, Smart, and Sustainable Development, ICIDSSD 2020, New Delhi, India, 27–28 February 2020. [CrossRef]
39. Gonfalonieri, A. How to Build A Data Set for Your Machine Learning Project. *Medium*. 2019. Available online: https://towardsdatascience.com/how-to-build-a-data-set-for-your-machine-learning-project-5b3b871881ac (accessed on 20 February 2023).
40. Pham, K. How to Create a Dataset for Machine Learning. *Medium*. 2022. Available online: https://medium.com/@khang.pham.exxact/how-to-create-a-dataset-for-machine-learning-ec8707207e8e (accessed on 20 February 2023).
41. Pearkes, K. Improving Dataset Creation for Machine Learning. MistyWest. 2021. Available online: https://www.mistywest.com/posts/improving-dataset-creation-for-machine-learning/ (accessed on 20 February 2023).
42. Theodorou, L.; Massiceti, D.; Zintgraf, L.; Stumpf, S.; Morrison, C.; Cutrell, E.; Harris, M.T.; Hofmann, K. *Disability-First Dataset Creation: Lessons from Constructing a Dataset for Teachable Object Recognition with Blind and Low Vision Data Collectors*; ASSETS '21; Association for Computing Machinery: New York, NY, USA, 2021. [CrossRef]
43. Kashiyama, T.; Pang, Y.; Sekimoto, Y. Open PFLOW: Creation and evaluation of an open dataset for typical people mass movement in urban areas. *Transp. Res. Part Emerg. Technol.* **2017**, *85*, 249–267. [CrossRef]
44. Iozone. IOzone Disks Benchmark. Openbenchmarking. Available online: https://openbenchmarking.org/test/system/iozone (accessed on 14 June 2020).
45. yquake2. YQuake Graphics Benchmark. Openbenchmarking. Available online: https://openbenchmarking.org/test/pts/yquake2 (accessed on 14 June 2020).
46. Left4Dead2. Left 4 Dead 2 Graphics Benchmark. Openbenchmarking. Available online: https://openbenchmarking.org/test/pts/l4d2 (accessed on 14 June 2020).
47. Larabel, M. NVIDIA CUDA Blender 3.2 Benchmark. Openbenchmarking. Available online: https://openbenchmarking.org/test/pts/blender (accessed on 14 June 2020).

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