sensors

MDPI

Article

Porting Rulex Software to the Raspberry Pi for Machine Learning Applications on the Edge †

Ali Walid Daher [1,2,3,4], Ali Rizik [1,2], Marco Muselli [3,4], Hussein Chible [2] and Daniele D. Caviglia [1,*]

[1] COSMIC Lab, Department of Electrical, Electronic and Telecommunications Engineering and Naval Architecture (DITEN), University of Genoa, 16145 Genoa, Italy; ali.daher@edu.unige.it (A.W.D.); ali.rizik@edu.unige.it (A.R.)
[2] MECRL Laboratory, Ph.D. School for Sciences and Technology, Lebanese University, Beirut 6573/14, Lebanon; hchible@ul.edu.lb
[3] Consiglio Nazionale delle Ricerche, Institute of Electronics Computer and Telecommunication Engineering (IEIIT), 16149 Genoa, Italy; marco.muselli@ieiit.cnr.it
[4] Rulex Innovation Labs, Rulex Inc., 16122 Genoa, Italy
* Correspondence: daniele.caviglia@unige.it; Tel.: +39-010-33-56-587

† This paper is an extended version of our paper published in Daher, A.W.; Rizik, A.; Muselli, M.; Chible, H.; Caviglia, D.D. Porting Rulex Machine Learning Software to the Raspberry Pi as an Edge Computing Device. In Proceedings of the International Conference on Applications in Electronics Pervading Industry, Environment and Society, Online Event, 19–20 November 2020.

Abstract: Edge Computing enables to perform measurement and cognitive decisions outside a central server by performing data storage, manipulation, and processing on the Internet of Things (IoT) node. Also, Artificial Intelligence (AI) and Machine Learning applications have become a rudimentary procedure in virtually every industrial or preliminary system. Consequently, the Raspberry Pi is adopted, which is a low-cost computing platform that is profitably applied in the field of IoT. As for the software part, among the plethora of Machine Learning (ML) paradigms reported in the literature, we identified Rulex, as a good ML platform, suitable to be implemented on the Raspberry Pi. In this paper, we present the porting of the Rulex ML platform on the board to perform ML forecasts in an IoT setup. Specifically, we explain the porting Rulex's libraries on Windows 32 Bits, Ubuntu 64 Bits, and Raspbian 32 Bits. Therefore, with the aim of carrying out an in-depth verification of the application possibilities, we propose to perform forecasts on five unrelated datasets from five different applications, having varying sizes in terms of the number of records, skewness, and dimensionality. These include a small Urban Classification dataset, three larger datasets concerning Human Activity detection, a Biomedical dataset related to mental state, and a Vehicle Activity Recognition dataset. The overall accuracies for the forecasts performed are: 84.13%, 99.29% (for SVM), 95.47% (for SVM), and 95.27% (For KNN) respectively. Finally, an image-based gender classification dataset is employed to perform image classification on the Edge. Moreover, a novel image pre-processing Algorithm was developed that converts images into Time-series by relying on statistical contour-based detection techniques. Even though the dataset contains inconsistent and random images, in terms of subjects and settings, Rulex achieves an overall accuracy of 96.47% while competing with the literature which is dominated by forward-facing and mugshot images. Additionally, power consumption for the Raspberry Pi in a Client/Server setup was compared with an HP laptop, where the board takes more time, but consumes less energy for the same ML task.

Keywords: edge computing; internet of things; machine learning; image classification; pre-processing

Citation: Daher, A.W.; Rizik, A.; Muselli, M.; Chible, H.; Caviglia, D.D. Porting Rulex Software to the Raspberry Pi for Machine Learning Applications on the Edge. *Sensors* **2021**, *21*, 6526. https://doi.org/10.3390/s21196526

Academic Editors: Francesco Bellotti, Riccardo Berta, Sergio Saponara and Alessandro De Gloria

Received: 15 August 2021
Accepted: 27 September 2021
Published: 29 September 2021

1. Introduction

The Internet of Things paradigm is rapidly extending to many sectors of society because it allows to substantially improve the monitoring or control of complex and extensive processes, offering an innovative approach for multiple fields of application,

such as quality of life, urban challenges, logistics, agriculture and livestock, climate change, mass production, health, energy and water production and distribution, and many more.

The huge amount of data produced by the nodes of the networks of which the Internet of Things is made up must be processed in an efficient and effective way, and ML techniques are certainly among the most suitable for this purpose. Therefore, it is straightforward that Machine Learning tools can play a key role in further expanding the scope of applications, as well as their effectiveness. In past implementations, Machine Learning forecasts have been performed on a remote server before delivering results on the IoT Computing Node to limit network traffic, Edge Computing [1,2] setups can be employed to avoid intensive cloud access and keep that data storage and processing on the IoT device as much as possible.

Our work is placed in this perspective. Notably, this paper reports an extended version of the work previously presented at ApplePies 2020 [3]. The system we report is based on the Raspberry Pi platform [4,5], which is a low-cost, low-power credit-card-sized board that is used for embedded system and general-purpose computing applications. As for Machine Learning software, we have adopted for this investigation Rulex [6], which can be found online in [7], an AI (Artificial Intelligence) environment intended for non-domain experts, and we have ported it to the Raspberry Pi platform.

Rulex was ported to three different Operating Systems, namely to Windows 32 Bits, Ubuntu 64 Bits, and on Raspbian 32 Bits which is the official Operating System (OS) of the Raspberry Pi. All external and internal dependencies have been compiled [3] and verified. Moreover, the Client/Server setup has been used to perform forecasts on the edge after debugging the software through its source code.

To explore the application possibilities of this operating environment, in addition to the Radar Classification dataset already reported in [3], three new pre-processed datasets taken from diverse domains were also implemented using Rulex on the Edge. These encompass a Human Activity Detection dataset using Smartphones, a Brainwave Mental State Classification dataset, and an Activity Recognition dataset for Dumpers in earth moving sites that are also recorded using Smartphones. We also investigated the problem of performing gender detection. To this end, a new pre-processing Algorithm for image classification was developed to convert facial images into Time-series using a contour-based approach.

The contributions presented in this paper include the porting of a high performance ML package on the Raspberry Pi in a Client/Server setup, as well as the development of a novel pre-processing Algorithm that converts images into Time-series using statistical measurements, that are directly applicable to any ML configuration.

The methodology in the paper consists of compiling all the required libraries on the target platforms. The correct version of the libraries should be chosen where all libraries should be compatible with their predecessor since libraries may be built on top of each other. Also, in the linking process, the code needs to be changed to satisfy all the target platforms (Windows 64-bits, Windows 32-bits, Raspbian 32-bits, and Ubuntu 64-bits). Finally, the fully featured Rulex software package can be run on open-source hardware. Additionally, the pre-processing contribution and the ML tests on the system using multiple datasets and, in a Client/Server arrangement demonstrate the system effectiveness.

In the rest of this paper Section 2 presents the literature review, Section 3 describes the actions taken to bring Rulex on Raspberry Pi in a Client/Server arrangement [8], Section 4 presents in detail the image pre-processing Algorithm, and reports the results of all the ML forecasts and energy consumption achieved. Finally, Section 5 draws the conclusions.

2. Literature Review

2.1. Hardware Platform

The hardware platform adopted in this work is the multi-purpose Raspberry Pi [4,5]. It was conceived to handle application-specific tasks, as well as being used for everyday computing operations. It supports Universal Serial Bus (USB), HDMI, and SD Card

connections, in addition to having standard digital Input/Output pins controlled through the on-board ARM microcontroller.

Regarding Communication protocols, the Raspberry Pi orts Local Area Network (LAN) and Wi-Fi connectivity. Furthermore, the Raspberry Pi's digital pins can be used for interfacing with a wide variety of peripherals, for applications ranging from motor control, LCD Display functions, and may more and can be interfaced with compatible smart sensor modules.

2.2. Machine Learning Platform

The AI suite named Rulex (an acronym for Rule Extraction) has been created specifically for the management, the visualization, and the analysis of data: it consists in an integrated visual platform which allows to perform any operation in a simple and direct way, freeing the user from the necessity of knowing implementation details about memorization and execution [4]. It actually implements many Machine Learning Algorithms (both supervised and unsupervised) such as Logic Learning machine (LLM) [9], Neural Networks, Binary Trees, K-Nearest Neighbor (KNN), and others through an easy-to-use Graphical User Interface (GUI). Specifically, the Logic Learning Machine (LLM) is a method of supervised analysis based on an efficient implementation of the Switching Neural Network [10] and Monotone Boolean Reconstruction [11] through the Shadow Clustering technique [12] and is able to extract intelligible rules from data.

Rulex, through its GUI, also allows you to choose and apply other standard ML Algorithms to perform predictions. Through the interface, it is also possible to manipulate and filter data before applying forecasts using the same software package. Rulex [4] is natively supported on Windows 64 Bits PC's, where it operates in a standalone setup, storing its workflows on a local database.

2.3. Machine Learning and IoT Systems

As mentioned in the previous section, IoT is being applied to many applications and systems. For example, authors in [13] propose and edge computing framework for collaboration among nodes with the aim to improve resources management and achieve optimal offloading directed towards healthcare systems. Also, energy consumption on the edge and the used of ML to improve its performance is addressed in [14–16], since energy consumption is essential during ML forecasts due the limited power supplies available for light-weight IoT devices. Furthermore, authors in [17] apply ML algorithms for an indoor classification applications which uses features collected from radio frequency measurements. Also, ML forecasts are applied on time-series in [18] to predict failure on a slitting machine by relying on data collected by IoT sensors. Additionally, ML is used to secure IoT networks in [19,20] while improving systems security using Neural Networks.

Therefore, due to the vast scope of application of ML on the edge including its possible use in energy, healthcare, security, and resource allocation, the main purpose of this paper was to deploy a fully featured ML package on the edge to expand its services to the field of IoT.

3. Porting Techniques and Tools

The fundamental task of this project is the porting of the Rulex from 64 bit to 32-bit platforms. In the case of Windows, the Visual Studio environment to accomplish this task, however, in the case of Raspbian 32-bits and Ubuntu 64-bits, CMake [21,22] was employed to compile the external and internal dependencies and port Rulex.

In Figure 1, a tree-based file structure is shown, which presents a set of libraries or dependencies that exist in a porting process. The header files contain functions that are called in Cpp files. These header files may depend on other header files, however, Cpp files cannot call a function unless the corresponding header file is included. The Cpp files generate their output files which produce an overall output binary file. A binary file contains a compiled or encrypted version of the functions found in a header file.

Consequently, binary files could depend on other binary files, where in general, the final target is an executable.

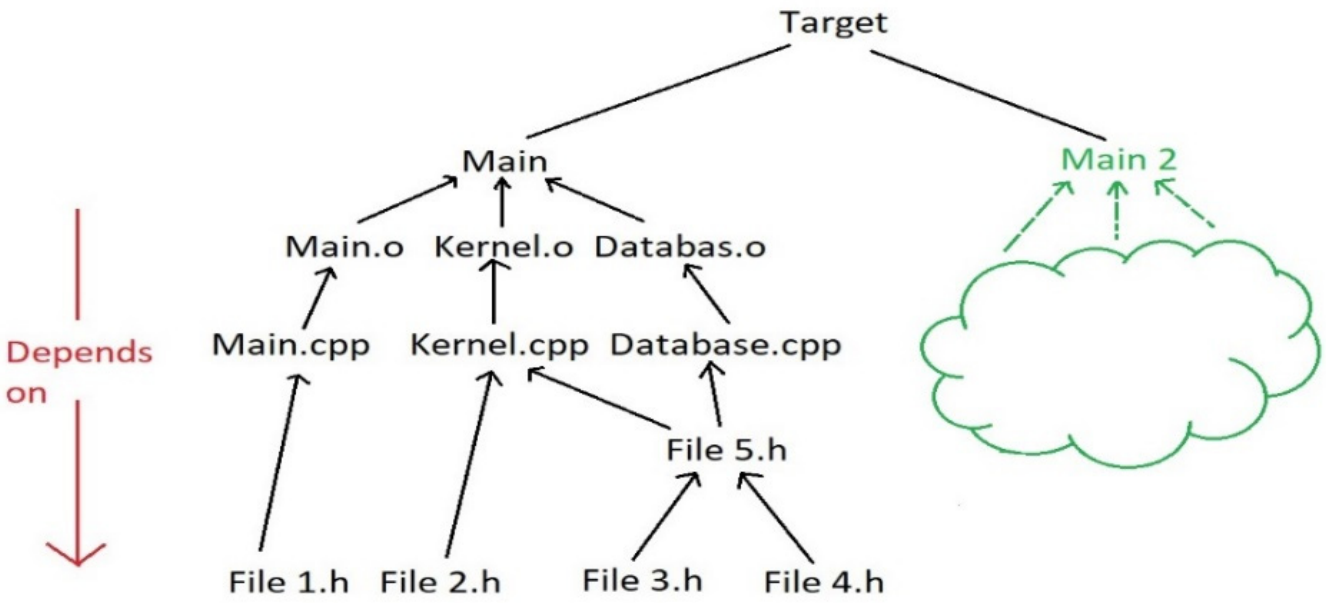

Figure 1. Dependencies file structure [3].

Rulex on the Raspberry Pi can operate in Standalone mode or Client/Server mode. In the latter case, a Windows 32-bit system is used as a client, where the Rulex GUI is running, and the Raspberry Pi functions as a server or ML engine. An SSH connection [23] is used in case the connection is over a public or private network. After connecting to the Rulex Engine on the Raspberry Pi, remote development was used to debug the code so it can operate on both Windows 32 Bits as well as Linux 32 Bit/ 64 Bit such that it runs without any manual modification.

In the process of testing, the software runs showed that some of the original C/C++ variables from the Windows 64-bit version are not compatible with 32 Bit systems. Therefore, it was necessary to make the source redundant concerning this issue. So, the flow of the code was diverted to a path or snippet specific to the running OS, which is implemented using Macros. For example, a Macros such as _WIN32 was used to detect a Windows 32-bits OS and _WIN64 for detecting Windows 64-bits. Also, to detect ARM, the __arm__ Macro was executed.

Rulex GUI source code was debugged through the SSH connection where a PC acts as the Client, and the Raspberry Pi forming the ML Server. In the Client/Server setup, a Docker-based PostgreSQL container [24] is placed as the common storage point between the Client and server nodes.

4. Forecast Results

To verify the effectiveness of the proposed solution, after porting Rulex to Windows 32 Bits as a Client, and Raspberry Pi as an application server, we tested Rulex on multiple datasets from five diverse applications having a different number of samples and with varying dimensions. Additionally, we've tested the accuracy of the Image-to-Timeseries pre-processing Algorithm on a gender classification dataset.

4.1. Radar Classification

The Urban Classification dataset was recorded by a short-range 24 GHz radar, based on the Infineon BGT24MTR11 RF transceiver [25], and particularly, the Distance2Go development kit by Infineon [26].

For ML forecasts, four classes were considered: One for *Humans* and three vehicle classes. Namely, Car, Truck, and Motorcycle where the dataset contains 120 records. In [27], a multiclass tree-based classification technique was implemented to improve the prediction accuracy using this dataset.

The radar data has been recorded using another separate system dedicated to feature extraction. This standalone system is composed of three parts. Firstly, A 24 GHz radar, a second Raspberry PI 3B+, and a PC running MATLAB. This second Raspberry PI was employed to connect the MATLAB station to the radar board. The Raspberry PI collects the data from the radar, and then it sends it to MATLAB running on a PC where feature extraction is applied [28,29].

Below is a list of the features extracted using the radar measurements:

1. R: The spread in the range-FFT spectrum caused by the target.
2. R_1: Variance of the range-FFT spectrum.
3. R_2: Standard deviation of the range-FFT spectrum.
4. R_3: Average of the range-FFT spectrum.
5. V: The spread in the Doppler-FFT spectrum caused by the target movement.
6. V_1: Variance of the Doppler-FFT spectrum.
7. V_2: Standard deviation of the Doppler-FFT spectrum.
8. V_3: Average of the Doppler-FFT spectrum.
9. RCS: Radar Cross-Section, that gives a measure for the reflectivity of the target.
10. V_{est}: The estimated speed of the target.

After extracting features using the second system, the first Client/Server setup which consists of Rulex running on a Raspberry PI is used for ML predictions. An example of a workflow in Rulex GUI running on the Raspberry is presented in Figure 2, where there are data processing blocks followed by an LLM, and finally a confusion matrix. Moreover, a block that splits data for training and testing is shown, where the ratio is 65% for training and 35% for testing.

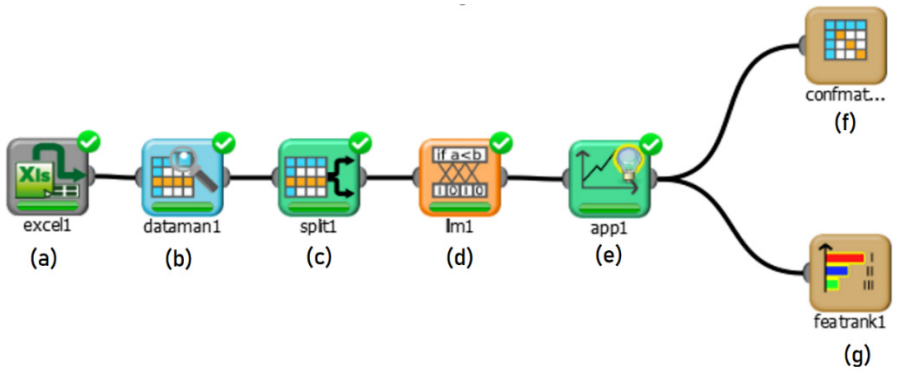

Figure 2. Rulex processing blocks: consisting of (**a**) excel1 which imports data, (**b**) dataman1 for viewing and data mining, (**c**) split1 for splitting dataset, (**d**) lm1 which performs the machine learning algorithm, (**e**) app1 to apply the model, and finally (**f**) confmatrix1 and (**g**) featrank1 which visually display results [3].

Figures 3 and 4 show the training and testing accuracies using LLM. In Figure 3, Cars and Humans are classified with a rate of 100%. As for Motorcycles and Trucks, they are 84.2% and 89.5% respectively. Figure 4 shows the testing accuracies where Cars are detected with a rate of 73.3%, Humans at 100%, Trucks at 72.7%, and Motorcycles were recognized with a rate of 90.9%.

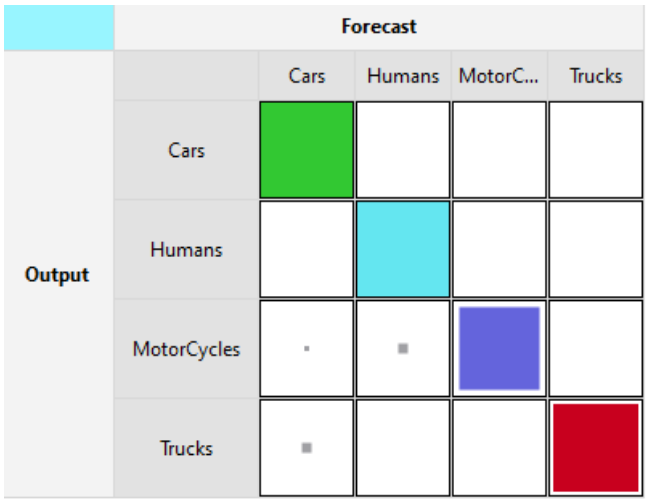

Figure 3. Radar training forecast [3].

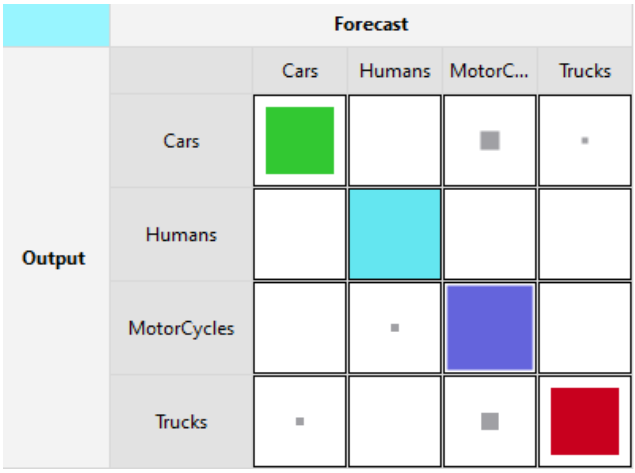

Figure 4. Radar testing forecast [3].

4.2. Human Activity Detection Using Smartphones

The Activity dataset from [30] consists of features that have been recorded using the Accelerometer and Gyroscope that are included in a Smartphone. The dataset is pre-processed such that it can be directly applied in Rulex since the features are composed of basic statistical operations based on the measurements. These consist of the mean, STD, variance, and others. The activity performed by a subject is divided into six classes: Laying, Standing, Sitting, Walking, Walking Upstairs, and Walking Downstairs.

A subset of the original dataset was used which is reduced to 7530 samples and 560 features, excluding the subject field which identifies the person or test subject specifically. When this field is included, the accuracy is increased considerably, however, it does not consider that the Smartphone can be carried by different people. This has been done to test the robustness of the ML forecasts and investigate the effectiveness of the present experiment. Furthermore, the dataset labels are distributed equally as shown in the histogram from Figure 5.

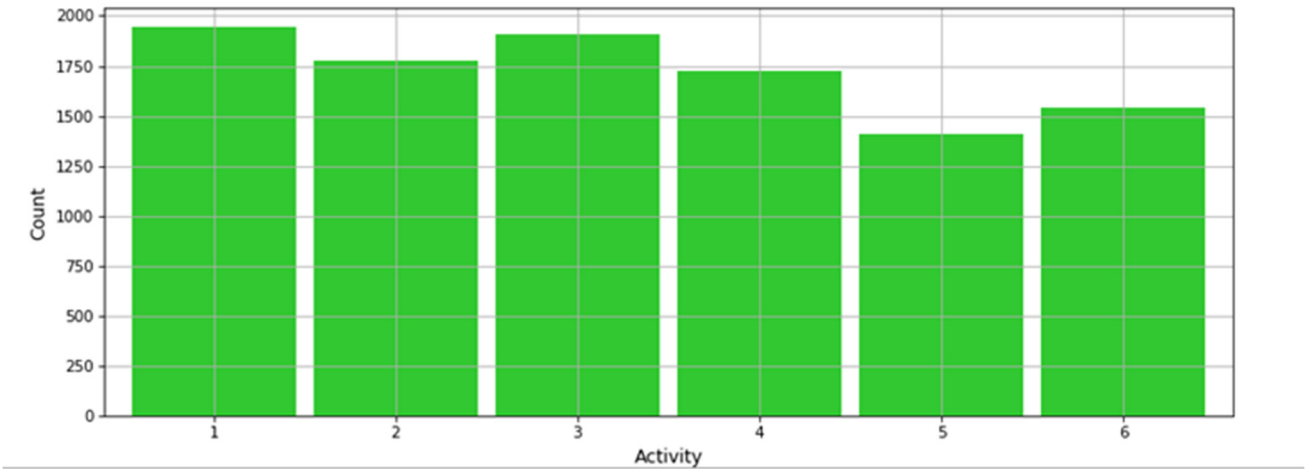

Figure 5. Low class skewness for the Human Activity Detection dataset.

The ML forecasts for the Smartphone Activity Detection dataset are provided by Table 1, wherein the Rulex Software, three basic ML Algorithms are implemented: LLM, K-Nearest-Neighbor (KNN), and SVM. All tests have been applied in a Client/Server arrangement with the Rulex Engine running on the Raspberry Pi. As shown in Table 1, high testing accuracy was reached in every forecast, where most notably, in the case where SVM is applied, a near-perfect accuracy is achieved.

Table 1. Forecast accuracy for the Smartphone Activity Detection application.

	Human Activity Classification with Rulex		
Algorithm	**LLM**	**KNN**	**SVM**
Laying	100%	100%	100%
Standing	86.8%	94.4%	98.4%
Sitting	88.48%	91.86%	97.63%
Walking	94.74%	100%	100%
Walking Upstairs	89.89%	99.47%	100%
Walking Downstairs	82.93%	100%	100%

4.3. Brainwave Mental State Classification

In [31], Electroencephalogram (EEG) recordings were used to predict the mental state of a subject through ML techniques. In this paper, the features from the concerned dataset are used in multiple forecasts to classify the mental into three given classes: Relaxed, Neural, and Concentrating states. Multiple Algorithms have been implemented consisting of the three basic ML Algorithms: SVM, Neural Networks and KNN, where SVM achieved the best performance. Figure 6 presents a plot of one frequency-based feature value vs. class labels, where this feature exhibits a lower value for one of the labels. Additionally, Figure 7 presents the same plot for another mean-based feature from the dataset, where results show that this feature has varying levels for each of the three classes. This illustrates the influence on every feature on each of the labels.

In this paper, we applied forecasts using Rulex running on the Raspberry Pi and in a Client/Server setup. Regarding the dataset, a total of 2480 samples and 988 features were employed for both training and testing in a 70/30 split. The accuracies for each forecast are presented in Table 2.

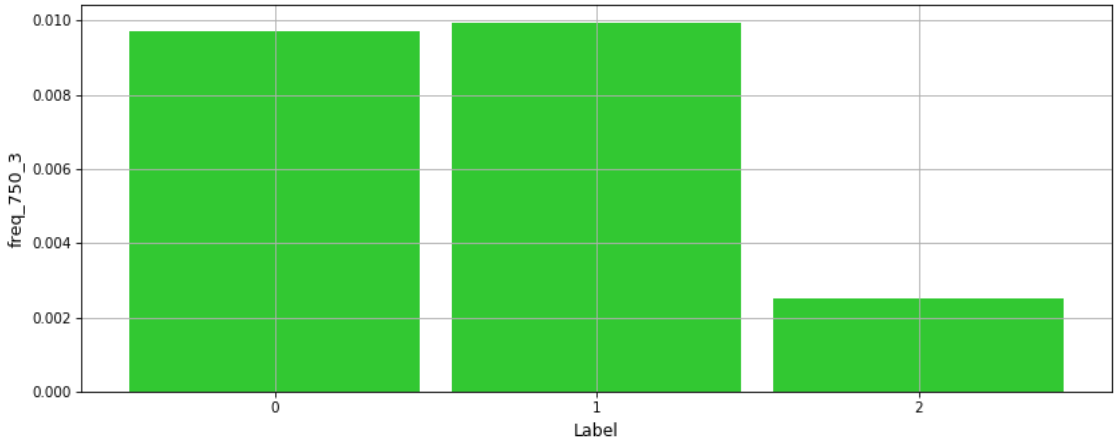

Figure 6. Distribution of frequency-based feature values vs. class labels.

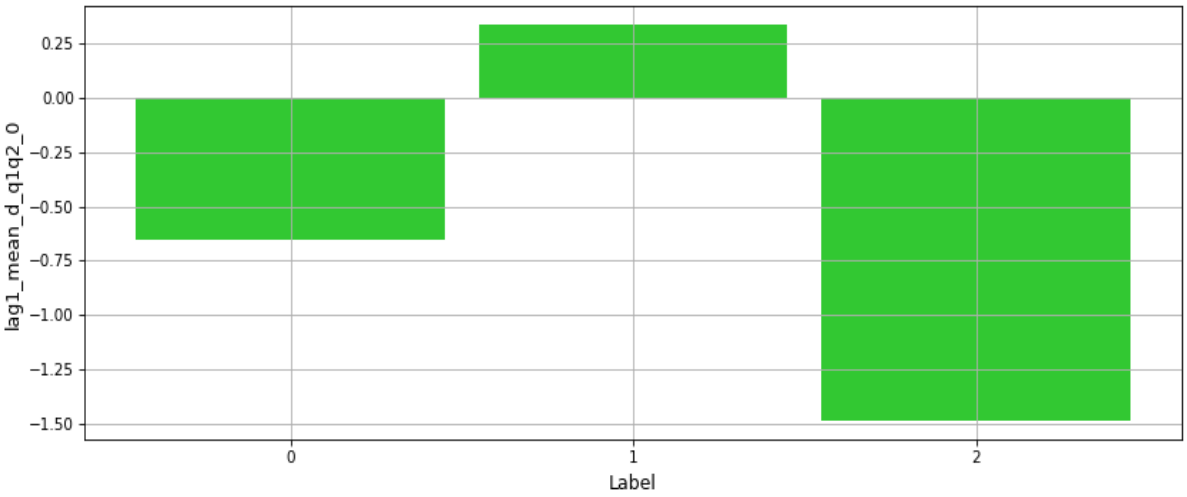

Figure 7. Distribution of mean-based feature value vs. class labels.

Table 2. Forecast accuracy for Mental State Classification.

Mental State Classification with Rulex			
Algorithm	**LLM**	**KNN**	**SVM**
Relaxed	92.77%	94.78%	96.39%
Neutral	76.02%	83.74%	89.84%
Concentrating	97.19%	100%	99.60%

As shown, LLM, KNN, SVM have high testing accuracies thus demonstrating the robustness of the experiment, where the most accurate forecasts were achieved using SVM with an overall accuracy of 95.47%.

In addition to applying forecasts on the edge for this dataset, we've studied the power consumption of the Raspberry Pi board by recording the elapsed time and estimating the power consumption, while comparing it with the energy consumed by an HP Laptop. The results can be viewed in Tables 3 and 4, whereas shown for LLM and SVM, the Raspberry Pi achieves more time but less energy to perform the same task.

Table 3. Power consumption for Mental State Classification for HP laptop.

Mental State Classification with Rulex Power Consumption		
Station	**LLM**	**SVM**
Time taken	0.27 min.	0.22 min.
Energy (70 Watts)	1120 J.	6230 J.

Table 4. Power consumption for Mental State Classification with the Raspberry Pi.

Mental State Classification with Rulex Power Consumption		
Station	**LLM**	**SVM**
Time taken	3.33 min.	0.98 min.
Energy (4 Watts)	800 J.	356 J.

4.4. Vehicle Activity Recognition

A dataset for Activity Recognition of Dumpers in earth moving sites is presented in [32]. It utilizes data taken from Smartphone sensors such as Gyroscopes and Accelerometers to record signals (while the Dumper is working) for feature extraction. An illustration of the data collection phase is presented in Figure 8, with the Smartphone installed inside the Dumper for taking measurements.

Figure 8. Taking sensor measurements using Smartphones to classify the state of a Dumper in an earth-moving site [32].

The pre-processed dataset was used to classify the state of the vehicle, whether a Dumper is in one of six states, namely: idle, driving, loading, dumping, engine-off, and unknown. Figure 9 illustrates the dispersal of the labels in the dataset where there is clear skewness in the class distribution. A subset of around 216,000 samples having just 8 features were used as a data source to apply ML training and testing with a 70/30 split. Forecasts were applied through Rulex running on the Raspberry Pi in a Client/Server arrangement using the KNN Algorithm. The testing accuracy in confusion matrix format

is presented in Figure 10, where the overall forecast accuracy using KNN on the Edge is 95.27%. Furthermore, with regards to power consumption, as employed in the previous dataset, the time elapsed to perform an ML task is recorded to estimate the consumed energy. In the case of the HP laptop which consumes around 70 W, as shown in Table 3, the time taken for KNN is 11.82 min and the energy is 49.63 KJ. As for the Raspberry Pi, where the power is 4 W, the time taken is 142.4 min and the energy consumed is 34.18 KJ.

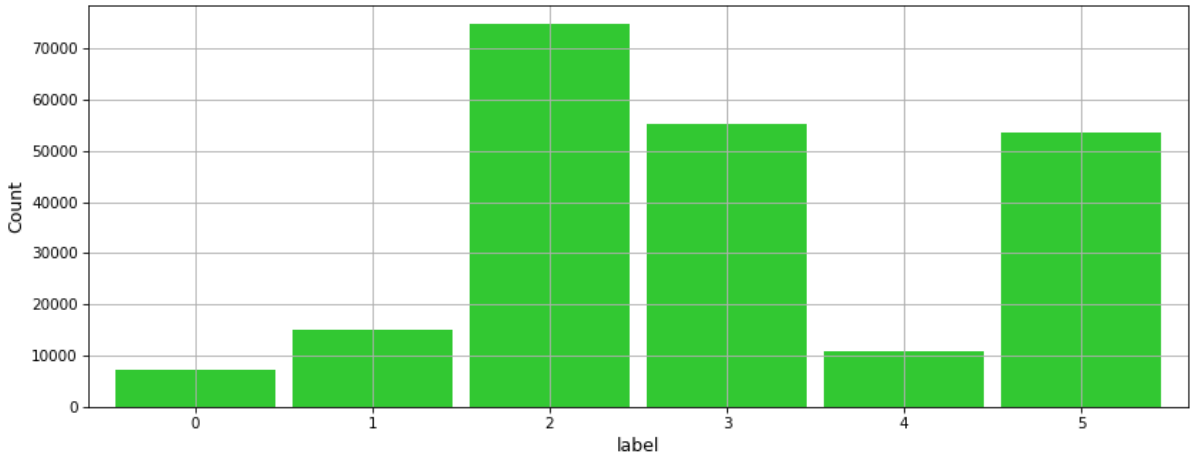

Figure 9. High class skewness for the Vehicle Activity Recognition dataset.

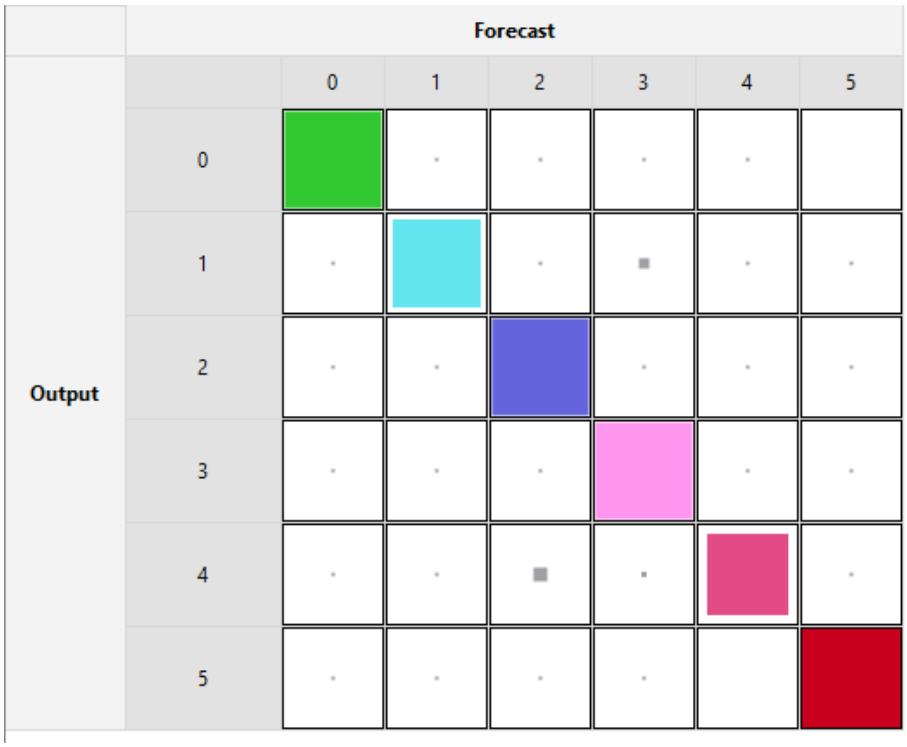

Figure 10. Testing Accuracy for Vehicle Activity Recognition using KNN.

4.5. Gender Classification

Gender classification can be useful for performing studies if implemented in an automated manner. In [33], gender detection using names, countries, and facial images is discussed where authors found that the accuracy is strongly dependent on the country, meaning that ethnicity can play a role in prediction accuracy.

Classification of Gender that is based on forward-facing images is reported in [34], where a minimum error rate of 2.85% is achieved. A Classifier implemented using Support Vector Machines (SVM) that detects gender with a minimum error of 3.4% is presented [35], where mug-shot images are used for training and testing.

In practice, when implementing gender classification, facial images can be misaligned in contrast to some forward-facing datasets that are seldom used in the literature. Therefore, authors in [36] apply the dropout technique and SVM to tackle this issue and classify age and gender under "in the wild " conditions.

In this paper, we've implemented a novel Algorithm that converts facial images from various ethnicities, poses, zoom levels, and ages into Time-series that are generated using a radial scanning-based approach [37]. Moreover, a Sobel filter [38] is used to detect the edges of an image and with the 360° radial scan, the distance from the center (which is also determined in Algorithm 1 till every visibly detected pixel is calculated. statistical functions of this distance for every degree in the scan are computed, before generating a Time-series for every function. Finally, the Time-series are grouped to form a dataset that is used in an ML forecast.

Algorithm 1: Extracting center points from sobel filter output

Input: *image dataset*
Output: *arrays sx, sy, centerx, centery*

```
for i in all images do:
    sobel = sobel(current image)
    sx [i], sy [i] = shape of sobel
    for positions of all horizontal points:
        sumx = sum of points where (pixel > threshold)
        if sumx > 0 then:
            meanx = meanx + position
            countx++
        end if
    end for
    centerx [i] = meanx/countx
    for positions of all vertical points:
        sumy = sum of points where (pixel > threshold)
        if sumy > 0 then:
            meany = meany + position
            county++
        end if
    end for
    centery [i] = meany/county
end for
Return sx, sy, centerx, centery
```

4.5.1. Image-to-Time Series Using Statistical Radial Scanning and Sobel Filters

Image Classification usually requires specialized ML Algorithms such as Convolutional Neural Networks (CNN) which are used for applications like object detection and face recognition [39]. Most applications using CNN's are dependent on the color of an image and its distribution. Although, some applications may not be so much reliant on color but rather the shape or contour of an image.

With a different approach, we developed a novel Algorithm aimed to perform feature extraction for these types of applications. Specifically, it has been designed to reduce the execution time, requiring less processing power compared to CNN which demands sufficient resources [40]. Furthermore, using this approach it is possible to apply the created Time-series to any ML Algorithm thus increasing the number of possible setups.

Firstly, as shown in the pseudo-code reported in Algorithm 1, a Sobel filter extracts the edges of each input image before determining the center points of Sobel output. Initially,

for every horizontal level, the sum of pixels having brightness greater than a predefined threshold is computed. Then, for each horizontal level, the mean (meanx) of the positions where the threshold was exceeded is computed. The value meanx corresponds to centrex, which is the center of filtered image. The same steps are followed to extract centery which is the vertical center of the same image.

In summary, Algorithm 1 calculates the vertical and horizontal means for every level to determine the image center's coordinates. Subsequently, As shown in Algorithm 2, radial scanning is performed on the Sobel filter's output. In Algorithm 2, the outer loop is used to scan the entire 360° around the center with coordinates (centerx, centery), and In case a pixel is brighter than a Threshold, the distance from the center is calculated, where statistical functions (mean, median, STD, variance, maximum, and minimum) are computed for the distances of every angle. Therefore, multiple corresponding Time-series are generated, however, with different lengths for each record. Finally, these Time-series are interpolated (To make each set of Time-series equal, and to have the same number of features for every record) and grouped in one file to form a dataset applicable for ML predictions. The pseudo-code from Algorithms 1 and 2 has been implemented in the Python programming language to perform the Image-to-Timeseries pre-processing before performing ML forecasts on the Edge using Rulex.

Algorithm 2: Statistical radial scanning of images

Input: *image dataset and arrays sx, sy, centerx, cenetry*
Output: *pre-processed dataset*

for *i in all images* ***do:***
  for *every angle y in sobel (0–90°)* ***do:***
    if *2nd iteration in loop or more* ***then:***
      rotate sobel by 90°
    end if
    if *y* < *45°* **then:**
      ratio = y/45
    end if
    if *y* > *45°* ***then:***
      ratio = 45/(90.01 − y)
    end if
    for *k = centery[i]; k < sy[i]; k = k + 1* ***do:***
      for *j = centerx[i]; j < sx[i]; j = j + 1* ***do:***
        if *k > (j * ratio) * 0.9* ***and*** *k < (j * ratio) * 1.1* ***then:***
          if *sobel(j, k) > treshsold* ***then:***
            append $\sqrt{(j - centerx[i])^2 + (k - centery[i])^2}$ to D
          end if
        end if
      end for
    end for
    "Add mean, median, STD, variance, maximum, and minimum of D to a Times-series"
  end for
end for
task: *Interpolate all six Time-series such that every unit (record) is expanded till the max length before grouping them into a dataset to apply ML forecasts.*

4.5.2. Gender Classification Experimental Results

To test the novel image pre-processing Algorithm described in the previous section, we considered a gender image classification dataset taken from [41]. It consists of two classes: Male and Female. The images are random where no specific angle is adopted as a reference, but rather the images of both classes have been taken from various inconsistent conditions in terms of angle, dimensions, zoom level, and background (cfr. Figure 11).

Figure 11. Six random images of women with varying ages, ethnicities, image background, dimensions, poses, and zoom level taken from the dataset found in [41].

Also, some of the photos contain part of the body of the subject. Furthermore, the people in the photos have very wide ranges of age, ranging from youthful to elderly, as well as belonging to different ethnicities while taking different poses. This is demonstrated in Figure 11, where an example of six random female subjects is shown.

The dataset consists of 2700 female photos and 2720 male photos. These photos were pre-processed using the novel algorithm developed in this paper to generate the corresponding dataset that consists of sets of Time-series.

The generated dataset has been used for ML forecasts using Rulex on the Raspberry Pi. KNN and SVM were used to perform training and testing with an 80/20 split, where the results for the forecasts performed on the Edge are presented in Table 5.

Table 5. Forecast accuracy for Gender Image Classification.

Gender Detection with Rulex		
Algorithm	**KNN**	**SVM**
Female	98.72%	95.60%
Male	94.24%	88.29%

In [34], the images used for gender classification were composed of people belonging to a single ethnicity using that are all forward-facing to the camera. A minimum error rate of 2.84% (accuracy of 97.16%) was achieved using SVM. Another case where SVM is used is reported in [35], where low-resolution forward-facing images are used to reach an error rate of 3.4% (96.6% accuracy). However, mug-shots or forward-facing photos may not be available in real-time practical situations, therefore, authors in [36] address this issue and attempt to determine the age and gender of random inconsistent images where the highest gender classification accuracy is 88.4%, (which is significantly lower than the publications where forward-facing photos were used) taken as the best from numerous forecasts.

As shown in Table 5, the Image-to-Timeseries conversion Algorithm is capable of pre-processing random images (Where color is not an issue) and can be applied to various ML Algorithms to achieve high gender detection accuracy. Consequently, it can compete

with previously published techniques that rely on impractical (consistent) mug-shot images for classification.

5. Conclusions

In this paper, we refer to the porting of Rulex, a machine learning software that natively runs on Windows 64 Bits, on the Raspberry Pi, for Edge Computing applications. We also reported the results obtained in different application domains, namely with five unrelated datasets, which we used to test the performance of our implementation in real IoT environments.

The main result obtained is that Rulex now operates in a Client/Server setup with the interface being operated on a PC as a Windows 32 Bits application, with the machine learning Algorithms being run on the Raspberry Pi ARM 32-Bit microcontroller.

As regards the performance obtainable through this implementation, first we considered a dataset related to the classification of pedestrians and vehicles using a high-frequency radar: An ML workflow was implemented on our platform and good results were achieved. However, as such dataset is relatively small, we considered four additional datasets from diverse application fields. Three pre-processed datasets having a larger number of samples and with low and high dimensionality and varying skewness were adopted. The pre-processed datasets include a Human activity Recognition dataset using Smartphones, a vehicle activity detection dataset, and an EEG Classification dataset related to mental state.

Furthermore, a novel pre-processing Algorithm was developed and implemented in Python, that converts images into Time-series to pre-process a gender detection dataset that contains inconsistent facial images in terms of the background and dimensions of the image itself and the age, ethnicity, and pose of the subject. Also, the accuracy achieved for gender classification is considerably competitive with the literature, where the competition is dominated by mug-shot and forward-facing images. Moreover, in every forecast, the training and testing were performed using Rulex on the Edge, where in general, each experiment achieved high classification accuracy through the Client/Server interface. Also, the power consumption was investigated by comparing the Raspberry Pi as an edge computing node with an HP laptop, where for the same ML Algorithm and dataset, the Raspberry Pi consumes less energy.

Author Contributions: A.W.D. developed and implemented the Image-to-Timeseries pre-processing Algorithm. A.W.D. applied machine learning Forecasts. A.W.D. Wrote the main draft. A.W.D. and M.M. ported Rulex onto the Raspberry Pi. A.R. collected the Radar data. A.W.D., A.R., H.C., M.M. and D.D.C. discussed the results and revised the manuscript. 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 Human Activity Detection Dataset using Smartphones can be found in the Kaggle online repository through the following address: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones. The Mental Sate Detection Dataset can be found in the Kaggle online repository through the following address: https://www.kaggle.com/birdy654/eeg-brainwave-dataset-mental-state. The Vehicle Activity Recognition Dataset can be found in the Kaggle online repository through the following address: https://www.kaggle.com/smartilizer/commercial-vehicles-sensor-data-set. The image-based Gender Classification Dataset can be found in the Kaggle online repository through the following address: https://kaggle.com/ashwingupta3012/male-and-female-faces-dataset.

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

References

1. Premsankar, G.; Di Francesco, M.; Taleb, T. Edge Computing for the Internet of Things: A Case Study. *IEEE Internet Things J.* **2018**, *5*, 1275–1284. [CrossRef]
2. Yu, W.; Liang, F.; He, X.; Hatcher, W.G.; Lu, C.; Lin, J.; Yang, X. A Survey on the Edge Computing for the Internet of Things. *IEEE Access* **2017**, *6*, 6900–6919. [CrossRef]
3. Daher, A.W.; Rizik, A.; Muselli, M.; Chible, H.; Caviglia, D.D. Porting Rulex Machine Learning Software to the Raspberry Pi as an Edge Computing Device. In *Applications in Electronics Pervading Industry, Environment and Society*; Springer: Cham, Switzerland, 2021; pp. 273–279. [CrossRef]
4. Vujovic, V.; Maksimovic, M. Raspberry Pi as a Wireless Sensor node: Performances and constraints. In Proceedings of the 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 26–30 May 2014; pp. 1013–1018. [CrossRef]
5. Maksimovic, M.; Vujovic, V.; Davidović, N.; Milosevic, V.; Perisic, B. Raspberry Pi as Internet of Things hardware: Performances and Constraints. In Proceedings of the IcETRAN, Vrnjacka Banja, Serbia, 2–5 June 2014.
6. Muselli, M. Extracting knowledge from biomedical data through Logic Learning Machines and Rulex. *EMBnet J.* **2012**, *18*, 56. [CrossRef]
7. Rulex AI. Rulex, the Platform for Data-Driven Decisions. Available online: https://www.rulex.ai/ (accessed on 16 September 2021).
8. Hajdarevic, K.; Konjicija, S.; Subasi, A. A low energy APRS-IS client-server infrastructure implementation using Raspberry Pi. In Proceedings of the 2014 22nd Telecommunications Forum Telfor (TELFOR), Belgrade, Serbia, 25–27 November 2014; pp. 296–299. [CrossRef]
9. Parodi, S.; Manneschi, C.; Verda, D.; Ferrari, E.; Muselli, M. Logic Learning Machine and standard supervised methods for Hodgkin's lymphoma prognosis using gene expression data and clinical variables. *Health Inform. J.* **2016**, *24*, 54–65. [CrossRef] [PubMed]
10. Muselli, M. Switching Neural Networks: A New Connectionist Model for Classification. In *Neural Nets*; Springer: Berlin, Germany, 2006; pp. 23–30. [CrossRef]
11. Muselli, M.; Quarati, A. Reconstructing positive Boolean functions with shadow clustering. In Proceedings of the 2005 European Conference on Circuit Theory and Design, Cork, Ireland, 29 August–1 September 2005. [CrossRef]
12. Muselli, M.; Ferrari, E. Coupling Logical Analysis of Data and Shadow Clustering for Partially Defined Positive Boolean Function Reconstruction. *IEEE Trans. Knowl. Data Eng.* **2009**, *23*, 37–50. [CrossRef]
13. Al-Khafajiy, M.; Webster, L.; Baker, T.; Waraich, A. Towards fog driven IoT healthcare: Challenges and framework of fog computing in healthcare. In Proceedings of the ICFNDS'18: International Conference on Future Networks and Distributed Systems, Amman, Jordan, 26–27 June 2018; Volume 9, pp. 1–7. [CrossRef]
14. Cecilia, J.; Cano, J.-C.; Morales-García, J.; Llanes, A.; Imbernón, B. Evaluation of Clustering Algorithms on GPU-Based Edge Computing Platforms. *Sensors* **2020**, *20*, 6335. [CrossRef] [PubMed]
15. Lapegna, M.; Balzano, W.; Meyer, N.; Romano, D. Clustering Algorithms on Low-Power and High-Performance Devices for Edge Computing Environments. *Sensors* **2021**, *21*, 5395. [CrossRef] [PubMed]
16. Novac, P.-E.; Hacene, G.B.; Pegatoquet, A.; Miramond, B.; Gripon, V. Quantization and Deployment of Deep Neural Networks on Microcontrollers. *Sensors* **2021**, *21*, 2984. [CrossRef] [PubMed]
17. Classification of Indoor Environments for IoT Applications: A Machine Learning Approach. Available online: https://ieeexplore.ieee.org/abstract/document/8458184 (accessed on 9 September 2021).
18. Kanawaday, A.; Sane, A. Machine learning for predictive maintenance of industrial machines using IoT sensor data. In Proceedings of the 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 24–36 November 2017; pp. 87–90. [CrossRef]
19. Canedo, J.; Skjellum, A. Using machine learning to secure IoT systems. In Proceedings of the 2016 14th Annual Conference on Privacy, Security and Trust (PST), Auckland, New Zealand, 12–14 December 2016; pp. 219–222. [CrossRef]
20. Hodo, E.; Bellekens, X.; Hamilton, A.; Dubouilh, P.-L.; Iorkyase, E.; Tachtatzis, C.; Atkinson, R. Threat analysis of IoT networks using artificial neural network intrusion detection system. In Proceedings of the 2016 International Symposium on Networks, Computers and Communications (ISNCC), Yasmine Hammamet, Tunisia, 11–13 May 2016; pp. 1–6. [CrossRef]
21. Clemencic, M.; Mato, P. A CMake-based build and configuration framework. *J. Phys. Conf. Ser.* **2012**, *396*, 052021. [CrossRef]
22. Fober, D.; Orlarey, Y.; Letz, S. Building Faust with CMake. Available online: https://hal.archives-ouvertes.fr/hal-02158978 (accessed on 16 September 2021).
23. Linux Journal. Eleven SSH Tricks. Available online: https://dl.acm.org/doi/abs/10.5555/860397.860402 (accessed on 11 May 2021).
24. Bellavista, P.; Zanni, A. Feasibility of Fog Computing Deployment based on Docker Containerization over RaspberryPi. In Proceedings of the 18th International Conference on Distributed Computing and Networking, Hyderabad, India, 5–7 January 2017. [CrossRef]
25. Infineon Technologies. BGT24MTR11. Available online: https://www.infineon.com/cms/en/product/sensor/radar-sensors/radar-sensors-for-iot/24ghz-radar/bgt24mtr11/ (accessed on 11 May 2021).
26. Infineon Technologies. DEMO DISTANCE2GO. Available online: https://www.infineon.com/cms/en/product/evaluation-boards/demo-distance2go/ (accessed on 11 May 2021).

27. Daher, A.W.; Rizik, A.; Randazzo, A.; Tavanti, E.; Chible, H.; Muselli, M.; Caviglia, D.D. Pedestrian and Multi-Class Vehicle Classification in Radar Systems Using Rulex Software on the Raspberry Pi. *Appl. Sci.* **2020**, *10*, 9113. [CrossRef]
28. Rizik, A.; Randazzo, A.; Vio, R.; Delucchi, A.; Chible, H.; Caviglia, D.D. Feature Extraction for Human-Vehicle Classification in FMCW Radar. In Proceedings of the 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Genova, Italy, 27–39 November 2019; pp. 131–132. [CrossRef]
29. Rizik, A.; Tavanti, E.; Chible, H.; Caviglia, D.D.; Randazzo, A. Cost-Efficient FMCW Radar for Multi-Target Classification in Security Gate Monitoring. *IEEE Sens. J.* **2021**, *21*, 20447–20461. [CrossRef]
30. Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; Reyes-Ortiz, J.L. Public Domain Dataset for Human Activity Recognition Using Smartphones. In Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 24–36 April 2013.
31. Bird, J.J.; Manso, L.J.; Ribeiro, E.P.; Ekart, A.; Faria, D.R. A Study on Mental State Classification using EEG-based Brain-Machine Interface. In Proceedings of the 2018 International Conference on Intelligent Systems (IS), Funchal, Portugal, 25–27 September 2018; pp. 795–800. [CrossRef]
32. Axelsson, H.; Wass, D. Machine Learning for Activity Recognition of Dumpers 2019. Available online: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-260256 (accessed on 14 May 2021).
33. Karimi, F.; Wagner, C.; Lemmerich, F.; Jadidi, M.; Strohmaier, M. Inferring Gender from Names on the Web: A Comparative Evaluation of Gender Detection Methods. In Proceedings of the 25th International Conference Companion on World Wide Web, Republic and Canton of Geneva, CHE, Montréal, QC, Canada, 11–15 April 2016; pp. 53–54. [CrossRef]
34. Yang, Z.; Li, M.; Ai, H. An Experimental Study on Automatic Face Gender Classification. In Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06), Hong Kong, China, 20–24 August 2006; Volume 3, pp. 1099–1102. [CrossRef]
35. Moghaddam, B.; Yang, M.-H. Gender classification with support vector machines. In Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580), Grenoble, France, 26–30 March 2002. [CrossRef]
36. Eidinger, E.; Enbar, R.; Hassner, T. Age and Gender Estimation of Unfiltered Faces. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 2170–2179. [CrossRef]
37. Converting Images into Time Series for Data Mining. Available online: https://izbicki.me/blog/converting-images-into-time-series-for-data-mining (accessed on 24 July 2021).
38. Bora, D. A Novel Approach for Color Image Edge Detection Using Multidirectional Sobel Filter on HSV Color Space. *Int. J. Comput. Sci. Eng.* **2017**, *5*, 154–159.
39. Levi, G.; Hassner, T. Age and Gender Classification Using Convolutional Neural Networks. Available online: https://www.cv-foundation.org/openaccess/content_cvpr_workshops_2015/W08/html/Levi_Age_and_Gender_2015_CVPR_paper.html (accessed on 22 July 2021).
40. Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a convolutional neural network. In Proceedings of the 2017 International Conference on Engineering and Technology (ICET), Antalya, Turkey, 21–23 August 2017; pp. 1–6. [CrossRef]
41. Male and Female Faces Dataset. Available online: https://kaggle.com/ashwingupta3012/male-and-female-faces-dataset (accessed on 1 August 2021).