sensors

MDPI

Article

A Smart Autonomous Time- and Frequency-Domain Analysis Current Sensor-Based Power Meter Prototype Developed over Fog-Cloud Analytics for Demand-Side Management

Yung-Yao Chen [1] and **Yu-Hsiu Lin** [2,*]

[1] Graduate Institute of Automation Technology, National Taipei University of Technology, Taipei 106, Taiwan; yungyaochen@mail.ntut.edu.tw

[2] Department of Electrical Engineering, Ming Chi University of Technology, New Taipei City 24301, Taiwan

[*] Correspondence: yhlin@mail.mcut.edu.tw; Tel.: +886-2-2908-9899 (ext. 4829)

Received: 3 September 2019; Accepted: 12 October 2019; Published: 14 October 2019

Abstract: Electrical energy management, or demand-side management (DSM), in a smart grid is very important for electrical energy savings. With the high penetration rate of the Internet of Things (IoT) paradigm in modern society, IoT-oriented electrical energy management systems (EMSs) in DSM are capable of skillfully monitoring the energy consumption of electrical appliances. While many of today's IoT devices used in EMSs take advantage of cloud analytics, IoT manufacturers and application developers are devoting themselves to novel IoT devices developed at the edge of the Internet. In this study, a smart autonomous time and frequency analysis current sensor-based power meter prototype, a novel IoT end device, in an edge analytics-based artificial intelligence (AI) across IoT (AIoT) architecture launched with cloud analytics is developed. The prototype has assembled hardware and software to be developed over fog-cloud analytics for DSM in a smart grid. Advanced AI well trained offline in cloud analytics is autonomously and automatically deployed onsite on the prototype as edge analytics at the edge of the Internet for online load identification in DSM. In this study, auto-labeling, or online load identification, of electrical appliances monitored by the developed prototype in the launched edge analytics-based AIoT architecture is experimentally demonstrated. As the proof-of-concept demonstration of the prototype shows, the methodology in this study is feasible and workable.

Keywords: artificial intelligence; demand-side management; fog-cloud analytics; Industry 4.0; internet of things; machine learning; smart grid; smart homes

1. Introduction

Electricity is one of the most commonly used forms of energy in modern society. Consumer electrical energy demands are continuously increasing. During the last few decades rapid growth of Internet of Things (IoT) technologies has occurred. Modern IoT technologies have been developed and implemented for smart grids. A smart grid exploiting modern IoT technologies for a more efficient, reliable, and flexible traditional power grid upgraded in this revolution can be regarded as the key enabler that makes existing cities ready for tomorrow's needs, including continuously increasing demands for electrical energy. A smart grid improves two-way communication between power utilities and consumers and opens new technical challenges, such as intelligent sensing and decision making. In a smart grid exploiting modern IoT technologies, to redefine the role of end users as power producers and the role of power utilities as service providers, consumers are "prosumers," which allows them to generate, store, and sell their own electrical energy. Demand-side management (DSM) is very attractive and the most promising enabling technology of smart grids developed by power utilities and adopted to entice consumers to actively participate in demand response (DR) schemes [1–4]. DSM refers to initiatives [5] and technologies that encourage consumers to optimize their electrical energy consumption such that: (1) electrical energy requests from downstream sectors of a smart grid can be met; (2) the efficiency, reliability, and flexibility of the traditional power grid can be upgraded and enhanced; and (3) greenhouse gas emissions can be abated. Effective DSM alleviates such continuously increasing electrical energy demands. The benefits of participating in DSM are potentially twofold. First, consumers can reduce their electricity bills by adjusting their electrical energy consumption. Second, the power grid will adopt distributed energy resources, such as renewables and electric vehicles, which forces power utilities to rethink how the power grid can work intelligently and beneficially by shifting electrical energy consumption from peak hours to non-peak hours, in order to meet consumers' electrical energy demands with flexible load shapes [6].

Smart energy management systems (EMSs) such as a home EMS, which have been considered as an integral part of successful DSM in a smart grid [7], can contribute to cutting electricity production costs by power utilities while still meeting electrical energy demands from downstream sectors of the smart grid. Many recent central EMSs [8–12] for DSM in a smart grid produce huge amounts of IoT data and transmit the data to resource-rich centralized data centers for data science analytics. However, networks may be congested, and centralized data centers and cloud analytics may be overloaded [13]. In addition, in cases where fast response with low communication overhead is required, user-centric IoT service-oriented EMS applications are network latency-sensitive and require real-time responsiveness and local interpretable and actionable data science analytics for rapid decision making (in case of immediate and adverse events) [14,15].

Centralized cloud analytics usually cannot remedy such drawbacks and fill such requirements [15]. Also, IoT and artificial intelligence (AI) (aka machine learning) have joined together to be envisioned as the next wave in the era of the IoT; IoT devices need informative AI models learned from IoT data in cloud analytics and used as edge analytics to provide real-time and onsite actionable data insights to customers [16,17]. Edge analytics and fog analytics, instead of cloud analytics, move cloud analytics close to IoT end devices to minimize network latency and enhance location awareness [18,19]. Not much attention has been paid to developing an EMS architecture over fog-cloud analytics for DSM in a smart grid. Thus, in this study, a smart autonomous time and frequency analysis current sensor-based power meter prototype, a novel IoT end device compared in [20–23], in an edge analytics-based AI across IoT (AIoT) architecture launched with an open and powerful cloud analytics platform is designed and implemented, which has assembled hardware and software to be developed over fog-cloud analytics for DSM in a smart grid. Fog-cloud analytics, which is considered for DSM in this study, concerned in a smart grid, and introduced conceptually in [24], concurrently leverages resources from cloud-centered data science analytics (advanced AI) to edge connectivity (advanced AI autonomously and automatically deployed onsite on the developed prototype via the Internet). AI well trained offline in cloud analytics and autonomously and automatically deployed as edge analytics on IoT end devices via the Internet was taken into account and developed for IoT applications in [15,21,22]. For a practical application of DSM in this study, auto-labeling, or load identification as nonintrusive load monitoring [8,10,25–28], of electrical appliances monitored by the developed prototype is experimentally investigated.

A summary of this study is as follows:

- A smart autonomous time- and frequency-domain analysis current sensor-based power meter prototype, which co-operates with cloud analytics (AI trained offline in cloud analytics) for collaborative learning and serves as edge analytics to identify electrical appliances onsite and online over fog-cloud analytics for DSM in a smart grid (well-trained AI autonomously and automatically deployed onsite on the prototype via the Internet (representational state transfer (REST) application programming interface (API))), is presented in this study. The prototype presented in our previous work [23] is upgraded in this study, which was based on time-domain feature extraction and was not autonomously and automatically collaborated with cloud analytics.

The upgraded prototype in this study is a novel IoT end (edge) device, as compared with the work done in [20–23]. The prototype developed over fog-cloud analytics and presented in this study is experimentally examined. The prototype has assembled hardware and software to be developed.

- An open and powerful cloud analytics platform is configured and used as a data science analytics engine without developing custom web software for DSM to implement lightweight AI trained offline in cloud analytics and deployed, via the Internet (REST API), onsite on the prototype as edge analytics for online load identification in the presented fog-cloud analytics-based AIoT architecture.
- In AI, a model such as an artificial neural network (ANN) trying too hard to respond to noisy data in a training dataset should avoid overfitting. Regularization is one of the most important techniques used in AI to overcome overfitting. A radial basis function ANN (RBF-ANN) that takes a regularization risk function into account is employed in this study as lightweight AI. It has the low computational complexity of AI (lightweight AI) implemented in cloud analytics and deployed via the Internet (REST API) onsite on the presented prototype as edge analytics autonomously and automatically for online load identification in DSM. The RBF-ANN is a typical feed-forward ANN, which has the merits of (1) a simple network structure and (2) conspicuous fast and high learning and generalization performance.

This study is organized as follows: The developed prototype is presented in Section 2. Section 3 shows a proof-of-concept demonstration of the prototype developed over fog-cloud analytics for DSM, where the prototype is experimentally validated in a realistic household environment. Finally, this study is concluded in Section 4.

2. Methodology: Smart Autonomous Time and Frequency Analysis Current Sensor-Based Power Meter Prototype in an Edge Analytics-Based AIoT Architecture Launched with an Open Cloud Analytics Platform

This section presents the developed smart autonomous time and frequency analysis current sensor-based power meter prototype in an edge analytics-based AIoT architecture launched with an open and powerful cloud analytics platform, which is designed and implemented over fog-cloud analytics for DSM. Fog-cloud analytics is crucial for user-centric IoT applications in DSM in a smart grid, which is network latency-sensitive and requires real-time and onsite data science analytics for immediate decision making. Figure 1 depicts a generic three-tier architecture of fog-cloud analytics consisting of an IoT device layer, a fog layer, and a cloud layer [29,30]. In such a generic architecture, IoT data sensed and gathered in an IoT device layer are exchanged with a fog layer that allows for communication of IoT end edge devices, including edge gateways (fog devices can be gateways [31]), with a powerful cloud analytics platform. An IoT device layer is the basis of a generic architecture of fog-cloud analytics. IoT end edge devices are the main entities of an IoT device layer, and they can be deployed over a wide area of interest for user-centric IoT applications. A configured fog layer is used as a technical bridge between an IoT device layer and a cloud layer to process gathered IoT data through data science analytics at IoT data sources and exchange processed IoT data via IoT communication technology [32]. A fog layer is made of edge gateways that respond rapidly to IoT end edge devices in an IoT device layer, and IoT end edge devices, including edge gateways, avoid network latency over the Internet because of their proximity to them. A cloud layer, which can be distributed in cloud storage and user-centric IoT applications, is responsible for storing gathered and processed IoT data and providing collaborative user-centric IoT services to customers. In a generic architecture of fog-cloud analytics, IoT end edge devices, including edge gateways, in a fog layer gather processed IoT data from IoT end edge devices in an IoT device layer and pass them to a cloud layer. Mukherjee et al. in [33] conducted a comprehensive survey relating to fog-cloud analytics-based IoT architectures; interested researchers can refer to the survey article. This study develops an AIoT architecture based on fog-cloud analytics for DSM in a smart grid, which is similar to the generic architecture depicted in Figure 1. Figure 2 shows a conceptual vision of the developed smart autonomous power meter prototype as a new edge computing device [20–23] considering time and frequency analysis and advanced AI in a future smart sensing infrastructure based on fog-cloud analytics for DSM in a smart grid [23]. Future advances, (1) user-centric IoT service-oriented single and/or multiple sensing modalities and applications and (2) fault detection and classification in industry 4.0 (smart factories), for DSM in a smart grid can be built upon. As cloud analytics applied to gathered IoT data that need to be processed for real-time and onsite actionable data insights at IoT data sources is based on network connectivity that is not always available or is limited, edge (fog) analytics is a promising technique dedicated to data science analytics enabled directly at the edge of the network. In this sense, edge analytics extends cloud analytics and covers its shortage.

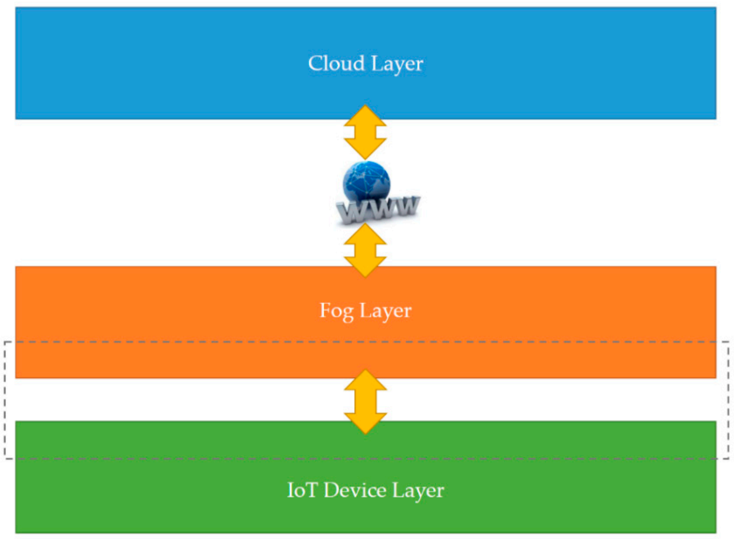


Figure 1. Generic three-tier architecture of fog-cloud analytics. IoT = Internet of Things.

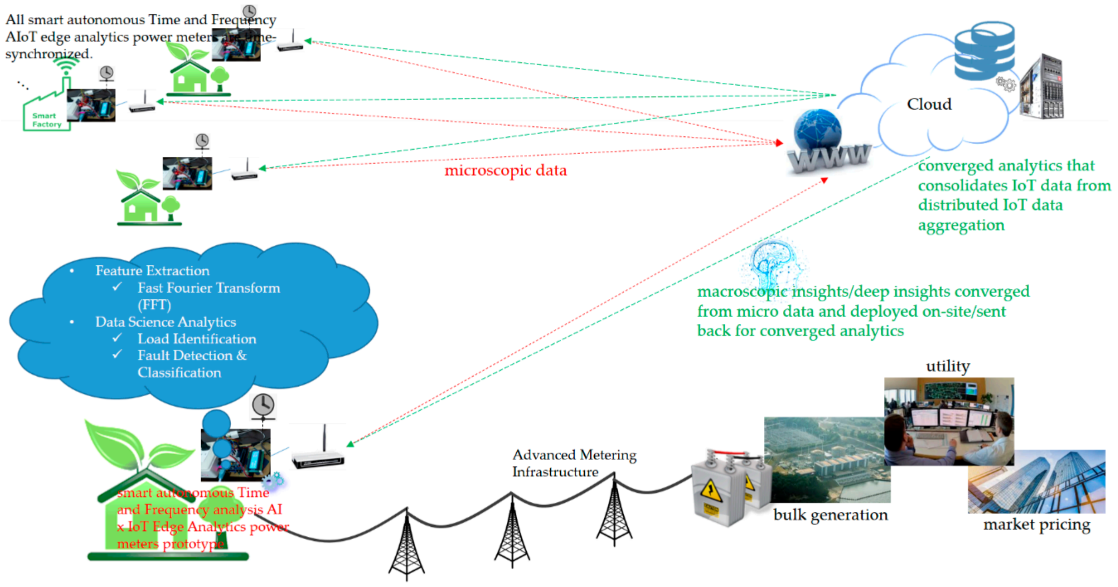

All smart autonomous Time and Frequency AIoT edge analytics power meters are time-synchronized.
Smart Factory
microscopic data
Cloud
WWW
converged analytics that consolidates IoT data from distributed IoT data aggregation
macroscopic insights/deep insights converged from micro data and deployed on-site/sent back for converged analytics
Feature Extraction
Fast Fourier Transform (FFT)
Data Science Analytics
Load Identification
Fault Detection & Classification
smart autonomous Time and Frequency analysis AI x IoT Edge Analytics power meters prototype
Advanced Metering Infrastructure
utility
bulk generation
market pricing

Figure 2. Conceptual vision of the smart autonomous power meter prototype considering time- and frequency-domain analysis (spectral analysis) and advanced artificial intelligence (AI) in a future smart sensing infrastructure based on fog-cloud analytics for demand-side management (DSM) in a smart grid.

Fog-cloud analytics serves as converged analytics that consolidates IoT data from distributed IoT data aggregation. Based on fog-cloud analytics developed for user-centric IoT applications in smart homes, manufacturers, and cities considering DSM in a smart grid, as shown in Figure 2, the developed prototype, which is installed in multiple fields of interest in a smart grid, time-synchronized, and autonomously and automatically deployed onsite as edge analytics with advanced AI (trained offline in cloud analytics and ported on the prototype via the Internet), is able to perform converged analytics onsite and online. Converged analytics consolidates IoT data from distributed IoT data aggregation. The smart autonomous power meter prototype presented in this section and developed over fog-cloud analytics for DSM in this study is a preliminary design toward such a scenario as a next-generation AMI (smart sensing) infrastructure expected for smart homes, manufacturers, and cities.

2.1. Architectural Design of an Edge Analytics-Based AIoT Considering the Developed Smart Autonomous Time and Frequency Analysis Current Sensor-Based Power Meter Prototype with Open Cloud Analytics for DSM in a Smart Grid

Figure 3 illustrates the architecture of the prototype designed and implemented in this study, which is a version of a single-home demonstration of the generic three-tier fog-cloud analytics architecture depicted in Figure 1. In Figure 3, the fog-cloud analytics architecture is implemented in a home in Taiwan. The residential environment is equipped with an energy management controller that manages home appliances in response to DR signals and is connected via an advanced metering infrastructure to a smart grid that comes up with DR programs in which a power company-owned smart meter instead of a traditional wattmeter is installed and used to transmit electrical energy consumption data records to and receive DR signals from a power utility for DSM. One of the most important functionalities of a smart grid is self-decision-making ability [34]. To enable this functionality, DR, a viable approach to motivate consumers toward shifting their electrical energy demand during peak load periods [35], emerges. As electrical appliances monitored in the residential environment in Figure 3 are identified and reacted with DR signals, the developed prototype in this study is used to identify them based on time- and frequency-domain analysis.

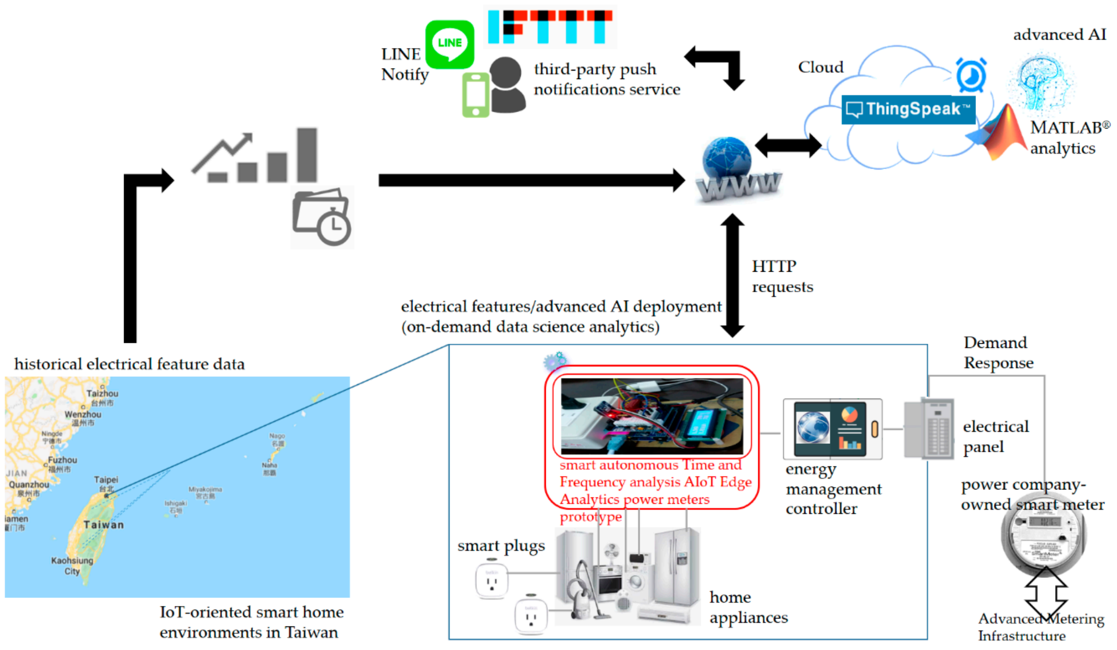

LINE Notify
third-party push notifications service
Cloud
advanced AI
ThingSpeak
MATLAB® analytics
HTTP requests
electrical features/advanced AI deployment (on-demand data science analytics)
historical electrical feature data
Demand Response
electrical panel
smart autonomous Time and Frequency analysis AIoT Edge Analytics power meters prototype
energy management controller
power company-owned smart meter
smart plugs
home appliances
Advanced Metering Infrastructure
IoT-oriented smart home environments in Taiwan
Taipei
Taiwan
Kaohsiung City

Figure 3. Launched edge analytics-based artificial intelligence across Internet of Things (AIoT) with open and powerful cloud analytics configured to collaborate with the developed smart autonomous time and frequency analysis current sensor-based power meter prototype as edge analytics over fog-cloud analytics for DSM in a smart grid in this study.

The developed smart autonomous power meter prototype in the residential environment in a smart grid is time-synchronized and autonomously and automatically deployed onsite as edge analytics, which is designed and implemented to interact with an open and powerful cloud analytics platform configured for advanced AI trained offline in cloud analytics and ported on the prototype via the Internet.

The cloud analytics platform configured to collaborate with the prototype over fog-cloud analytics is based on ThingSpeak™ [36] with MATLAB® analytics, which can run MATLAB® codes on demand in the cloud. In the architecture, push notifications service is also developed. The push notifications service provided in the architecture in Figure 3, If-This-Then-That (IFTTT) [37], with Webhooks [38] publishing a new trigger received or action alerted is conducted, configured, and used for LINE-Notify mobile devices. The developed prototype is presented in Section 2.1.1. The open and powerful cloud analytics platform based on ThingSpeak™ with MATLAB® analytics is presented in Section 2.1.2, where advanced AI implemented in MATLAB® programming language, trained offline in cloud analytics and ported as edge analytics on the developed prototype via the Internet for onsite and online load identification in DSM, is mathematically described in Section 2.2.

2.1.1. Hardware: Smart Autonomous Time and Frequency Analysis Current Sensor-Based Power Meter Prototype as Edge Analytics

The smart autonomous time and frequency analysis current sensor-based power meter prototype designed and implemented as edge analytics with push notifications service in Figure 3 is depicted in Figure 4. The core entity of the prototype in Figure 4 is based on an Arduino® micro-controller unit (MCU) board. More specifically, Arduino MEGA 2560 [21,23,39] is chosen and designed with hardware and software implementation in this study. Arduino®, an open-source and inexpensive electronics prototyping MCU based on flexible and easy-to-use hardware and software, has become very popular among artists, designers, hobbyists, and professionals. It is an excellent IoT tool to quickly test and deploy IoT ideas and applications. Arduino® is programed in Arduino® language, which is based on C/C++ programming language and comes with a user-friendly integrated development environment (IDE) [40]. The Arduino MEGA 2560 MCU used in this study provides sufficient analog pins for possible future uses in smart homes.

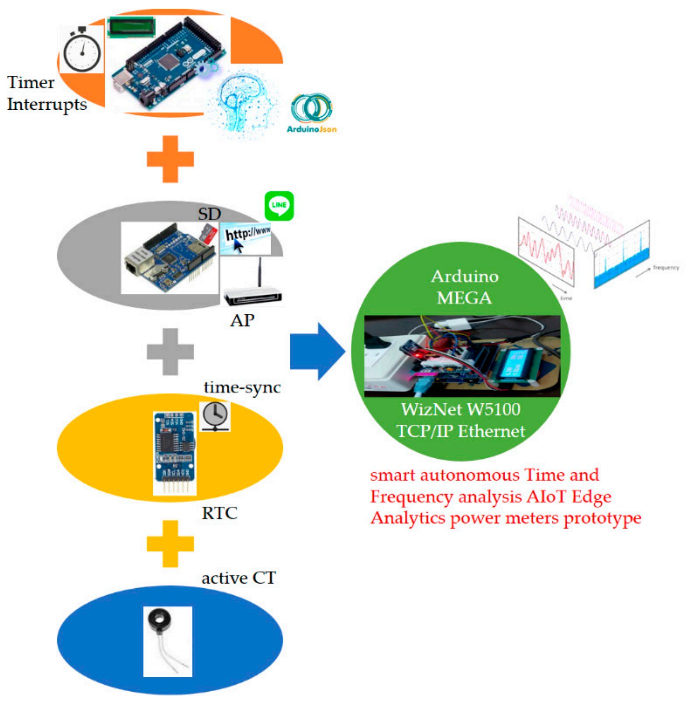

Timer
Interrupts
SD
AP
time-sync
RTC
active CT
Arduino
MEGA
WizNet W5100
TCP/IP Ethernet
smart autonomous Time and
Frequency analysis AIoT Edge
Analytics power meters prototype

Figure 4. Generic three-tier architecture of fog-cloud analytics.

The general specifications of the Arduino MEGA 2560 MCU are listed in Table 1. It is an MCU board based on the Atmel® 8-bit ATmega2560. It has 54 digital input/output (I/O) pins (of which 15 can be used for pulse width modulation (PWM)), 16 analog inputs, four universal asynchronous receivers/transmitters (UARTs; hardware serial ports), a larger memory space for a coded Arduino sketch, a 16 MHz crystal oscillator, a universal serial bus (USB) connection, a power jack, an in-circuit serial programming (ICSP) header, and a reset button. In an Arduino® MCU board, the flash memory, program space, is where an Arduino sketch coded is stored. The static random access memory (SRAM) is where the coded Arduino sketch creates and manipulates variables when the board runs. The electrically erasable programmable read-only memory (EEPROM) is the memory space that programmers can use to store long-term data.

Table 1. Technical specifications of Arduino MEGA 2560 micro-controller unit (MCU) [21,23,39]. SRAM, static random access memory; EEPROM, electrically erasable programmable read-only memory; PWM, pulse width modulation.

Arduino® MCU board	**ATmega 2560**
Operating voltage	5 V
Input voltage (recommended)	7–12 V
Digital input/output (I/O) pins	54 (of which 15 provide PWM output)
Analog input pins	16
Direct current (DC) per I/O pin	20 mA
DC for 3.3 V pin	50 mA
Flash memory	256 KB (8 KB used by the bootloader)
SRAM	8 KB
EEPROM	4 KB
Clock speed	16 MHz
LED_BUILTIN, number of pins on-board LED is connected to	13
Size (length × width)	101.52 mm × 53.3 mm

The Arduino MEGA 2560 MCU is compatible with most Arduino® shields, such as a WIZNet W5100 hardwired Transmission Control Protocol/Internet Protocol (TCP/IP) embedded Ethernet shield [41], produced for Arduino UNO, and former microcontroller boards such as Arduino Duemilanove. A WIZNet W5100 hardwired TCP/IP embedded Ethernet shield is used in this study for the developed prototype (Figure 4). The designed and implemented prototype has the following modules:

(1) A current transducer (CT) coil, which is clipped on a live electrical wire and used to acquire current signals. Acquired current signals reflect electrical appliances monitored by the prototype, analyzed through time- and frequency-domain analysis, and identified by advanced AI. For a CT connected to an Arduino® MCU, the output signal from the CT needs to be conditioned so it meets the input requirements, positive voltage between 0 V and the ADC reference voltage (AREF; 5 V on 5 V MCU boards or 3.3 V on 3.3 V boards), of the Arduino® MCU's analog inputs.

(2) A WIZNet W5100 hardwired TCP/IP embedded Ethernet shield, which is used to support Internet connectivity where data processed through time and frequency analysis are transmitted to the open and powerful cloud analytics platform in Figures 2 and 3 for advanced AI, and where advanced AI well trained offline in cloud analytics is retrieved from the cloud analytics platform and autonomously and automatically deployed as edge analytics on the Arduino MEGA 2560 MCU in a regular periodic manner.

(3) A real-time clock (RTC) chip, which is used to keep track of current time, timestamps, via network time protocol (NTP; the developed prototype designed and implemented to interact with the cloud analytics platform in Figures 2 and 3 for advanced AI trained offline in cloud analytics and ported as edge analytics on the prototype via the Internet is time-synchronized).

(4) A micro SD card, which is used to store data (the SD library built on 'sdfatlib' by Greiman allows for reading data from and writing data to the SD card, and data gathered are reliably replicated on the cloud analytics platform).

(5) Advanced AI, which is ported from the cloud analytics platform to the developed prototype via the Internet and used as edge analytics to identify electrical home appliances (represented as electrical features) based on time- and frequency-domain analysis onsite and online.

Wireless communication technology such as Bluetooth [22], ZigBee [42], Wi-Fi [43,44], and 2G General Packet Radio Services (GPRS) [45] can be conducted and developed in the architecture in Figure 3. In this sense, the WIZNet W5100 hardwired TCP/IP embedded Ethernet shield mounted on the Arduino MEGA 2560 MCU, the core entity of the designed and implemented smart prototype shown in Figure 4 can be replaced with an ESP8266 ESP-01 system-on-a-chip (SoC) Wi-Fi microchip or an ESP8266 ESP-12E Wi-Fi SoC NodeMCU [46] where advanced AI well trained offline in cloud analytics can be autonomously and automatically deployed onsite on the NodeMCU as edge analytics via the over-the-air programming.

The process of data science analytics comprising feature extraction and load identification by the developed prototype considering advanced AI for DSM is shown in Figure 2. Below, feature extraction is described. Advanced AI well trained offline in cloud analytics and autonomously and automatically deployed onsite on the developed prototype as edge analytics is mathematically described in Section 2.2.

To enable feature extraction for load identification in DSM in this study, real power (P) [46,47], turn-on transient power [46], and current harmonics extracted from current signals conditioned and acquired by the CT are the electrical features identified by advanced AI. The turn-on transient power is defined as the real power, which is consumed by an electrical appliance plugged into the developed prototype and monitored, that transient power is captured and extracted when the appliance is turned on (the appliance turned on will settle down). For current harmonics, the conditioned and acquired current signals in the time domain are transformed by fast Fourier transform (FFT) [48–53] into the frequency domain.

The FFT is an algorithm that performs a discrete Fourier transform; it is a mathematical tool that allows signals conditioned and acquired in the time domain to be displayed in the frequency domain, and it is widely used in signal processing. The FFT performed for feature extraction for load identification in DSM in this study was implemented in Arduino® language [52] and run on the Arduino MEGA 2560 MCU. In the Arduino® IDE, programmers only need to program two functions to make an executable Arduino® sketch: *setup()* and *loop()*. The *loop()* function is not ideal for high-speed data acquisition, as it runs continuously without using a timer [54]. Instead, interrupts are used to allow predictable timing for high-speed data acquisition. In this study, interrupts triggered for high-speed data acquisition were implemented by the FlexiTimer2 library [55]. FlexiTimer2, which is based on MsTimer2 [56] by Javier Valencia, offers more time-spatial flexibility than MsTimer2, since it has a configurable submillisecond timer resolution comprising the 1 millisecond timer resolution on timer2 for an Arduino® MCU's interrupts. In an Arduino® MCU, events that trigger interrupts are internal timer overflows. Each time there is a timer overflow, a callback routine is called and executed. In our case, the callback routine is a function in which raw current data are read through *analogRead()* from a specified analog pin of the MCU circuited with a CT (Figure 4). The MCU contains a multichannel and 10-bit analog-to-digital converter (ADC), which maps input voltages between 0 and 5 V into integer values ranging from 0 to 1023. *analogRead()* reads mapped integer values from the specified analog pin of the MCU, and they are shifted by their mean value, computed with a voltage step of ~4.88 mV, and then transformed by FFT to feature extraction for load identification in DSM.

To enable load identification for DSM in this study, advanced AI trained offline in cloud analytics (i.e., applied on electrical features extracted through feature extraction for onsite and online load identification and transmitted to the cloud analytics platform for advanced AI in the architecture shown in Figures 2 and 3) is autonomously/automatically deployed onsite on the developed smart autonomous power meter prototype as edge analytics and used to identify electrical appliances. Across the architecture over fog-cloud analytics, load identification for DSM in this study is performed onsite and online.

The cloud analytics platform that collects data transmitted from the developed smart autonomous power meter prototype as edge analytics for advanced AI and runs advanced AI trained offline in cloud analytics for DSM is described in Section 2.1.2.

2.1.2. Software: Open and Powerful Cloud Analytics Platform Based on ThingSpeak™ with MATLAB® Analytics

The open and powerful cloud analytics platform that collects data transmitted from the developed smart autonomous power meter prototype as edge analytics for advanced AI and runs advanced AI trained offline in cloud analytics for DSM is described here. As shown in Figure 3, the cloud analytics platform as a data science analytics engine without developing custom web software for DSM is based on ThingSpeak™ [36] with MATLAB® analytics. ThingSpeak™ is a free, open, and powerful IoT platform that can run MATLAB® codes on demand in the cloud. Also, it offers an easy way to collect IoT data from things for IoT application developers, analyze data, and visualize data processed by MATLAB®, where application developers can act on their collected and processed IoT data [36]. With MATLAB® from MathWorks®, application developers have access to powerful toolboxes such as the Signal Processing Toolbox™, Statistics and Machine Learning Toolbox™, and Neural Network Toolbox™ for their analyzed and visualized IoT data. Figure 5 shows the workflow of the cloud analytics platform, ThingSpeak™ with MATLAB® analytics, configured over fog-cloud analytics for advanced AI implemented for DSM in this study, which is completed in the following three steps:

(1) Collect (historical) IoT data (electrical feature data). Electrical features are extracted from appliances and learned by advanced AI in cloud analytics for DSM, such as load identification, fault detection and classification, and prognostics and health management. Advanced AI well trained in cloud analytics will be autonomously and automatically deployed on, or ported on, onsite the developed smart autonomous power meter prototype via the Internet for on-demand data science analytics and data visualization.

(2) Perform data science analytics in the ThingSpeak™ cloud.

    (2.1) Analyze collected IoT data (electrical feature data), which are statistically inspected and cleaned to remove noise, outliers, missing or erroneous values, and anomalies.

    (2.2) Implement advanced AI applied on analyzed data for interpretable and actionable insights (code advanced AI in a MATLAB®script), publish the coded script in the ThingSpeak™ cloud, and then schedule the published script to regularly/periodically execute the implemented advanced AI for DSM such as load identification, fault detection and classification, and prognostics and health management.

(3) Deploy on-demand data science analytics over fog-cloud analytics for the implemented advanced AI onsite on the developed prototype at the edge of the Internet. Over fog-cloud analytics, the prototype communicates with the ThingSpeak™ cloud for model updates and transmits analyzed and classified data to the ThingSpeak™ cloud for model scheduling and updating.

Advanced AI involved in the fog-cloud analytics architecture in this study is described in Section 2.2.

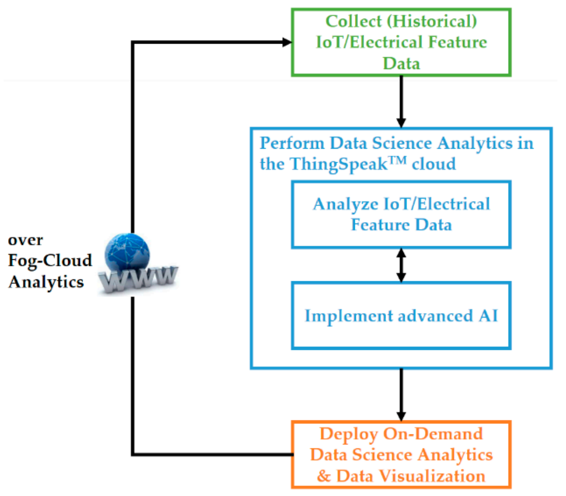

Collect (Historical) IoT/Electrical Feature Data
Perform Data Science Analytics in the ThingSpeak™ cloud
Analyze IoT/Electrical Feature Data
Implement advanced AI
Deploy On-Demand Data Science Analytics & Data Visualization
over Fog-Cloud Analytics
WWW

Figure 5. Workflow of the open and powerful cloud analytics platform, ThingSpeak™ with MATLAB® analytics, configured over fog-cloud analytics for DSM in this study.

2.2. Advanced AI: K-Means Clustering Combined with RBF-ANN Trained Offline in Cloud Analytics and Autonomously and Automatically Deployed Onsite on the Developed Smart Autonomous Power Meter Prototype as Edge Analytics via the Internet

A biologically inspired RBF-ANN [57] facilitated by k-means clustering [58–60] is developed and used as advanced AI trained offline in cloud analytics and autonomously and automatically deployed onsite on the developed smart autonomous power meter prototype as edge analytics over fog-cloud analytics for DSM. The RBF-ANN [57], a rather simple three-layer ANN, used in this study consists of one input layer, one hidden layer comprising artificial neurons with radial basis functions taking on the role of nonlinear activation functions, and one output layer, as shown in Figure 6. The RBF-ANN has conspicuous fast and high learning and generalization performance. In this study, k-means clustering, also known as hard c-means clustering or Lloyd's algorithm [58], is used to characterize the standard RBF-ANN in Figure 6. K-means clustering is a data-partitioning-based clustering algorithm that assigns data points into k clusters (centroids) across a number of variables through an iterative process.

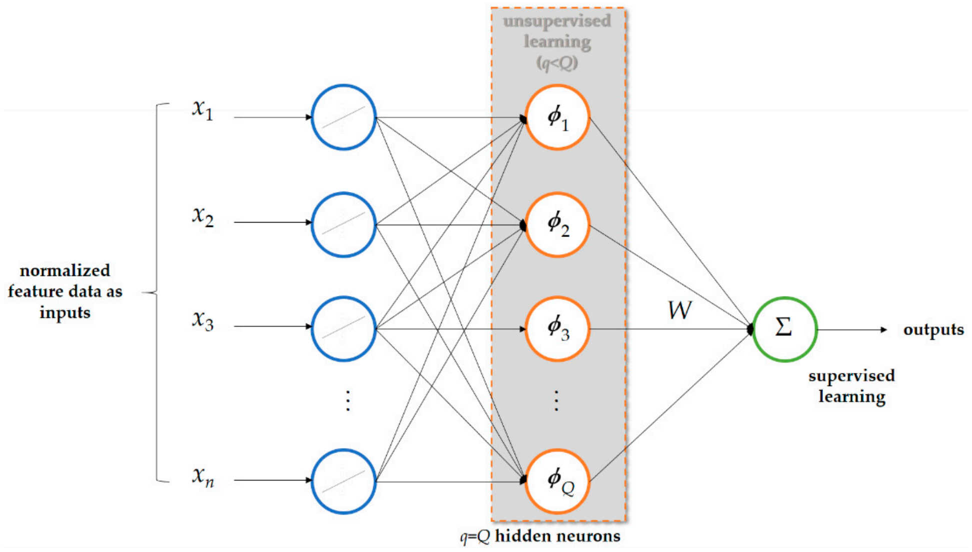


Figure 6. Standard radial basis function artificial neural network (RBF-ANN), lightweight AI.

Let $\Im = \{X_k, d_k\}_{k=1}^{Q}$ be a training dataset containing Q input–output pairs, where $X_k \in R^n$ and $d_k \in R$. The goal of training the RBF-ANN is to search for map f that takes each input X_k $(k = 1, \ldots, Q)$ and then maps exactly it onto its desired output d_k: $f(X_k) = d_k$.

With the purpose of mapping $f(X_k) = d_k$ to be learned through the training process of the RBF-ANN, the RBF-ANN assumes a set of exactly q nonlinear basis functions $\phi(\|X_k - \mu_i\|)$ whose argument involves a Euclidean distance metric $\|X_k - \mu_i\|$. The Euclidean distance metric measures the distance between the kth input X_k and the ith center (cluster mean) μ_i, where $i = 1, 2, \ldots, q$ (= c cluster means found by k-means, or hard c-means, clustering) and $\mu_i \in R^n$. In this study, the radially symmetric Gaussian-style basis function is used.

Map f is then generated by taking a weighted linear superposition of q nonlinear basis functions, as given in Equation (1):

$$f(X) = \sum_{i=1}^{q} w_i \phi(\|X_k - \mu_i\|) \tag{1}$$

where w_i is the ith weight coefficient.

Map f is solved in a least squares sense, where the familiar squared error function that computes the squared error summed over all Q data points in the training dataset is introduced in Equation (2):

$$\varepsilon = \frac{1}{2}\sum_{k=1}^{Q} [d_k - \sum_{i=1}^{q} w_i \phi(\|X_k - \mu_i\|)]^2 \tag{2}$$

To obtain the optimal weight coefficients in a least squares sense, we differentiate Equation (2) with respect to w_i and set it equal to zero, as shown in Equation (3):

$$\sum_{k=1}^{Q} \phi_{ki}(\sum_{l=1}^{q} w_l \phi_{kl}) = \sum_{k=1}^{Q} d_k \phi_{ki} \tag{3}$$

By the following matrix definitions:

$\Phi = \begin{bmatrix} \phi_{11} & \cdots & \phi_{1q} \\ \phi_{21} & \cdots & \phi_{2q} \\ \vdots & \ddots & \vdots \\ \phi_{Q1} & \cdots & \phi_{Qq} \end{bmatrix}$, D = $\left[d_1, \ldots, d_Q\right]^T$, and W = $\left[w_1, \ldots, w_q\right]^T$, Equation (3) is recast into a matrix form, as shown in Equation (4):

$$(\Phi^T \Phi) W = \Phi^T D \tag{4}$$

Finally, the weight vector W in Equation. (4) is solved by a singular value decomposition (SVD) technique [57], which is used to train the RBF-ANN once the specification of the center and spread parameters of each Gaussian basis function has been completed through the k-means clustering, as given in Equation (5):

$$W = (\Phi^T \Phi)^{-1} \Phi^T D = \Phi^* D \tag{5}$$

where Φ^*, a q-by-Q matrix, is the pseudo-inverse [57,61].

From the regularization theory, where the standard error minimization problem is replaced with minimization of a regularization risk functional [57] applied to Equation (2) with a smoothness functional, Equation (5) becomes Equation (6):

$$W = (\mathbf{G}^T \mathbf{G} + \lambda \tilde{\mathbf{G}})^{-1} \mathbf{G}^T D = \mathbf{G}^* D \tag{6}$$

where $\mathbf{G}$ denotes a Green's function (multivariate Gaussian is a Green's function) [57]: $\mathbf{G} = \begin{bmatrix} G(\|X_1-\mu_1\|) & \cdots & \mathbf{G}(\|X_1-\mu_q\|) \\ G(\|X_2-\mu_1\|) & \cdots & G(\|X_2-\mu_q\|) \\ \vdots & \ddots & \vdots \\ G(\|X_Q-\mu_1\|) & \cdots & G(\|X_Q-\mu_q\|) \end{bmatrix}$; $\tilde{\mathbf{G}} = \begin{bmatrix} G(\|\mu_1-\mu_1\|) & \cdots & G(\|\mu_1-\mu_q\|) \\ G(\|\mu_2-\mu_1\|) & \cdots & G(\|\mu_2-\mu_q\|) \\ \vdots & \ddots & \vdots \\ G(\|\mu_q-\mu_1\|) & \cdots & G(\|\mu_q-\mu_q\|) \end{bmatrix}$; and λ is a regularization parameter that controls the trade-off between the closeness to the fitted data and the smoothness of the regularized Tikhonov functional [57] (the Tikhonov functional, also called a stabilizer, forces the approximation to become as smooth as possible). The pseudo-inverse $\mathbf{G}^*$ in Equation (6) trends to Φ^* in Equation (5) as $\lambda \rightarrow 0$ (there is no smoothness constraint; W is completely determined by the training dataset $\mathfrak{I}$).

In this study, the radially symmetric Gaussian basis functions, $\phi(\|X_k-\mu_i\|) = \exp(-\frac{\|X_k-\mu_i\|^2}{2\sigma^2})$, involving the distributed center and spread parameters partially overlapped and heuristically determined by k-means clustering are adopted for the RBF-ANN in Equation (1) (Figure 6). Assuming that the Gaussian basis functions are centered at $\{\mu_i\}_{i=1}^{q}$ as cluster means by k-means clustering, we define the maximum distance α between any of the chosen center parameters as $\alpha = \max\limits_{1\le i,j\le q}(\|\mu_i-\mu_j\|)$. Then, the spread parameters σ of the centered Gaussian basis functions are computed by Equation (7) [57]:

$$\sigma = \frac{\alpha}{\sqrt{2q}} \tag{7}$$

Overall, Equation (6) admits the RBF-ANN shown in Figure 6. In summary, the standard RBF-ANN of Equation (1) characterized through k-means clustering, trained by Equation (6), and developed in this study is involved in the following two stages and is outlined in Algorithm 1:

(1) Stage 1. Apply k-means clustering to the onsite collected training data points, to heuristically determine the center and spread parameters of each Gaussian basis function of Equation (1). The Gaussian-type basis function's centers are spanned, as computed by Equation (7). K-means clustering, which is used in this stage to characterize the RBF-ANN of Equation (1) trained in the following stage, is summarized below:

Step 1. Suppose an onsite collected training dataset $\mathfrak{I} = \{X_k, d_k\}_{k=1}^{Q}$ with $X_k \in R^n$ is given. Fix c (k-means or hard c-means clustering), where $2 \le c < Q$. Initialize $U^{(0)} \in M_c$, where $M_c = \{U \in V_{cn} \mid u_{ik} \in \{0, 1\}, 1 \le i \le c, 1 \le k \le Q; \sum_{i=1}^{c} u_{ik} = 1, \forall k \in \{1,2,\ldots,Q\}; 0 < \sum_{k=1}^{Q} u_{ik} < Q, \forall i \in \{1,2,\ldots,c\}$ are true$\}$; V_{cn} is the set of real $c \times Q$ matrices $U = [u_{ik}]$ (hard c-partition space for $\mathfrak{I}$).

Step 2. At iteration l, where l = 0, 1, 2, … , compute the c-mean centers using Equation (8). In Equation (8), $[{u_{ik}}^{(l)}] = U^{(l)}$ and i = 1, 2, … c:

$$V_i^{(l)} = \frac{\sum_{k=1}^{Q} X_k {u_{ik}}^{(l)}}{\sum_{k=1}^{Q} {u_{ik}}^{(l)}} \tag{8}$$

Step 3. Update $U^{(l)}$ to $U^{(l+1)} = [{u_{ik}}^{(l+1)}]$ using Equation (9), where data points with short distances among them should be grouped together:

$${u_{ik}}^{(l+1)} = \begin{cases} 1\,, & \left\|X_k - {V_i}^{(l)}\right\| = \min_{1\le j\le c}\left(\left\|X_k - {V_j}^{(l)}\right\|\right) \\ 0\,, & \textit{otherwise} \end{cases} \tag{9}$$

Step 4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \varepsilon$ for a prespecified small tolerance, stop; otherwise, set $l = l + 1$ and go to Step 2. The cluster centers found by Equation (8) through k-means clustering are used to heuristically determine the center and spread parameters of each Gaussian basis functions of Equation. (1). Data points with short distances among them are clustered together by k-means clustering and learned by an exponential of a distance measure between the data points and their cluster center that characterizes the Gaussian basis function at a hidden neuron in the hidden layer of the RBF-ANN in Figure 6.

(2) Stage 2. Use the SVD technique applied to Equation (6) with the training data points collected onsite to train the standard RBF-ANN of Equation (1) characterized in Stage 1. Once the k-means clustering combined with RBF-ANN has been well trained in the ThingSpeak$^{\text{TM}}$ cloud, it is autonomously and automatically deployed onsite on (ported on) the developed smart autonomous power meter prototype as edge analytics via the Internet. Then, the developed prototype is used to perform online load identification for DSM.

IoT edge devices can be hosted in devices such as industrial controllers, MCUs, switches, routers, embedded systems or servers, and video surveillance cameras. As many such IoT devices may not have sufficient and powerful computation resources to efficiently perform AI (data science analytics) for real-time responsiveness and local interpretable and actionable data insights, it is necessary to use lightweight AI deployed and run on IoT devices (lightweight AI does not require high-end computational resources [62]). Hence, in this study, the biologically inspired RBF-ANN lightweight AI in Equation (6) is conducted over fog-cloud analytics in the described architecture. Due to its simplicity in the training process, the RBF-ANN has much shorter training time than a multilayer perceptron model [63] trained by the commonly used back-propagation algorithm [62,63]. It has the merits of (1) a simple network structure and (2) conspicuous fast and high learning and generalization performance [23,64,65].

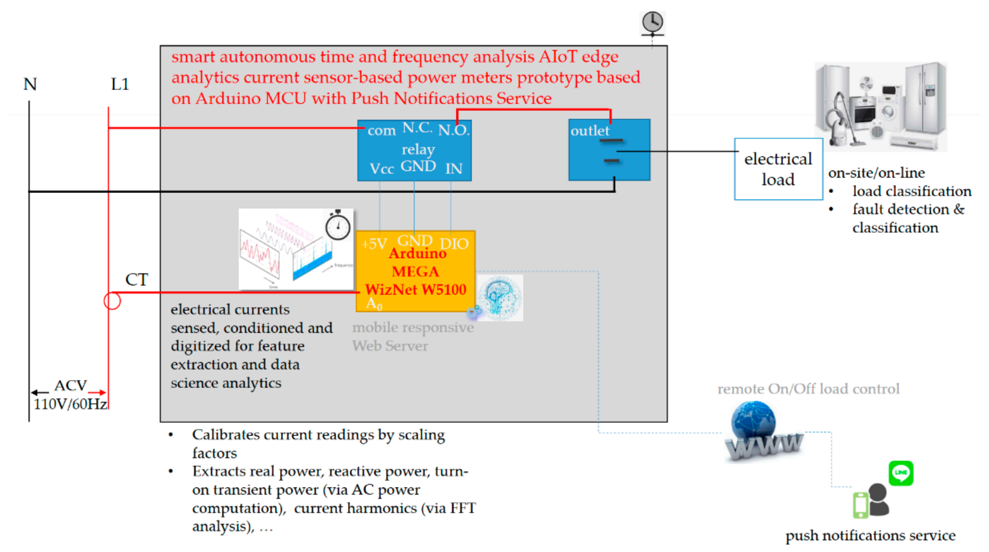
3. Experiments

3.1. Demo Prototype and Evaluation

In this section, the developed smart autonomous time and frequency analysis current sensor-based power meter prototype in the edge analytics-based AIoT architecture launched with the open and powerful ThingSpeak™ IoT platform with MATLAB® analytics as cloud analytics for DSM in a smart grid is experimentally and practically demonstrated. Figure 7 shows the proof-of-concept demonstration of the developed prototype. The prototype is installed for an electrical network topology and used as a smart electrical outlet (wall socket), as shown in Figure 7a. Figure 7b shows the experimental setup of the prototype where advanced AI well trained by Algorithm 1 in the ThingSpeak™ cloud is autonomously and automatically deployed via the Internet, as shown in Figure 7c.

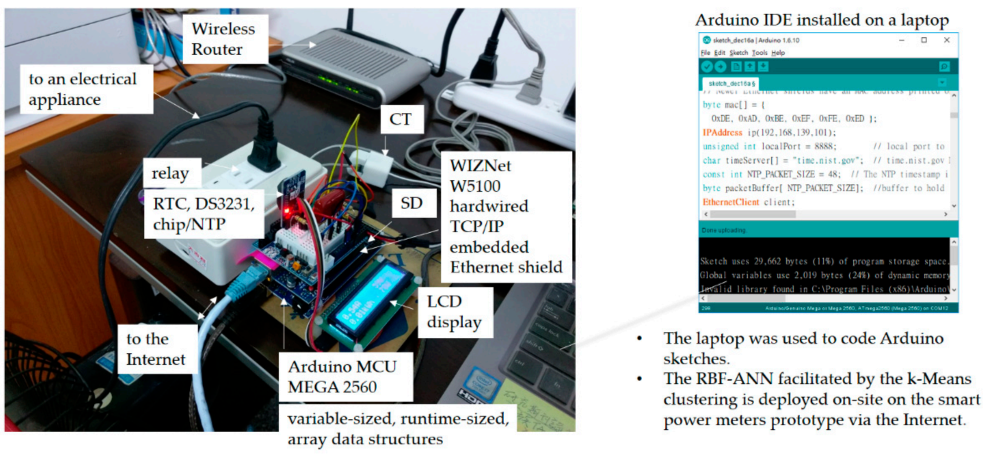
As shown in Figure 7b, we assembled hardware and software of the Arduino MEGA 2560 MCU mounted with the Arduino W5100 Ethernet shield, which was used to: (1) acquire current signals sensed by the CT and conditioned by the MCU; (2) extract and identify electrical features extracted through the time- and frequency-domain analysis from the conditioned and digitized current signals and identified by advanced AI autonomously and automatically deployed onsite; (3) transmit data (such as electrical energy consumption and electrical features including P [46,47], turn-on transient power [46], and current harmonics) to and retrieve data (such as trained advanced AI entries) from the ThingSpeak™ cloud via the Internet (refer to Figure 7c, showing the prototype working over fog-cloud analytics); and (4) provide mobile push notification service to end users for DSM. In the fog-cloud analytics architecture in this study, load identification for DSM shown in Figures 2 and 3 is performed onsite and online. The prototype can be further developed based on analysis of daily electrical energy consumption for DSM, to disaggregate whole-house load data into appliance-level load information with no intrusive deployment of individual smart plugs installed for electrical appliances monitored in a residential field of interest.

Figure 7c shows that the prototype in Figure 7a,b communicates with the ThingSpeak™ cloud via the Internet, where they collaborate to work over fog-cloud analytics for load identification in DSM. The ThingSpeak™ cloud configured for advanced AI is shown later.



N
L1
smart autonomous time and frequency analysis AIoT edge analytics current sensor-based power meters prototype based on Arduino MCU with Push Notifications Service
com N.C. N.O.
relay
Vcc GND IN
outlet
electrical load
on-site/on-line
• load classification
• fault detection & classification
CT
+5V GND DIO
Arduino MEGA
WizNet W5100
A0
mobile responsive Web Server
electrical currents sensed, conditioned and digitized for feature extraction and data science analytics
ACV 110V/60Hz
remote On/Off load control
• Calibrates current readings by scaling factors
• Extracts real power, reactive power, turn-on transient power (via AC power computation), current harmonics (via FFT analysis), …
push notifications service

(**a**)



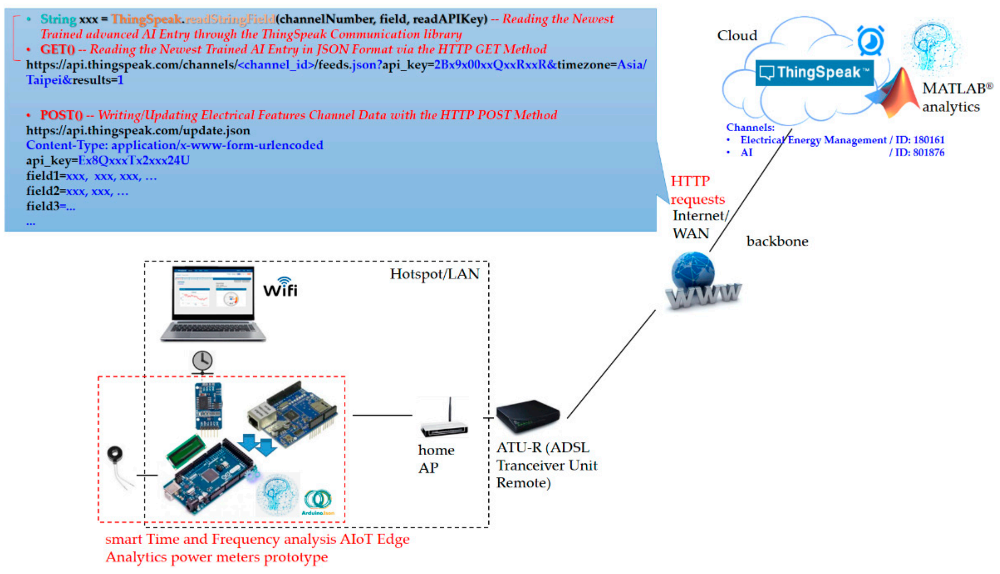
Wireless Router
to an electrical appliance
CT
relay
RTC, DS3231, chip/NTP
SD
WIZNet W5100 hardwired TCP/IP embedded Ethernet shield
LCD display
to the Internet
Arduino MCU MEGA 2560
variable-sized, runtime-sized, array data structures
Arduino IDE installed on a laptop
sketch_dec16a | Arduino 1.6.10
File Edit Sketch Tools Help
sketch_dec16a §
Done uploading.
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM12

```
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,139,101);
unsigned int localPort = 8888;       // local port to
char timeServer[] = "time.nist.gov";  // time.nist.gov
const int NTP_PACKET_SIZE = 48;  // The NTP timestamp i
byte packetBuffer[ NTP_PACKET_SIZE];  //buffer to hold
EthernetClient client;
```

```
Sketch uses 29,662 bytes (11%) of program storage space.
Global variables use 2,019 bytes (24%) of dynamic memory
Invalid library found in C:\Program Files (x86)\Arduino\
```

- The laptop was used to code Arduino sketches.
- The RBF-ANN facilitated by the k-Means clustering is deployed on-site on the smart power meters prototype via the Internet.

(**b**)



Cloud
ThingSpeak™
MATLAB® analytics
Channels:
• Electrical Energy Management / ID: 180161
• AI / ID: 801876
HTTP requests
Internet/WAN
backbone
Hotspot/LAN
Wifi
home AP
ATU-R (ADSL Tranceiver Unit Remote)
smart Time and Frequency analysis AIoT Edge Analytics power meters prototype

- String xxx = ThingSpeak.readStringField(channelNumber, field, readAPIKey) -- *Reading the Newest Trained advanced AI Entry through the ThingSpeak Communication library*
- GET() -- *Reading the Newest Trained AI Entry in JSON Format via the HTTP GET Method*

```
https://api.thingspeak.com/channels/<channel_id>/feeds.json?api_key=2Bx9x00xxQxxRxxR&timezone=Asia/
Taipei&results=1
```

- POST() -- *Writing/Updating Electrical Features Channel Data with the HTTP POST Method*

```
https://api.thingspeak.com/update.json
Content-Type: application/x-www-form-urlencoded
api_key=Ex8QxxxTx2xxx24U
field1=xxx,  xxx, xxx, …
field2=xxx, xxx, …
field3=...
...
```

(**c**)

Figure 7. Proof-of-concept demonstration of the developed smart autonomous time and frequency analysis current sensor-based power meter prototype as edge analytics in the launched edge analytics-based AIoT

architecture. (**a**) Sketch of the prototype designed, implemented, and installed for an electrical network topology and used as a smart electrical outlet for DSM in a smart grid. (**b**) Experimental setup of the prototype experimentally and practically evaluated (the prototype developed and experimentally tested in a laboratory environment in [23] is upgraded in this study). (**c**) The prototype automatically communicates with the ThingSpeak™ cloud via the Internet (updating ThingSpeak™ channel data with HTTP GET() or POST()). Its total controllability was shown in [23].

In the experiment demonstrated in this study, instantaneous current signals sensed and conditioned are acquired and the electrical features, P, turn-on transient power, and current harmonics are extracted by the prototype shown in Figure 7a,b from the acquired instantaneous current signals. According to the Nyquist–Shannon sampling theorem, in which the sampling frequency specified to indicate how frequently a signal one is trying to acquire is sampled per second needs to be at least twice the frequency of the signal acquired, the sampling frequency specified in this experiment is 2 kHz (0.5 ms/sample).

Algorithm 1. The two-stage method developed for the advanced AI k-means clustering-integrated RBF-ANN over fog-cloud analytics in this study.

```
• Use k-means clustering on a training dataset collected onsite, $\Im = \{X_k, d_k\}_{k=1}^{Q}$, to heuristically determine
  the center and spread parameters of each Gaussian basis function of the RBF-ANN of Equation (1).

↬ **Require** the values of the number of cluster centers $c$ and a small constant of the tolerance $\varepsilon$ to be specified.
↬ **Randomly initialize** the $c$ cluster centers, where $c$ has been specified.

$l \leftarrow 0$
↬ **Repeat**
            **Compute** the $c$ cluster centers using Equation (8).
            **Update** the hard c-partition space $U$ using Equation (9).

    $l \leftarrow l + 1$
  **Until**
        $\|U^{(l+1)} - U^{(l)}\|$ is less than the specified small tolerance $\varepsilon$.
↬ **Return** the resulting $c$ cluster centers with the hard c-partition space for $\Im$.

↬ **Center** the Gaussian basis functions of Equation (1) at the returned $c$ cluster centers.
↬ **Compute** the spread parameters of the centered Gaussian basis functions of Equation (1) using
  Equation (7).

• Use Equation (6) involving the SVD technique applied to the training dataset collected onsite, $\Im$, to train
  the RBF-ANN of Equation (1) characterized by k-means clustering.

↬ **Train** Equation (1), whose Gaussian basis functions were heuristically determined by k-means clustering
applied to the training data points collected onsite using Equation (6) in the ThingSpeak$^{TM}$ cloud to get $W$.
The well-trained RBF-ANN is then autonomously and automatically deployed onsite on the developed smart
autonomous power meter prototype as edge analytics via the Internet for online load identification in DSM.
In this study, the RBF-ANN facilitated with k-means clustering is developed over fog-cloud analytics.
```

This sampling frequency means analog current signals conditioned and acquired for the time- and frequency-domain analysis are sampled 2000 times per second (i.e., sampled in a time resolution of 1/2000th of a second); the high-speed predictable interrupts enabled by the Arduino MEGA 2560 MCU are specified through the configurable submillisecond timer resolution of the FlexiTimer2 library [55]. The specified sampling frequency limits the speed at which analog current signals from monitored electrical appliances can be sampled.

Figure 8 shows the analog current signals conditioned and acquired by the specified MCU's analog pin in Figure 7a, from two electrical appliances, an electric fan (Figure 8a) and a hair dryer (Figure 8b).

The conditioned and acquired analog current signals have an amplitude and offset such that they almost span the complete range of 0–5 V, well suiting the MCU's ADC properties. Figure 9 shows the single-sided amplitude spectrums by FFT applied on the conditioned and acquired analog current signals in Figure 8.

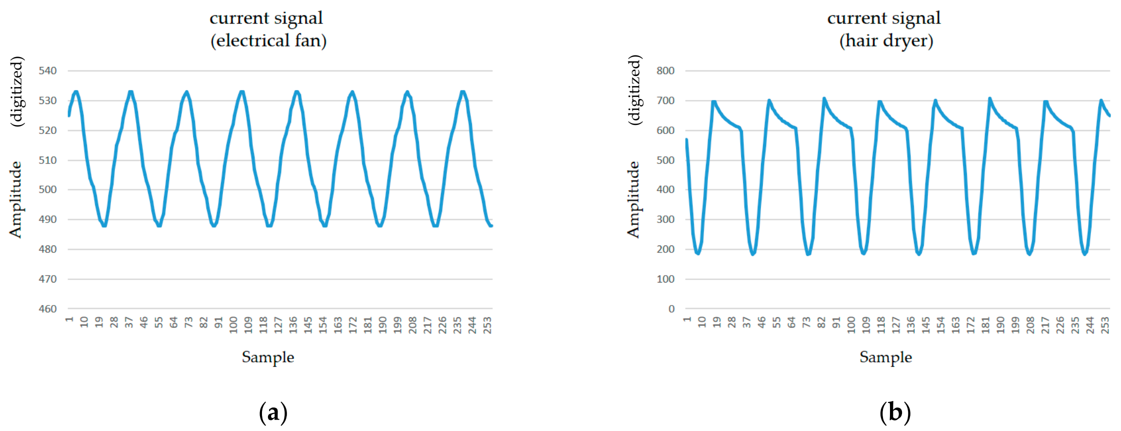


Figure 8. Current signals conditioned and acquired from (**a**) an electric fan and (**b**) a hair dryer.

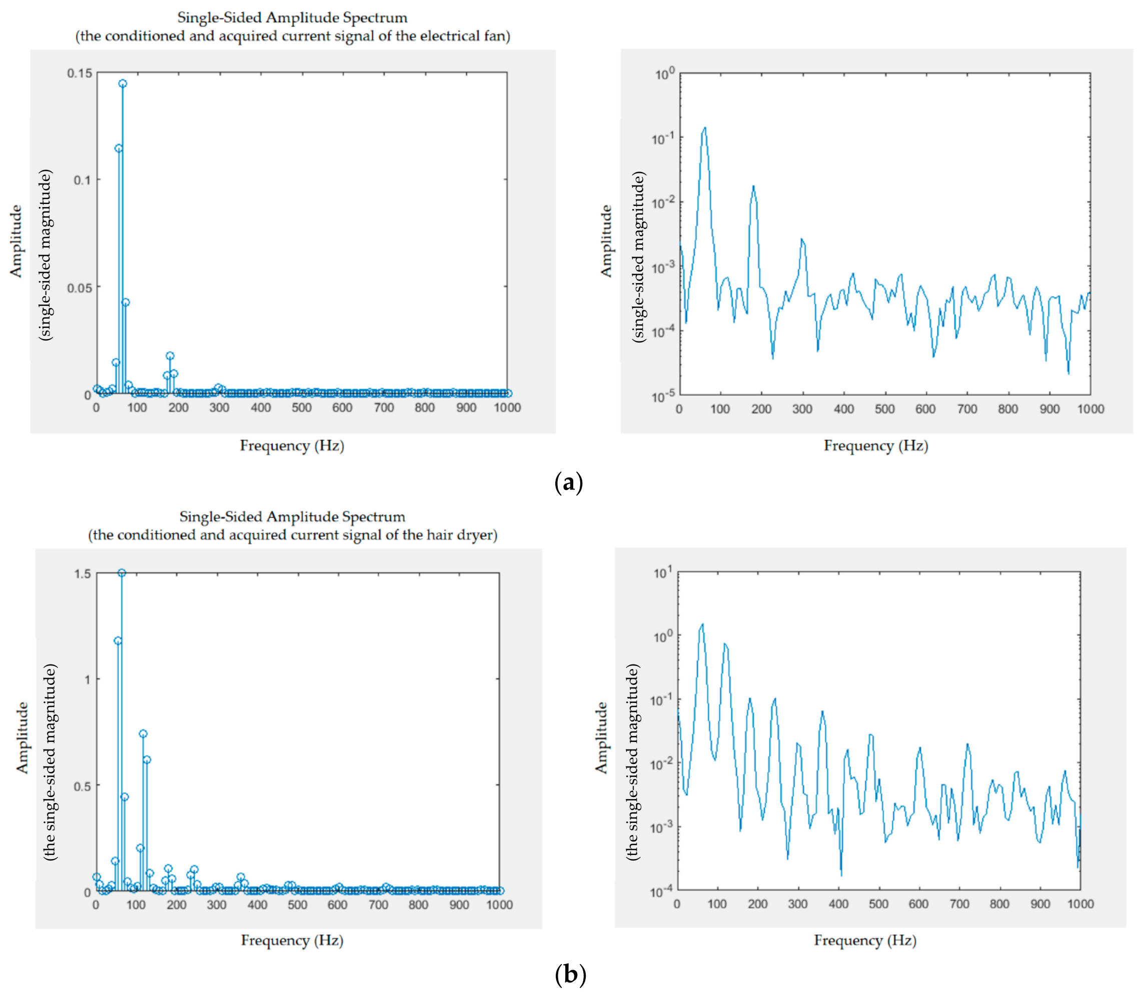


Figure 9. Frequency-domain single-sided amplitude spectra: (**a**) fast Fourier transform (FFT) applied for (**a**) electric fan in Figure 8a, and (**b**) hair dryer in Figure 8b. The fundamental frequency of the conditioned, acquired, and digitized current signals transformed is 60 Hz. On average, longer signals produce better frequency approximation. Note the logarithmic scale.

The conditioned and acquired analog current signals from the two electrical appliances are transformed through FFT from the time domain to the frequency domain, which shows that the current harmonics produced by the two appliances are distinguished. FFT enables seeing which frequencies are present and understanding which is the most dominant in analog signals conditioned, acquired, and digitized to be transformed, which works with a finite number of samples of such signals.

In the FFT applied in this study, the Hanning window, a mathematical function expressed as $w(n) = 0.5(1 - \cos(2\pi\frac{n}{N})), 0 \leq n \leq N$ (window length $L = N + 1$) [53], was used for general data analysis with a good trade-off between frequency and amplitude to mitigate the spectral leakage (i.e., the leakage effect) by smoothing the sinusoidal signals added together with different levels of frequency and amplitude in the time domain. Due to the 256 samples transformed through FFT in this experiment, a total of 128 bins (shown in Figure 9) distributed over 0–1 kHz because of the specified sampling frequency of 2 kHz were obtained; the number of bins obtained through FFT is half the amount of samples spanning the frequency range from 0 to half the sampling frequency. In the FFT implemented and run on the Arduino® MCU, the number of samples transformed (256 in this study) must always be a power of 2, and the specified sampling frequency (2 kHz in this study) has to comply with the Nyquist–Shannon sampling theorem and be less than the maximum reading rate of *analogRead()* (10 kHz in this study) due to the technical specification of the MCU's ADC.

Table 2 lists the current harmonics produced by the two electrical appliances and obtained through FFT in this experiment. As shown in Figure 9a,b, patterns and levels of amplitude (magnitude) at certain frequencies are used as electrical features to distinguish between the two appliances analyzed for feature extraction and used for load identification.

Table 2. Resulting current harmonics produced by two types of electrical appliances and obtained through fast Fourier transform (FFT) in this experiment.

Electric Fan	**DC Offset**	**Fundamental Harmonic (60 Hz)**	**2nd-Order Harmonic**	**3rd-Order Harmonic**	**4th-Order Harmonic**	**5th-Order Harmonic**
	0.002	0.144	0.000	0.018	0.000	0.003
	6th-order harmonic	**7th-order harmonic**	**8th-order harmonic**	**9th-order harmonic**		
	0.000	0.001	0.001	0.001		
Hair Dryer	**DC Offset**	**Fundamental Harmonic (60 Hz)**	**2nd-Order Harmonic**	**3rd-Order Harmonic**	**4th-Order Harmonic**	**5th-Order Harmonic**
	0.070	1.498	0.741	0.105	0.103	0.020
	6th-Order Harmonic	**7th-Order Harmonic**	**8th-Order Harmonic**	**9th-Order Harmonic**		
	0.066	0.016	0.028	0.002		

Current harmonics are needed, as ambiguous electrical features in △P and △Q (reactive power) [47] exist realistically for load identification; spectrum analysis conducted for feature extraction is suitable to complete load identification. More representative electrical features, such as total harmonic distortion of current signals computed from current harmonics, can be further extracted for feature extraction, evaluated for feature selection, and used for load identification. The open and powerful ThingSpeak™ cloud [36] configured in this study offers free cloud storage for electrical features extracted by the smart autonomous power meter prototype from different types of monitored electrical appliances. Table 3 summarizes the fields of two ThingSpeak™ channels, electrical energy management and (advanced) AI, configured over fog-cloud analytics for load identification in DSM in this study. Table 4 shows the code by which extracted electrical features are uploaded (transmitted) to the ThingSpeak™ cloud for advanced AI, which is executed by the prototype shown in Figures 3 and 7. Figure 10 shows electrical energy consumption identified by the prototype, where a simple web page is provided by ThingSpeak™ to user clients for data visualization.

Table 3. Fields of two ThingSpeak™ channels configured over fog-cloud analytics for load identification in DSM in this study.

Channel	Field	Physical Meaning/Sensor
Electrical energy management	Field 1 (RMSPower [1])	Real power/CT
	Field 2 (Ptransient/peakPower)	Turn-on transient power/CT
	Field 3 (maximum magnitude up to 9th-order current harmonics)	Current harmonics/CT
	Field 4 (label)	Advanced AI
AI (RBF-ANN facilitated by k-means clustering)	Field 1 (q (= c by k-means clustering))	–
	Field 2 (σ)	–
	Field 3 (centers by k-means clustering)	–
	Field 4 (centers by k-means clustering)	–
	Field 5 (centers by k-means clustering)	–
	Field 6 (centers by k-means clustering)	–
	Field 7 (centers by k-means clustering)	–
	Field 8 (W)	–

[1] Real power computed at each time interval, for example, 1 s.

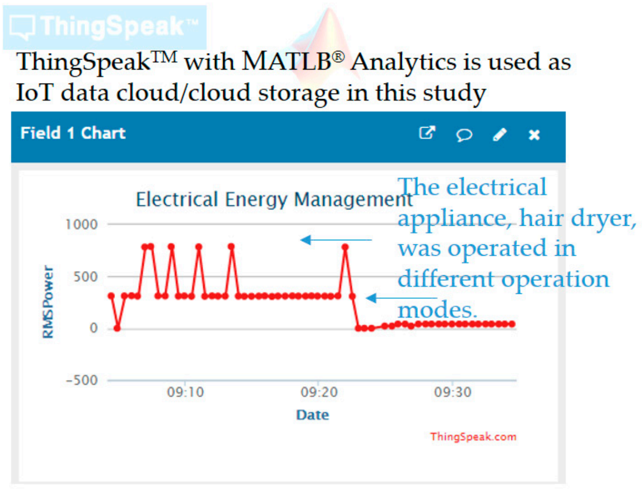

ThingSpeak™
ThingSpeak™ with MATLB® Analytics is used as IoT data cloud/cloud storage in this study
Field 1 Chart
Electrical Energy Management
The electrical appliance, hair dryer, was operated in different operation modes.
1000
500
0
−500
RMSPower
09:10
09:20
09:30
Date
ThingSpeak.com

Figure 10. Simple web page visualizing electrical energy consumption identified by the developed smart autonomous power meter prototype in this study.

Table 4. Code executed by the developed smart autonomous power meter prototype and used to upload (transmit) extracted electrical features to the ThingSpeak™ cloud for data storage to be further analyzed in cloud analytics.

```
/* . . . */
. . .
String currentHarmonics = "";
byte server[] = { 184, 106, 153, 149 }; // IP address (or api.thingspeak.com) for ThingSpeak
(https://thingspeak.com/)
String writeAPIKey = "Ex8QxxxTxxxxxx4U"; // Write API key for a ThingSpeak channel

. . .
. . .
updateThingSpeak("field1=" + String(RMSPower) + "&field2=" + String(peakPower) + "&field3="
+ currentHarmonics);

. . .
. . .
void updateThingSpeak(String tsData) {    // RMSPower, peakPower (Ptransient), current harmonics,
 . . . more

   if (client.connect(server, 80)) {
      Serial.println(F("Connected to ThingSpeak..."));
      client.print("POST /update HTTP/1.1\n");
      client.print("Host: api.thingspeak.com\n");
      client.print("Connection: close\n");
      client.print("X-THINGSPEAKAPIKEY: "+writeAPIKey+"\n");
      client.print("Content-Type: application/x-www-form-urlencoded\n");
      client.print("Content-Length: ");
      client.print(tsData.length());
      client.print("\n\n");
      client.print(tsData);
   }
 . . .

}

. . .
```

The ThingSpeak™ cloud used and configured in this study also offers data science analytics that provides MATLAB® analytics, and collected and stored electrical features are learned by advanced AI, the k-means clustering integrated with RBF-ANN described in Section 2.2, implemented in MATLAB® programming language and trained offline in cloud analytics. The k-means clustering-combined RBF-ANN is autonomously and automatically deployed via the Internet on the developed prototype for load identification (DSM) performed onsite and online.

Two electrical appliances, an electric fan and a hair dryer, were monitored in this experiment by the developed prototype. Electrical features, P, turn-on transient power, and current harmonics were extracted from the two appliances, transmitted to the ThingSpeak™ cloud for advanced AI, and identified by the prototype once the advanced AI applied on stored electrical feature data in the ThingSpeak™ cloud was well trained and autonomously and automatically deployed onsite on the prototype via the Internet for on-demand data science analytics for DSM. Figure 11 shows the MATLAB® script coded for advanced AI in this study, where the following steps were applied:

(1) Use the MATLAB® Analysis app [66,67] to read, analyze, and write data for advanced AI (data science analytics) implemented in MATLAB® programming language and trained offline in the ThingSpeak™ cloud.

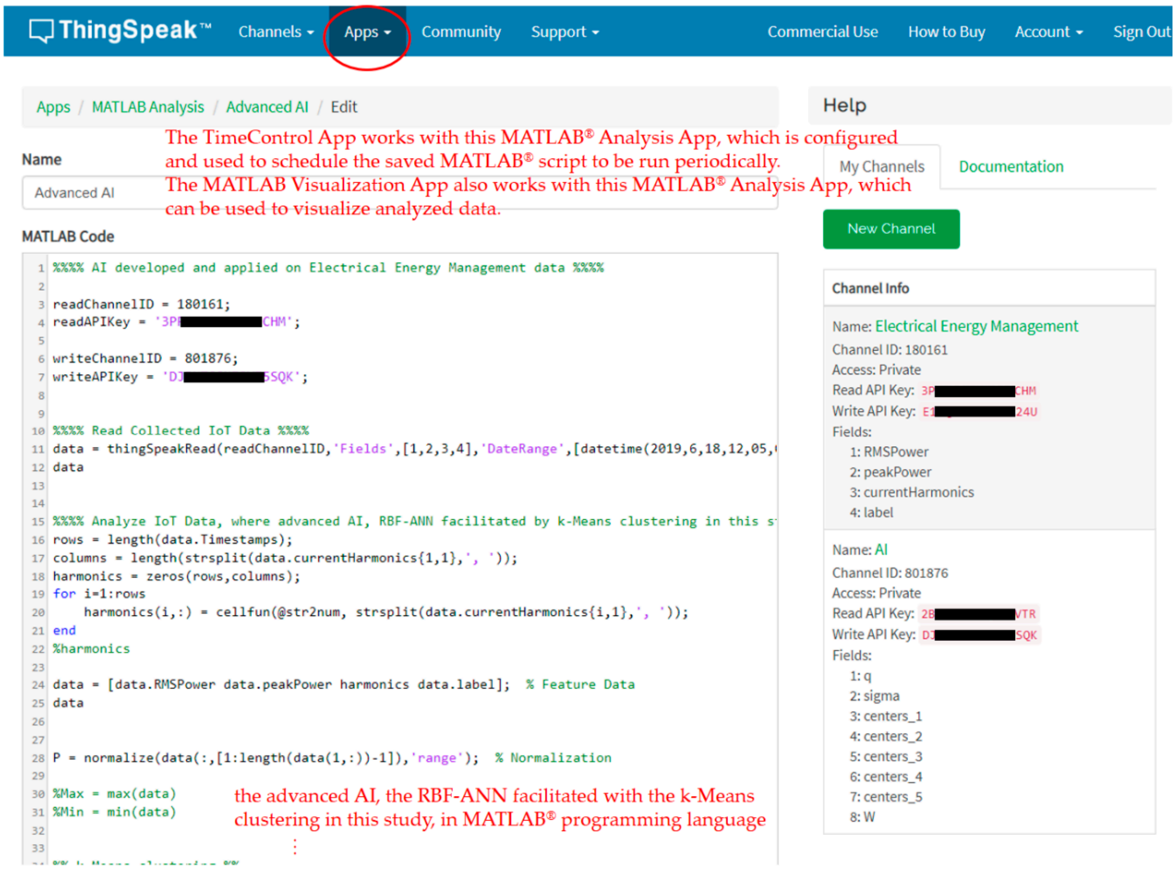
    (1.1) Read collected electrical feature data.

(1.2) Analyze collected electrical feature data accessed from the fields of the ThingSpeak™ channel, electrical energy management, where advanced AI, the RBF-ANN facilitated with k-means clustering, is investigated, implemented, and trained on the processed electrical feature data in the ThingSpeak™ cloud.

(1.3) Write the trained RBF-ANN facilitated with k-means clustering to the fields of the ThingSpeak™ channel, where advanced AI, the trained k-means clustering-combined RBF-ANN, is retrieved by the developed prototype via the Internet (refer to Figure 7c) and autonomously and automatically deployed onsite as edge analytics for on-demand data science analytics for DSM.

(2) Use the TimeControl app [66,67] to schedule the coded MATLAB® script to regularly train advanced AI, the RBF-ANN facilitated with k-means clustering, for model updates. The advanced AI trained and scheduled in the ThingSpeak™ cloud is remodeled under a specific date range of collected (historical) electrical feature data learned with timestamps in a regular periodic manner for model updates (historical electrical feature data can be weighed in chronological order for user-centric IoT-oriented applications in DSM [8] because there are more non-working than working hours in a week, activities of daily living by occupants related to electricity can lead to more electrical energy wasted during non-working hours than consumed during working hours [68–70]).

(3) Use the MATLAB® Visualization app [66,67] to visualize analyzed data (optional).



The TimeControl App works with this MATLAB® Analysis App, which is configured and used to schedule the saved MATLAB® script to be run periodically.
The MATLAB Visualization App also works with this MATLAB® Analysis App, which can be used to visualize analyzed data.

Name: Advanced AI

MATLAB Code

```
%%%% AI developed and applied on Electrical Energy Management data %%%%

readChannelID = 180161;
readAPIKey = '3P[redacted]CHM';

writeChannelID = 801876;
writeAPIKey = 'DJ[redacted]5SQK';


%%%% Read Collected IoT Data %%%%
data = thingSpeakRead(readChannelID,'Fields',[1,2,3,4],'DateRange',[datetime(2019,6,18,12,05,
data


%%%% Analyze IoT Data, where advanced AI, RBF-ANN facilitated by k-Means clustering in this s
rows = length(data.Timestamps);
columns = length(strsplit(data.currentHarmonics{1,1},', '));
harmonics = zeros(rows,columns);
for i=1:rows
    harmonics(i,:) = cellfun(@str2num, strsplit(data.currentHarmonics{i,1},', '));
end
%harmonics

data = [data.RMSPower data.peakPower harmonics data.label];  % Feature Data
data


P = normalize(data(:,[1:length(data(1,:))-1]),'range');  % Normalization

%Max = max(data)
%Min = min(data)


```

the advanced AI, the RBF-ANN facilitated with the k-Means clustering in this study, in MATLAB® programming language
⋮

Channel Info

Name: Electrical Energy Management
Channel ID: 180161
Access: Private
Read API Key: 3P[redacted]CHM
Write API Key: E1[redacted]24U
Fields:
1: RMSPower
2: peakPower
3: currentHarmonics
4: label

Name: AI
Channel ID: 801876
Access: Private
Read API Key: 2B[redacted]VTR
Write API Key: DJ[redacted]SQK
Fields:
1: q
2: sigma
3: centers_1
4: centers_2
5: centers_3
6: centers_4
7: centers_5
8: W

Figure 11. MATLAB® script coded for advanced AI in this experiment.

In this study, in the k-means clustering used to initialize the RBF-ANN, the specified number of cluster centers *c* was 5, the maximum number of iterations was 1500, and the (squared) Euclidean distance metric in which data clustered together have the least Euclidean distance was adopted. A 12-5-1 network structure of the RBF-ANN was structured, and its weighting connections, *W* in Equation (6), were trained offline in the ThingSpeak™ cloud, where the number of Gaussian basis functions of the RBF-ANN in Equation (1), $q(=c)$, was 5. The cluster centers found through the k-means clustering process and used to allocate the center parameters of the Gaussian basis functions of the RBF-ANN in Equation (1) are shown in Table 5.

Table 5. Center parameters found through k-means clustering are used to characterize the RBF-ANN of Equation (1), where electrical feature data are normalized in this experiment.

	P	**Turn-on transient power**	**DC offset**	**Fundamental harmonic (60 Hz)**	**2nd-order harmonic**	**3rd-order harmonic**
Cluster center 1	0.394	0.387	0.344	0.442	0.985	0.883
	4th-order harmonic	**5th-order harmonic**	**6th-order harmonic**	**7th-order harmonic**	**8th-order harmonic**	**9th order harmonic**
	0.903	0.676	0.865	0.529	0.829	0
	P	**Turn-on transient power**	**DC offset**	**Fundamental harmonic (60 Hz)**	**2nd-order harmonic**	**3rd-order harmonic**
Cluster center 2	0.997	0.984	0.684	0.990	0.022	0.152
	4th-order harmonic	**5th-order harmonic**	**6th-order harmonic**	**7th-order harmonic**	**8th-order harmonic**	**9th-order harmonic**
	0.067	0.611	0.024	1.000	0.056	0.333
	P	**Turn-on transient power**	**DC offset**	**Fundamental harmonic (60 Hz)**	**2nd-order harmonic**	**3rd-order harmonic**
Cluster center 3	0.048	0.050	0.071	0.059	0	0.168
	4th-order harmonic	**5th-order harmonic**	**6th-order harmonic**	**7th-order harmonic**	**8th-order harmonic**	**9th-order harmonic**
	0	0	0	0	0	0
	P	**Turn-on transient power**	**DC offset**	**Fundamental harmonic (60 Hz)**	**2nd-order harmonic**	**3rd-order harmonic**
Cluster center 4	0.391	1.000	0.474	0.437	0.970	0.818
	4th-order harmonic	**5th-order harmonic**	**6th-order harmonic**	**7th-order harmonic**	**8th-order harmonic**	**9th-order harmonic**
	0.900	0.667	0.857	0.500	0.667	0
	P	**Turn-on transient power**	**DC offset**	**Fundamental harmonic (60-Hz)**	**2nd-order harmonic**	**3rd-order harmonic**
Cluster center 5	0	0	0.042	0	0	0
	4th-order harmonic	**5th-order harmonic**	**6th-order harmonic**	**7th-order harmonic**	**8th-order harmonic**	**9th-order harmonic**
	0	0	0	0	0	0

The spread parameter of the Gaussian basis functions of the RBF-ANN that were computed by Equation (7) was 0.42. The input neurons of the RBF-ANN facilitated by the k-means clustering directly convey the normalized electrical features to the neurons in the hidden layer, where its output neuron is labeled as {0 (none), 1, 2, 3,..., N (total number of electrical appliances monitored and identified)}, normalized, and represented as a load indicator to indicate the identified electrical appliance that is present and active.

The RBF-ANN initialized by k-means clustering was well trained by Equation (6), where λ was 0.05 in the ThingSpeak™ cloud. In this experiment, the collected training dataset, electrical features used as input variables of the RBF-ANN, was normalized so that its range is in the interval [0, 1]. Table 6 shows the well-trained RBF-ANN, W, in this study.

Table 6. *W*.

W	0.478
	0.901
	4.966
	0.320
	−4.445

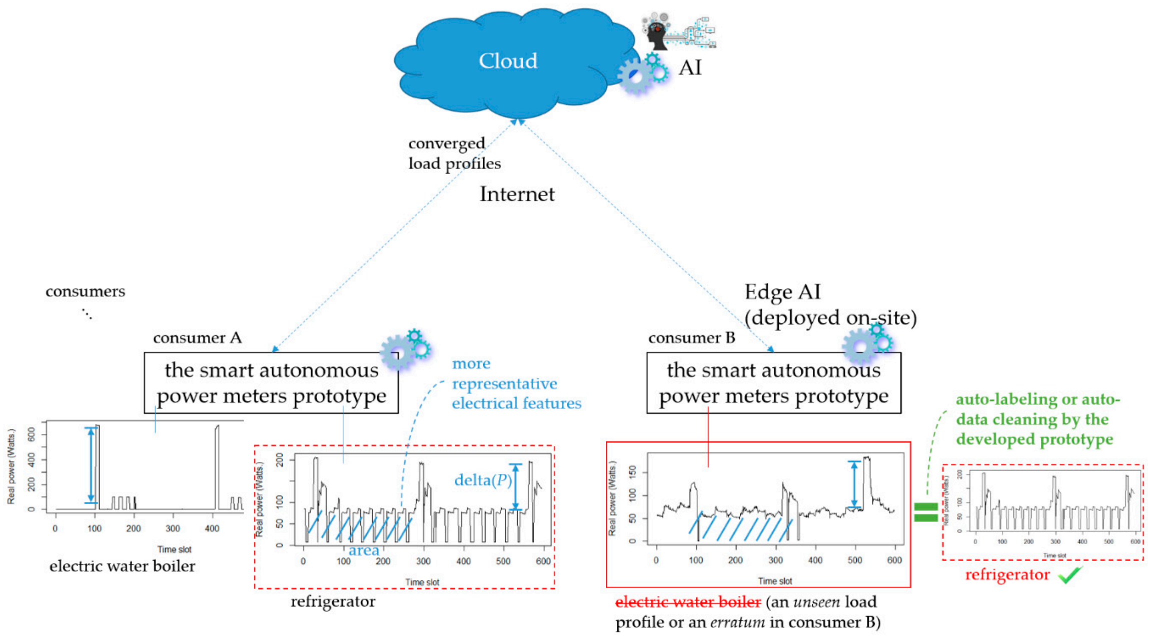
The well-trained and updated advanced AI in Tables 5 and 6 was autonomously and automatically deployed onsite on (ported on) the developed prototype via the Internet [71,72], where: (1) present time was retrieved from the RTC chip and recognized by the prototype for autonomous AI updates, and (2) the ThingSpeak Communication library [72] was included in the programmed Arduino sketch in Figure 7b and used to read the newest trained advanced AI entry from the fields of the created AI channel in the ThingSpeak™ cloud for the latest advanced AI update. The ThingSpeak Communication library enables an Arduino MCU or compatible hardware to read data from or write data to the ThingSpeak™ cloud. The latest advanced AI update was autonomously and automatically deployed onsite on (ported on) the prototype via the Internet as edge analytics for DSM in this study (refer to Figure 7c). ArduinoJson [73], a C++ JavaScript® Object Notation (JSON) [74,75] library for Arduino® and IoT, instead of the ThingSpeak Communication library, can be included in the programmed Arduino sketch and used to parse the responded latest JSON-formatted advanced AI.

Finally, the well-trained RBF-ANN facilitated with k-means clustering (shown in Tables 5 and 6) was autonomously and automatically deployed onsite and run on the Arduino MEGA 2560 MCU as edge analytics for online load identification in DSM, where the programmed Arduino sketch used 43,744 bytes (17%) of the maximum program storage space of 253,952 bytes; the global variables used 5622 bytes (68%) of the maximum dynamic memory of 8192 bytes, leaving 2570 bytes for local variables. Load identification in DSM in this study was performed by the developed prototype onsite and online. Table 7 shows the overall load identification rate obtained in this study: 94.26% (a total of 122 data instances are identified, and seven data instances are misidentified). An analysis of the k-means clustering (k = 5) combined with RBF-ANN with no regularization risk functional ($\lambda = 0$) in Equation (6) was conducted and compared with k-means clustering (k = 5) combined with RBF-ANN with a regularization risk functional ($\lambda = 0.05$) in Equation (6), as shown in Table 7; the overall load identification rate was improved by 5.74%.

Table 7. Overall load identification rate achieved in this study.

	K-means clustering (k = 5) combined with RBF-ANN ($\lambda = 0.05$) well trained in the cloud is autonomously and automatically deployed as edge analytics on the developed prototype for onsite and online load identification in DSM	**K-means clustering (k = 5) combined with RBF-ANN ($\lambda = 0$)**	**Overall load identification rate improvement (%)**
Overall load identification rate (%)	94.26	88.52	+5.74

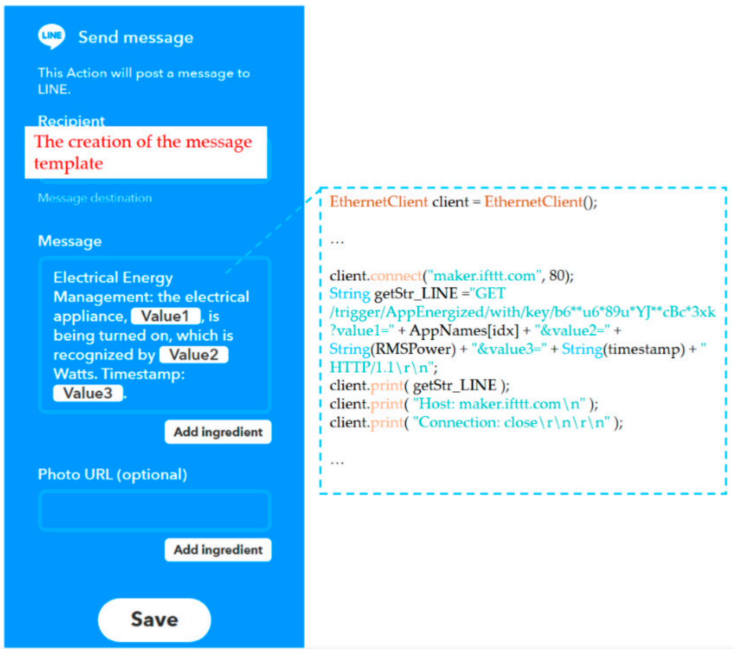
Figure 12 demonstrates auto-labeling or auto-data cleaning by means of online load identification performed onsite on the developed prototype through converged analytics (more IoT service-oriented applications for DSM could build upon this demonstration; also refer to Figure 2). In Figure 12, two more different types of electrical home appliances, a refrigerator and an electric water boiler, can be monitored by the developed prototype considering more representative extracted electrical features, where an unseen load profile by a new appliance monitored in a practical field (consumer B) can be identified through converged analytics (leveraging consumer A to provide data insights for consumer B) by the developed prototype as edge analytics (edge AI).



Cloud
AI
converged load profiles
Internet
consumers
consumer A
the smart autonomous power meters prototype
more representative electrical features
Edge AI (deployed on-site)
consumer B
the smart autonomous power meters prototype
auto-labeling or auto-data cleaning by the developed prototype
Real power (Watts)
Time slot
electric water boiler
area
delta(P)
refrigerator
~~electric water boiler~~ (an *unseen* load profile or an *erratum* in consumer B)
refrigerator ✓

Figure 12. Demonstration of auto-labeling or auto-data cleaning by means of online load identification performed onsite on the developed prototype through converged analytics (as edge-to-edge connectivity).

Regarding push notification service in the fog-cloud analytics architecture in this study, a practical paradigm of IFTTT [37] with Webhooks [38] was conducted, where an IFTTT rule with Webhooks publishing a new trigger received or an action made was created for LINE-Notify mobile devices, as illustrated in Figure 13. Figure 14 shows a load event message received by a LINE-Notify mobile phone over the IFTTT paradigm with Webhooks for load identification in DSM in this study. As shown in Figure 13, the sent LINE message was received when an appliance event, the monitored hair dryer energized and identified in this experiment, identified by the developed prototype with well-trained advanced AI was triggered and published by Webhooks [38].



Electrical Energy Management: the electrical appliance, Value1 , is being turned on, which is recognized by Value2 Watts. Timestamp: Value3 .

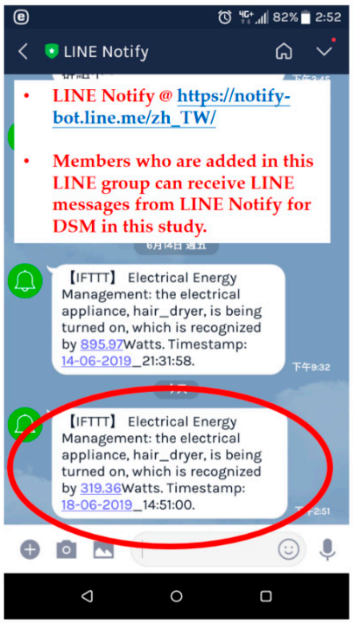
```
EthernetClient client = EthernetClient();

...

client.connect("maker.ifttt.com", 80);
String getStr_LINE ="GET
/trigger/AppEnergized/with/key/b6**u6*89u*YJ**cBc*3xk
?value1=" + AppNames[idx] + "&value2=" +
String(RMSPower) + "&value3=" + String(timestamp) + "
HTTP/1.1\r\n";
client.print( getStr_LINE );
client.print( "Host: maker.ifttt.com\n" );
client.print( "Connection: close\r\n\r\n" );

...
```

Figure 13. Illustration of the creation of an If-This-Then-That (IFTTT) rule with Webhooks in this experiment.



- LINE Notify @ https://notify-bot.line.me/zh_TW/
- Members who are added in this LINE group can receive LINE messages from LINE Notify for DSM in this study.

【IFTTT】 Electrical Energy Management: the electrical appliance, hair_dryer, is being turned on, which is recognized by 895.97Watts. Timestamp: 14-06-2019_21:31:58.

下午9:32

【IFTTT】 Electrical Energy Management: the electrical appliance, hair_dryer, is being turned on, which is recognized by 319.36Watts. Timestamp: 18-06-2019_14:51:00.

Figure 14. Load event message received by a LINE-Notify mobile phone over the IFTTT paradigm with Webhooks. A prespecified appliance event was identified and triggered for the monitored hair dryer.

3.2. Discussion

Figure 7 shows the proof-of-concept demonstration of the developed smart autonomous time- and frequency-domain analysis current sensor-based power meter prototype as edge analytics in an edge analytics-based AIoT architecture launched with an open and powerful cloud analytics platform. The prototype is based on an Arduino MEGA 2560 MCU mounted with a WIZNet W5100 hardwired TCP/IP embedded Ethernet shield; the configured cloud analytics platform is based on ThingSpeak™ with MATLAB® analytics. Also, the k-means clustering combined with RBF-ANN, advanced AI, is applied on electrical features, P, turn-on transient power, and current harmonics, where it is trained offline in the ThingSpeak™ cloud and autonomously and automatically deployed onsite on the developed prototype. Table 7 shows the overall load identification rate of 94.26% achieved and improved by 5.74% in that the regularization risk functional (λ = 0.05) is considered by k-means clustering combined with RBF-ANN (k = 5). Different types of electrical home appliances can be monitored by the developed prototype, as seen in Figures 2 and 12. More IoT service-oriented applications for DSM in a smart grid could be based on it. A digital signal processing (DSP) MCU-versioned powerful smart autonomous power meter prototype can be conducted alternatively and developed with the functionalities of the developed prototype in the described architecture over fog-cloud analytics in this study, which is given in Figure 15. In Figure 15, a Texas Instruments (TI)™ DSP MCU integrates (1) a TI™ MSP430 MCU as an e-meter for current and voltage measurements with (2) an IoT SoC/TI^{TM} Internet-on-a-chip Wi-Fi MCU as a portfolio of cloud integration with Internet connectivity (the REST API).

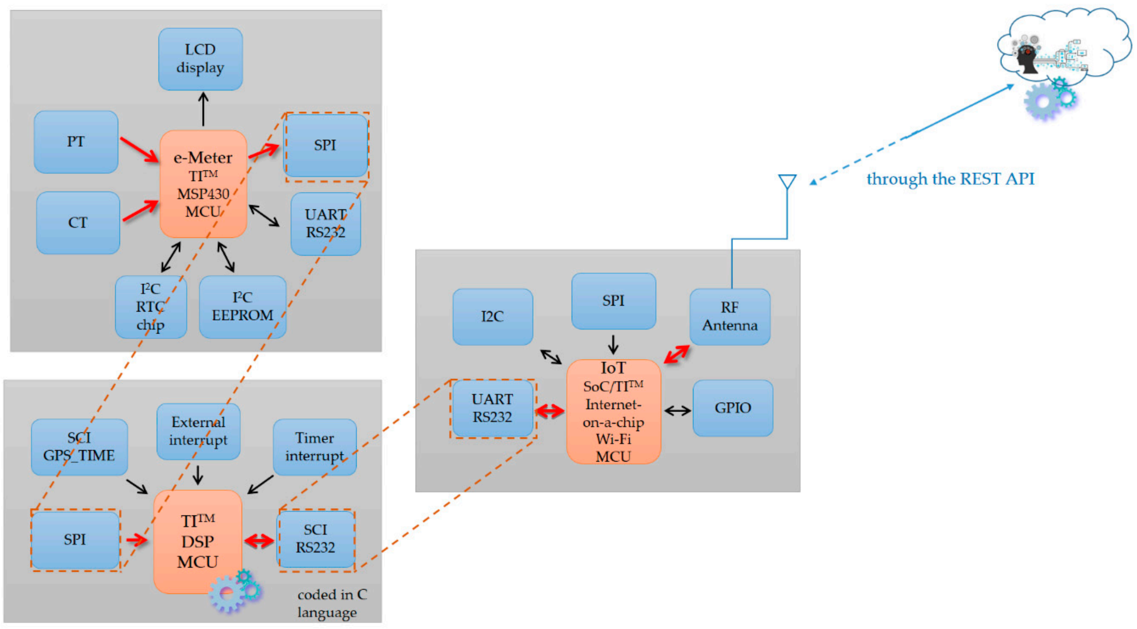


Figure 15. DSP MCU-versioned powerful smart autonomous power meter prototype.

4. Conclusions and Future Work

A smart grid is an innovative electrical energy network with two-way communication that can improve the reliability and flexibility of conventional power grids based on DSM. Fog-cloud analytics is extremely important to reduce network latency, conserve network bandwidth (throughput), and enable innovative AIoT applications. DSM, an essential part of a smart grid, enhances the efficiency, reliability, and flexibility of a traditional power grid upgraded to meet continuously increasing electrical energy demands by consumers. In this study, we designed and implemented a smart autonomous time- and frequency-domain analysis current sensor-based power meter prototype as edge analytics in an edge analytics-based AIoT architecture launched with an open and powerful cloud analytics platform for DSM in a smart grid. The prototype is based on an Arduino MEGA 2560 MCU mounted with a WIZNet W5100 hardwired TCP/IP embedded Ethernet shield; the cloud analytics platform is based on ThingSpeak™ with MATLAB® analytics. Also, advanced AI, the k-means clustering combined with RBF-ANN presented in this study, is: (1) applied on electrical features, P, turn-on transient power, and current harmonics; (2) trained offline in the ThingSpeak™ cloud; and (3) autonomously and automatically deployed onsite on the developed prototype. As demonstrated and reported in this study, an overall load identification rate of 94.26%, where the regularization risk functional is considered, is achieved for auto-labeling or auto-data cleaning by means of online load identification performed onsite on the developed prototype through converged analytics. More IoT service-oriented applications for DSM could build upon it. The prototype designed and implemented to collaborate with the configured ThingSpeak™ cloud over fog-cloud analytics for DSM in this study is feasible and workable.

An ensemble-based AI model such as multi-label classification based on random forest algorithms [76] will be developed and evaluated in the future, as various AI algorithms are effective approaches that can be used for load classification and long short-term load prediction in electrical energy management [77,78]. In the future, the developed smart autonomous power meter prototype in this study will be investigated for prognostics and health management in industry 4.0, where the availability and efficiency of equipment monitored by the prototype further investigated will be assessed. Electric energy consumption data from all consumers accumulate as Big Data (which is very computationally intense) in a power utility's cloud server, which will also be investigated and addressed through modern graphics processing unit technology.

Author Contributions: Y.-Y.C. and Y.-H.L. conceived, designed, and performed the experiments; Y.-H.L. wrote the paper; Y.-H.L. and Y.-Y.C. contributed related experimental tools/materials and analyzed the experimental data.

Funding: This study was supported in part by the Ministry of Science and Technology, Taiwan, under grant nos. MOST 108-2221-E-131-029- and MOST 108-3116-F-006-008-CC2.

Acknowledgments: The authors would like to sincerely thank the reviewers for their valuable comments and suggestions on this study.

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

References

1. Esther, B.P.; Kumar, K.S. A survey on residential Demand Side Management architecture, approaches, optimization models and methods. *Renew. Sustain. Energy Rev.* **2016**, *59*, 342–351. [CrossRef]
2. Javaid, N.; Ahmed, F.; Ullah, I.; Abid, S.; Abdul, W.; Alamri, A.; Almogren, A.S. Towards cost and comfort based hybrid optimization for residential load scheduling in a smart grid. *Energies* **2017**, *10*, 1546. [CrossRef]
3. Arun, S.L.; Selvan, M.P. Intelligent residential energy management system for dynamic demand response in smart buildings. *IEEE Syst. J.* **2018**, *12*, 1329–1340. [CrossRef]
4. Hussain, H.M.; Javaid, N.; Iqbal, S.; Hasan, Q.U.; Aurangzeb, K.; Alhussein, M. An efficient demand side management system with a new optimized home energy management controller in smart grid. *Energies* **2018**, *11*, 190. [CrossRef]
5. Gellings, C.W. The concept of demand-side management for electric utilities. *Proc. IEEE* **1985**, *73*, 1468–1470. [CrossRef]
6. Bozchalui, M.C.; Hashmi, S.A.; Hassen, H.; Cañizares, C.A.; Bhattacharya, K. Optimal operation of residential energy hubs in smart grids. *IEEE Trans. Smart Grid* **2012**, *3*, 1755–1766. [CrossRef]
7. Rosselló-Busquet, A.; Soler, J. Towards efficient energy management: Defining HEMS and smart grid objectives. *Int. J. Adv. Telecommun.* **2011**, *4*, 249–263.
8. Lin, Y.H.; Tsai, M.S. An advanced home energy management system facilitated by nonintrusive load monitoring with automated multiobjective power scheduling. *IEEE Trans. Smart Grid* **2015**, *6*, 1839–1851. [CrossRef]
9. Lin, Y.H.; Hu, Y.C. Residential consumer-centric demand-side management based on energy disaggregation -piloting constrained swarm intelligence: Towards edge computing. *Sensors* **2018**, *18*, 1365. [CrossRef]
10. Lin, Y.H. Design and implementation of an IoT-oriented energy management system based on non-intrusive and self-organizing neuro-fuzzy classification as an electrical energy audit in smart homes. *Appl. Sci.* **2018**, *8*, 2337. [CrossRef]
11. Taoa, M.; Zuo, J.; Liu, Z.; Castiglione, A.; Palmieri, F. Multi-layer cloud architectural model and ontology-based security service frame-work for IoT-based smart homes. *Future Generat. Comput. Syst.* **2018**, *78*, 1040–1051. [CrossRef]
12. Mengistu, M.A.; Girmay, A.A.; Camarda, C.; Acquaviva, A.; Patti, E. A cloud-based on-line disaggregation algorithm for home appliance loads. *IEEE Trans. Smart Grid* **2019**, *10*, 3430–3439. [CrossRef]
13. Tsai, P.H.; Hong, H.J.; Cheng, A.C.; Hsu, C.H. Distributed analytics in fog computing platforms using tensorflow and kubernetes. In Proceedings of the 2017 19th Asia-Pacific Network Operations and Management Symposium (APNOMS), Seoul, Korea, 27–29 September 2017; pp. 145–150.
14. Markakis, E.K.; Karras, K.; Zotos, N.; Sideris, A.; Moysiadis, T.; Corsaro, A.; Alexiou, G.; Skianis, C.; Mastorakis, G.; Mavromoustakis, C.X.; et al. EXEGESIS: Extreme edge resource harvesting for a virtualized fog environment. *IEEE Commun. Mag.* **2017**, *55*, 173–179. [CrossRef]
15. Froiz-Míguez, I.; Fernández-Caramés, T.M.; Fraga-Lamas, P.; Castedo, L. Design, implementation and practical evaluation of an IoT home automation system for fog computing applications based on MQTT and ZigBee-WiFi sensor nodes. *Sensors* **2018**, *18*, 2660. [CrossRef] [PubMed]
16. Jang, I.; Lee, D.; Choi, J.; Son, Y. An approach to share self-taught knowledge between home IoT devices at the edge. *Sensors* **2019**, *19*, 833. [CrossRef] [PubMed]
17. Yazici, M.T.; Basurra, S.; Gaber, M.M. Edge machine learning: Enabling smart Internet of Things applications. *Big Data Cogn. Comput.* **2018**, *2*, 26. [CrossRef]

18. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the Internet of Things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
19. Fernández-Caramés, T.M.; Fraga-Lamas, P.; Suárez-Albela, M.; Díaz-Bouza, M.A. A fog computing based cyber-physical system for the automation of pipe-related tasks in the Industry 4.0 shipyard. *Sensors* **2018**, *18*, 1961. [CrossRef]
20. Idrees, Z.; Zou, Z.; Zheng, L. Edge computing based IoT architecture for low cost air pollution monitoring systems: A comprehensive system analysis, design considerations & development. *Sensors* **2018**, *18*, 3021. [CrossRef]
21. Vidal-Pardo, A.; Pindado, S. Design and development of a 5-Channel Arduino-based data acquisition system (ABDAS) for experimental aerodynamics research. *Sensors* **2018**, *18*, 2382. [CrossRef]
22. Mnati, M.J.; Van den Bossche, A.; Chisab, R.F. A smart voltage and current monitoring system for three phase inverters using an Android smartphone application. *Sensors* **2017**, *17*, 872. [CrossRef]
23. Chen, Y.Y.; Lin, Y.H.; Kung, C.C.; Chung, M.H.; Yen, I.H. Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart homes. *Sensors* **2019**, *19*, 2047. [CrossRef] [PubMed]
24. Ruano, A.; Hernandez, A.; Ureña, J.; Ruano, M.; Garcia, J. NILM techniques for intelligent home energy management and ambient assisted living: A review. *Energies* **2019**, *12*, 2203. [CrossRef]
25. Hart, G.W. Non-intrusive appliance load monitoring. *IEEE Proc.* **1992**, *80*, 1870–1891. [CrossRef]
26. Zeifman, M.; Roth, K. Non-intrusive appliance load monitoring: Review and outlook. *IEEE Trans. Consum. Electron.* **2011**, *57*, 76–84. [CrossRef]
27. Zoha, A.; Gluhak, A.; Imran, M.A.; Rajasegarar, S. Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. *Sensors* **2012**, *12*, 16838–16866. [CrossRef] [PubMed]
28. Agyeman, K.A.; Han, S.; Han, S. Real-time recognition non-intrusive electrical appliance monitoring algorithm for a residential building energy management system. *Energies* **2015**, *8*, 9029–9048. [CrossRef]
29. Sarkar, S.; Chatterjee, S.; Misra, S. Assessment of the suitability of fog computing in the context of Internet of things. *IEEE Trans. Cloud Comput.* **2015**, *6*, 46–59. [CrossRef]
30. Tomovic, S.; Yoshigoe, K.; Maljevic, I.; Radusinovic, I. Software-defined fog network architecture for IoT. *Wirel. Pers. Commun.* **2017**, *92*, 181–196. [CrossRef]
31. Ain, Q.-U.; Iqbal, S.; Khan, S.A.; Malik, A.W.; Ahmad, I.; Javaid, N. IoT operating system based fuzzy inference system for home energy management system in smart buildings. *Sensors* **2018**, *18*, 2802. [CrossRef]
32. Jo, B.; Khan, R.M.A. An Internet of Things system for underground mine air quality pollutant prediction based on Azure machine learning. *Sensors* **2018**, *18*, 930. [CrossRef]
33. Mukherjee, M.; Shu, L.; Wang, D. Survey of fog computing: Fundamental, network applications, and research challenges. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 1826–1857. [CrossRef]
34. Amini, M.H.; Nabi, B.; Haghifam, M.R. Load management using multi-agent systems in smart distribution network. In Proceedings of the 2013 IEEE Power & Energy Society General Meeting, Vancouver, BC, Canada, 21–25 July 2013; pp. 1–5.
35. Bahrami, S.; Wong, V.W.S. An autonomous demand response program in smart grid with foresighted users. In Proceedings of the 2015 IEEE International Conference on Smart Grid Communications (SmartGridComm), Miami, FL, USA, 2–5 November 2015; pp. 205–210.
36. IoT Analytics—ThingSpeak Internet of Things. Available online: https://thingspeak.com/ (accessed on 12 December 2018).
37. Discover IFTTT and Applets—IFTTT (IFTTT). Available online: https://ifttt.com/discover (accessed on 6 January 2019).
38. Do More with Webhooks—IFTTT (Webhooks). Available online: https://ifttt.com/maker_webhooks (accessed on 6 January 2019).
39. Arduino. Arduino Mega 2560. 2014. Available online: https://www.arduino.cc/en/Guide/ArduinoMega2560 (accessed on 12 December 2018).
40. Arduino LLC. Language Reference. Available online: http://www.arduino.cc/en/Reference/HomePage (accessed on 2 October 2018).
41. ArduinoEthernetShieldV1—Arduino. Available online: https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1 (accessed on 12 December 2018).

42. Gao, L.; Wang, Z.X.; Zhou, J.L.; Zhang, C. Design of smart home system based on ZigBee technology and R&D for application. *Energy Power Eng.* **2016**, *8*, 13–22.
43. Ding, F.; Chen, X.; He, S.; Shou, G.; Zhang, Z.; Zhou, Y. Evaluation of a Wi-Fi signal based system for freeway traffic states monitoring: An exploratory field test. *Sensors* **2019**, *19*, 409. [CrossRef] [PubMed]
44. Kim, K.; Myung, H. Sensor node for remote monitoring of waterborne disease-causing bacteria. *Sensors* **2015**, *15*, 10569–10579. [CrossRef] [PubMed]
45. Strigaro, D.; Cannata, M.; Antonovic, M. Boosting a weather monitoring system in low income economies using open and non-conventional systems: Data quality analysis. *Sensors* **2019**, *19*, 1185. [CrossRef] [PubMed]
46. Lin, Y.H. Novel smart home system architecture facilitated with distributed and embedded flexible edge analytics in demand-side management. *Int. Trans. Electr. Energy Syst.* **2019**, *29*, e12014. [CrossRef]
47. Dinesh, C.; Nettasinghe, B.W.; Godaliyadda, R.I.; Ekanayake, M.P.B.; Ekanayake, J.; Wijayakulasooriya, J.V. Residential appliance identification based on spectral information of low frequency smart meter measurements. *IEEE Trans. Smart Grid* **2016**, *7*, 2781–2792. [CrossRef]
48. Cooley, J.W.; Tukey, J.W. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* **1965**, *19*, 297–301. [CrossRef]
49. Heideman, M.; Johnson, D.; Burrus, C. Gauss and the history of the fast Fourier transform. *IEEE ASSP Mag.* **1984**, *1*, 14–21. [CrossRef]
50. Viciana, E.; Alcayde, A.; Montoya, F.G.; Baños, R.; Arrabal-Campos, F.M.; Zapata-Sierra, A.; Manzano-Agugliaro, F. OpenZmeter: An efficient low-cost energy smart meter and power quality analyzer. *Sustainability* **2018**, *10*, 4038. [CrossRef]
51. MathWorks. Fast Fourier Transform (FFT). Available online: http://se.mathworks.com/help/matlab/math/fast-fourier-transform-fft.html (accessed on 2 June 2019).
52. kosme/arduinoFFT: Fast Fourier Transform for Arduino—GitHub. Available online: https://github.com/kosme/arduinoFFT (accessed on 7 June 2019).
53. Hann (Hanning) Window—MATLAB Hann—MathWorks. Available online: https://www.mathworks.com/help/signal/ref/hann.html (accessed on 7 June 2019).
54. Hjort, A.; Holmberg, M. *Measuring Mechanical Vibrations Using an Arduino as a slave I/O to an EPICS Control System*; The FREIA Report Series (Physics Project with a Research Connection 5 hp, 1FA605); Uppsala University: Uppsala, Sweden, 2015.
55. Arduino Playground—FlexiTimer2. Available online: https://playground.arduino.cc/Main/FlexiTimer2 (accessed on 20 April 2019).
56. Arduino Playground—MsTimer2. Available online: https://playground.arduino.cc/Main/MsTimer2 (accessed on 20 April 2019).
57. Kumar, S. *Neural Networks: A Classroom Approach*, International ed.; McGraw-Hill Education (Asis)/Tata McGraw-Hill Publishing Company Limited: New Delhi, India, 2005; pp. 304–329.
58. Lloyd, S.P. Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [CrossRef]
59. Wang, L.X. *A Course in Fuzzy Systems and Control*; IE-Paperback; Pearson Education Taiwan Ltd.: Taipei, Taiwan, 2005; pp. 342–347.
60. K-Means Clustering—MATLAB K-Means—MathWorks. Available online: https://www.mathworks.com/help/stats/kmeans.html (accessed on 27 June 2019).
61. Matrix Inverse—MATLAB Inv—MathWorks. Available online: https://www.mathworks.com/help/matlab/ref/inv.html (accessed on 27 June 2019).
62. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
63. Sudqi Khater, B.; Abdul Wahab, A.W.B.; Idris, M.Y.I.B.; Abdulla Hussain, M.; Ahmed Ibrahim, A. A lightweight perceptron-based intrusion detection system for fog computing. *Appl. Sci.* **2019**, *9*, 178. [CrossRef]
64. Moody, J.; Darken, C.J. Fast learning in network of locally-tuned processing units. *Neural Comput.* **1989**, *1*, 281–294. [CrossRef]
65. Wei, Y.; Wang, H.; Tsang, K.F.; Liu, Y.; Wu, C.K.; Zhu, H.; Chow, Y.-T.; Hung, F.H. Proximity environmental feature based tree health assessment scheme using Internet of Things and Machine Learning algorithm. *Sensors* **2019**, *19*, 3115. [CrossRef] [PubMed]
66. ThingSpeak (ThingSpeak Documentation—MathWorks). Available online: https://www.mathworks.com/help/thingspeak/ (accessed on 14 June 2019).

67. Get Started with ThingSpeak—MathWorks. Available online: https://www.mathworks.com/help/thingspeak/getting-started-with-thingspeak.html (accessed on 14 June 2019).
68. Masoso, O.T.; Grobler, L.J. The dark side of occupants' behaviour on building energy use. *Energy Build.* **2010**, *42*, 173–177. [CrossRef]
69. Rafsanjani, H.N.; Ahn, C.R.; Alahmad, M. A Review of approaches for sensing, understanding, and improving occupancy-related energy-use behaviors in commercial buildings. *Energies* **2015**, *8*, 10996–11029. [CrossRef]
70. Ahmed, N.; Levorato, M.; Li, G.P. Residential consumer-centric demand side management. *IEEE Trans. Smart Grid* **2017**, *9*, 4513–4524. [CrossRef]
71. REST API—MATLAB & Simulink—MathWorks. Available online: https://www.mathworks.com/help/thingspeak/rest-api.html (accessed on 15 June 2019).
72. Mathworks/Thingspeak-Arduino: ThingSpeak Communication Library for Arduino, ESP8266 and EPS32—GitHub. Available online: https://github.com/mathworks/thingspeak-arduino (accessed on 15 June 2019).
73. ArduinoJson: Efficient JSON Serialization for Embedded C++. Available online: https://arduinojson.org/ (accessed on 11 July 2019).
74. JSON Format—MATLAB & Simulink—MathWorks. Available online: https://www.mathworks.com/help/matlab/json-format.html (accessed on 5 July 2019).
75. Postman | API Development Environment. Available online: https://www.getpostman.com/ (accessed on 5 July 2019).
76. Wu, X.; Gao, Y.; Jiao, D. Multi-label classification based on random forest algorithm for non-intrusive load monitoring system. *Processes* **2019**, *7*, 337. [CrossRef]
77. Chakraborty, D.; Elzarka, H. Advanced machine learning techniques for building performance simulation: A comparative analysis. *J. Build. Perf. Simul.* **2019**, *12*, 193–207. [CrossRef]
78. Welikala, S.; Dinesh, C.; Ekanayake, M.P.B.; Godaliyadda, R.I.; Ekanayake, J. Incorporating appliance usage patterns for non-intrusive load monitoring and load forecasting. *IEEE Trans. Smart Grid* **2019**, *10*, 448–461. [CrossRef]