information

Article

Hardware-Based Emulator with Deep Learning Model for Building Energy Control and Prediction Based on Occupancy Sensors' Data

Zhijing Ye [1], Zheng O'Neill [2] and Fei Hu [1,*]

[1] Electrical and Computer Engineering Department, The University of Alabama, Tuscaloosa, AL 35487, USA; zye7@crimson.ua.edu
[2] Mechanical Engineering Department, Texas A&M University, College Station, TX 77843, USA; zoneill@tamu.edu
* Correspondence: fei@eng.ua.edu

Abstract: Heating, ventilation, and air conditioning (HVAC) is the largest source of residential energy consumption. Occupancy sensors' data can be used for HVAC control since it indicates the number of people in the building. HVAC and sensors form a typical cyber-physical system (CPS). In this paper, we aim to build a hardware-based emulation platform to study the occupancy data's features, which can be further extracted by using machine learning models. In particular, we propose two hardware-based emulators to investigate the use of wired/wireless communication interfaces for occupancy sensor-based building CPS control, and the use of deep learning to predict the building energy consumption with the sensor data. We hypothesize is that the building energy consumption may be predicted by using the occupancy data collected by the sensors, and question what type of prediction model should be used to accurately predict the energy load. Another hypothesis is that an in-lab hardware/software platform could be built to emulate the occupancy sensing process. The machine learning algorithms can then be used to analyze the energy load based on the sensing data. To test the emulator, the occupancy data from the sensors is used to predict energy consumption. The synchronization scheme between sensors and the HVAC server will be discussed. We have built two hardware/software emulation platforms to investigate the sensor/HVAC integration strategies, and used an enhanced deep learning model—which has sequence-to-sequence long short-term memory (Seq2Seq LSTM)—with an attention model to predict the building energy consumption with the preservation of the intrinsic patterns. Because the long-range temporal dependencies are captured, the Seq2Seq models may provide a higher accuracy by using LSTM architectures with encoder and decoder. Meanwhile, LSTMs can capture the temporal and spatial patterns of time series data. The attention model can highlight the most relevant input information in the energy prediction by allocating the attention weights. The communication overhead between the sensors and the HVAC control server can also be alleviated via the attention mechanism, which can automatically ignore the irrelevant information and amplify the relevant information during CNN training. Our experiments and performance analysis show that, compared with the traditional LSTM neural network, the performance of the proposed method has a 30% higher prediction accuracy.

Keywords: cyber-physical systems (CPS); heating; ventilation and cooling (HVAC); energy saving; real-time emulators; wireless sensor networks; sequence-to-sequence long short-term memory (Seq2Seq LSTM); self-attention models; convolutional neural network; short-term load forecasting

Citation: Ye, Z.; O'Neill, Z.; Hu, F. Hardware-Based Emulator with Deep Learning Model for Building Energy Control and Prediction Based on Occupancy Sensors' Data. *Information* **2021**, *12*, 499. https://doi.org/10.3390/info12120499

Academic Editors: Shahram Latifi and Doina Bein

Received: 11 October 2021
Accepted: 28 November 2021
Published: 1 December 2021

1. Introduction

The heating, ventilation, and air conditioning (HVAC) system provides thermal comfort and acceptable indoor air quality. It is the single largest contributor to a home's energy bills, accounting for 43% of residential energy consumption in the U.S., and 61% in Canada and the U.K., which have colder climates [1,2]. There has been some research focusing

on decreasing the energy consumption by 25% while turning off the HVAC system with nobody in the room [3], resulting in $15 being saved per month for the average U.S. household [1–3]. Therefore, it is necessary to develop different communication protocols to save energy and optimize the HVAC system, including the Automation Control Network (BACnet), Modbus, Local Operating Network (Lon Works) and Lon Talk.

Our research objectives include two aspects: (1) to build a low-cost hardware/software emulation platform to show the use of occupancy sensors for building energy analysis; (2) to use deep learning algorithms to predict the energy load in the next time window, based on the collected sensor data.

Our research principle includes two aspects, detailed below.

First, energy consumption is one of the biggest contributors to global warming, and most energy use is due to human activity; for example, electrical heating for commercial and residential buildings can consume almost half of the total energy production. Therefore, it is critical to manage energy use efficiently to reduce the negative environmental hazards, including carbon dioxide emissions. Energy systems often do not have dramatic changes in load supply, which makes it possible to perform a prediction of future loads with small errors. Due to the importance of energy scheduling and allocation, load prediction has attracted much attention from researchers.

Second, wireless sensors are widely used to collect energy-related data (such as the number of people in the building) due to their low cost and low power consumption. However, the popular uses of sensors also cause some issues, such as higher wireless transmission delay, poor RF transmission quality, and higher energy consumption. Meanwhile, some problems arise due to the reduction in the cost of a single sensor, including sensing errors, missing data, etc. To recover or correct the abnormal data, some features should be extracted when the data is being preprocessed. With data pre-processing, the data collection process can be optimized and the correlation between features may also be reduced. Therefore, the energy prediction models should overcome the above data collection issues.

There are two main scenarios: energy saving control in (1) commercial or (2) residential buildings. These will be used to evaluate the performance of the communication between sensors and the HVAC system. A commercial building consists of multiple single offices and classrooms, while residential buildings refer to individual houses. Various sensors are used to analyze the occupancy numbers for each room. There are two communication methods including wired and wireless, with Ethernet or USB connection being utilized for the former, and Wi-Fi and Bluetooth being used for the latter.

The interactions between occupancy sensors and HVAC controllers have the essential features of a typical cyber-physical system (CPS): (1) Physical-to-Cyber: The physical objects (occupants in the building) should be calculated to control the temperature levels of the HVAC systems in real-time. Such physical parameters can be captured by using cyber units; for example, some hardware with computation functions could collect occupancy numbers in any region/room of a building. Moreover, both wired and wireless sensors can be used. In this case, any required physical status can be collected by different sensors and sent to an HVAC server in real-time. (2) Cyber-to-Physical: The cyber units can be used to change the physical world. The occupancy allocation or distribution may be figured out by processing the sensor data. Therefore, it is possible to modify the physical objects based on the collected cyber data. For example, the fan/air circulation levels of HVAC units could be tuned here.

The architecture of such a building CPS is shown in Figure 1. Different sensors can be used to help save energy by controlling the room temperature efficiently. An occupancy sensor may consist of a timer, as well as a human motion detector. Some other sensors can measure CO_2 levels, acoustic signals, human images, and more.

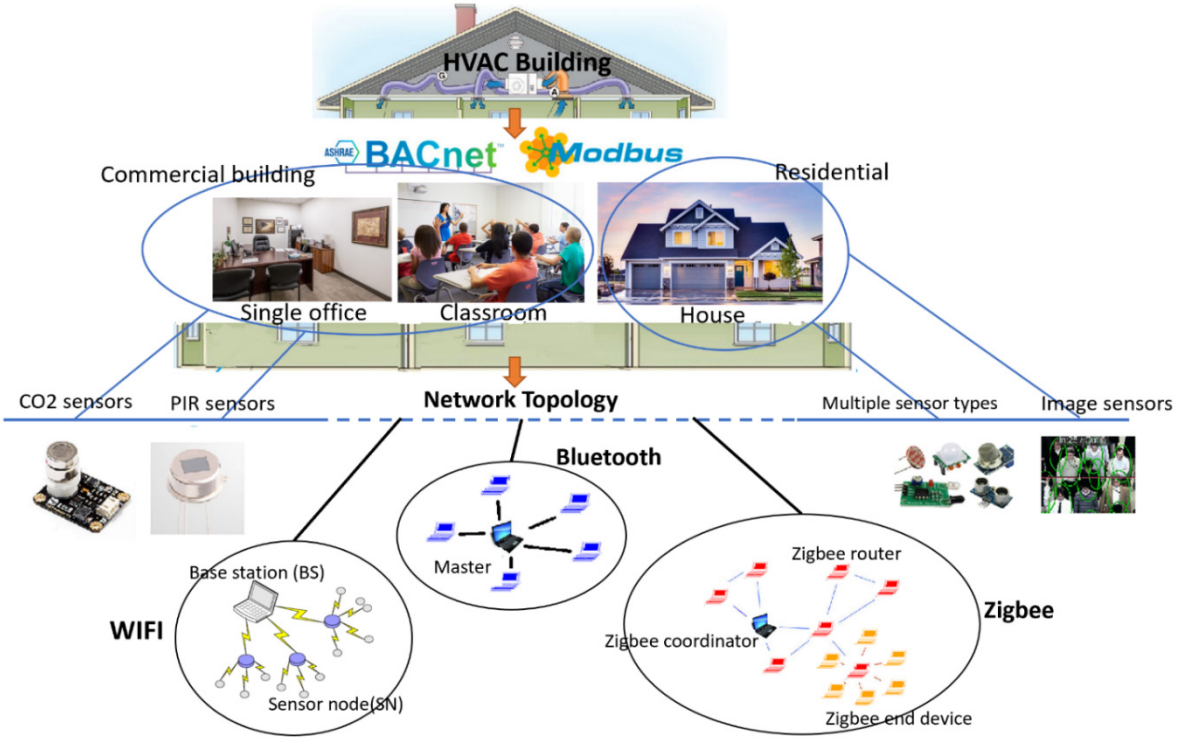

Figure 1. Overview of communication between the HVAC system and sensors.

However, the CPS emulators for the occupancy-sensor-based building energy control system are not available yet, and only some pure software-based simulators without hardware units have been built. In this paper, two hardware emulators are illustrated, and the Raspberry Pi wireless board is used as a communication platform between hardware units, such as PIR sensors, and the thermostat. Our emulation platform shows the CPS interactions with the HVAC/sensor communication control within residential and commercial building environments.

There are three features for our emulator designs:

First, different types of sensors, such as acoustic, CO_2, and PIR sensors, can be used in HVAC systems based on various application requirements. Wired, as well as wireless, communication links can be established here. Moreover, both analog and digital sensors are used for wired sensors. The sensors can send data back to the server through RF (radio frequency) waves or USB/Ethernet interfaces. Some wireless RF communication methods include Bluetooth, ZigBee, and Wi-Fi. Additionally, TCP/IP protocols are recommended for the use of communication between units. Compared with other products shown in Table 1, our entire wired and wireless hardware platform is portable and has a lower cost. The occupancy data can be collected from different sensors.

Second, various sensor deployments are considered for different building layouts. There are several attributes to modify for the deployment styles, including sensor locations such as in the door, walls, or ceilings, sensor density, and deployment topologies, such as a chain, mesh, or grid structure. Moreover, the multi-hop data relay protocols will also be used among sensors for transmission between sensors.

Third, sensor data fusion/processing is important since we need to establish an intelligent building energy-saving control system. In this case, it is easy to learn the occupancy distributions based on the sensor data, which is analyzed by the deep learning model. Specifically, deep learning models are used to predict the energy-saving patterns for the HVAC control system. Meanwhile, the occupancy number can be calculated for every room, after which the control signals are sent to the HVAC system in real-time. The occupancy number estimation is based on the coverage sizes of multiple sensors and considers overlapped sensing areas.

Table 1. Commercial people-counting sensors on the market.

Product Company	Sensor Type	Power	Wired/Wireless	Costs
Axper	3D camera	Powered via the network (PoE)	10/100 Mb Ethernet TCP/IP, DHCP	>$2000
SenSource	3D video	Ethernet connection, PoE	Need cable	$900–$1500
Bea America (LZR-SIGMA)	laser sensor	External DC power or PoE	cloud-based connectivity, HTTP(S) and (S)FTP	>$1500
Irisys Vector 4D	video camera	PoE	IPV4, DHCP, IP protocols supported by HTTPS	$800–$1800
Traf-Sys Gazelle People Counter	Video and thermal	PoE	Built-in Ethernet connectivity	>$800

Instead of processing all the raw sensor data, the data transformation models can be used to find the occupants' entrance/exit trends. Therefore, data processing could collect occupant information more efficiently and be used in real-time after adjusting the HVAC control.

Two different emulators of the building energy-saving systems have been established. The whole process is shown in Figure 2. The Raspberry Pi is utilized for the hardware emulator for real-time sensor data communication, as well as for aggregation functions. Afterwards, occupancy numbers are computed in the *residential* building. In this case, we can also get the energy prediction of the house with the deep learning model, and its airflow and temperature can be real-time changed.

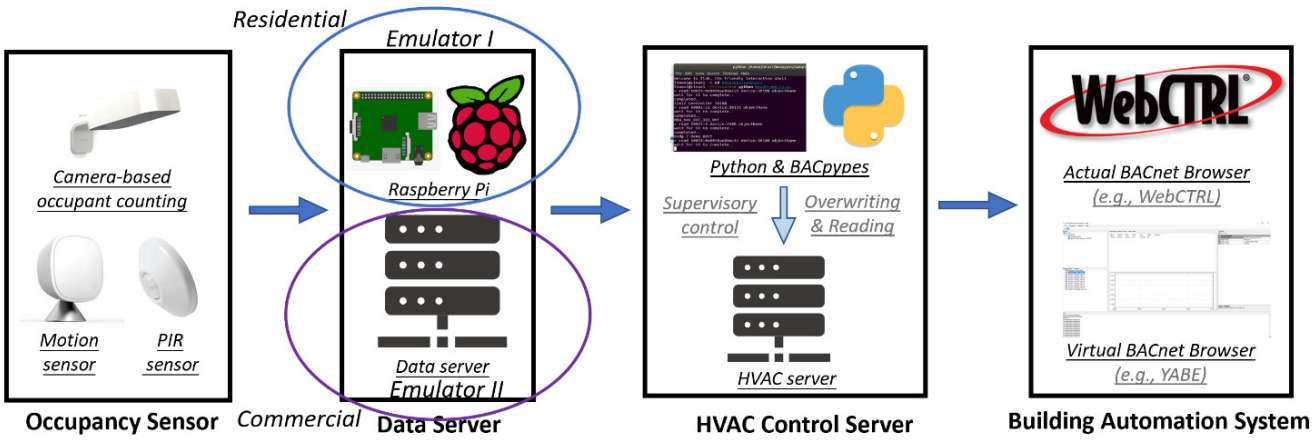

Figure 2. The whole process of the emulator.

The second emulator utilizes the BACnet protocol in the commercial building. In this case, it is easier to send the data from the data center to the HVAC control center. Wireless communication between the data collection center and control server could also be established with the BACnet protocol.

After extracting the data, the main uses of sensors also cause some issues, such as higher wireless transmission delay, poor RF transmission quality, and higher energy consumption. Meanwhile, due to the reduction of the cost of a single sensor, some problems arise, including hardware failure, data transmission failure, missing data, etc. To recover or correct the abnormal data, some features should be extracted when the data is being preprocessed.

Due to some stability of the surrounding environment (such as temperature consistency in the same room), there exist some correlations among sensor data. However, it is difficult to extract useful characteristics from the correlated data if we just use conventional machine learning methods. Therefore, in this paper, we propose to use deep learning to extract the relevant features. In particular, the convolutional neural network (CNN) can be utilized to process different types of data. CNN has translation invariance by calculating convolutional characteristics among data.

In this research, both the sensor data and energy consumption data from smart meters in the building will be used to build a load forecasting model. Compared with traditional machine learning models, such as Decision Trees and Support Vector Machine (SVM), deep neural networks (DNN) can achieve better prediction performance due to their powerful pattern extraction capability from raw data without the need for data preprocessing. However, general DNN can only take the *current* input to form the predictions. Therefore, some time-dependent features cannot be captured by DNN. The recurrent neural network (RNN) can be used to create a directed graph among nodes so that it is able to capture the time dependencies. Such a directed graph can be used to connect current inputs together with previous inputs. Thus, RNN could capture the dynamic behaviors of data. Additionally, long- and short-term memory (LSTM) is an improved model of RNN. Cell structure is used in the LSTM, and three gates in the cell (i.e., input gate, forget gate, and output gate) are used to determine whether data is chosen or discarded.

LSTMs have the advantage in analyzing temporal behaviors, and a Sequence-to-Sequence LSTM (Seq2Seq LSTM) is introduced in language translation models for powerful context capture. The Seq2Seq model structure consists of an encoder LSTM and a decoder LSTM. In this structure, data is encoded into a fixed-length input feature vector by an encoder, while the output data can be generated through the decoder. To find the most relevant information for each decoder output, *attention* mechanisms are introduced. The encoder can use an attention model to compress the required data into a fixed-size vector.

This paper is an extension of our conference paper [4]. It proposes the use of Seq2Seq LSTM with an attention mechanism for energy prediction. Seq2Seq LSTM is adopted for energy prediction and deals with medium-term data independence. The attention mechanism is used to obtain the most relevant information among input data. It can also ease the connection between the encoder and decoder.

Some major acronyms we used in this article include: cyber-physical systems (CPS); heating, ventilation, and cooling (HVAC); sequence-to-sequence long short-term memory (Seq2Seq LSTM); convolutional neural network (CNN).

The remainder of this paper is organized as follows: Section 2 summarizes the background and related work, Section 3 introduces two hardware emulators, Section 4 describes our testing platforms, Section 5 explains the deep learning model used in energy prediction, Section 6 discusses the experiments and results to validate the efficiency of our scheme, and Section 7 details the conclusions.

2. Background and Related Work

The thermostat became the pillar of energy conservation products shortly after its invention. A setpoint is utilized to control the temperature with its sensing results. There is already some research on reprogrammable thermostats to select the optimal setback schedule based on the historical data automatically [5]. Nevertheless, these studies only show how to generate a static control schedule. There are still some conflicts for energy-saving and human comfort, according to the dynamic occupancy patterns in real applications with a static schedule. Therefore, the real-time sensor data may be used for dynamic control of the HVAC system with the current occupancy numbers.

There are also some studies on how to calculate the energy consumption status by using pure simulations (without any hardware in the experiments). In [6], some energy-saving simulations are established, which uses a data-driven model to illustrate occupant behaviors. The work in [7] shows the wireless camera sensor network deployment to

gather the occupancy numbers. In [8], the researchers demonstrate a statistical model on the temporal occupancy of a building using a heterogeneous Markov chain model, which uses the occupancy number gathered from a sensor network. In [9], the concept of a smart thermostat is proposed to get the occupancy statistics.

Software emulators have been generated for both academic and commercial applications. In response to a run-time real case simulation and testing, hardware emulators and testing platforms are often established. FPGA boards are used with a run-time scheduling framework [10]. Besides, FPGA is also utilized as a hardware emulator for a DC motor controller in real-time [11]. Raspberry Pi might be the cheaper choice compared to others and can also be used for image processing and control systems. For instance, the Raspberry Pi board is also used for the face recognition [12] and vision navigation systems [13]. The Raspberry Pi based temperature monitoring system can be used to handle the MySQL data [14]. Therefore, Raspberry Pi could be the emulation and implementation platform. It contains a CPU core for data processing. In [15], an energy-efficient Fi-WSN is illustrated, which shows the scenario of a residential building. Compared to these simulators, our emulation platform has the entire system with data collection, analysis, modeling, and control for building energy cyber-physical control.

Our proposed emulator can use several computing steps, including denoising, data fusion, and prediction to save energy. Deep learning can be used to predict building energy.

In recent works, recurrent neural networks (RNN) have also been used as one of the most popular models to predict energy consumption. RNNs can be used to extract the patterns of time dependencies. Some LSTM-based RNN models have been introduced to forecast the short-term load. Similarly, a deep RNN model based on pooling is also proposed, as is a standard LSTM model with a generic algorithm for load forecasting in short- and mid-term estimation problems. RNNs are used to handle multiple input sequences to extract the most relevant features among several time slots. Daily load forecasting has also been modeled using dynamic time warping, together with GRUs. A cycle-based long-/short-term memory and a time-dependent convolutional neural network are used to increase the prediction performance.

The sequence-to-sequence (S2S) models, together with the encoder and decoder neural networks, can be used to deal with some classification problems so that it can improve the statistical machine translation performance.

3. Hardware Emulators

There are two hardware emulator platforms illustrated in this section. For the residential building, Raspberry Pi boards and sensors are used to control the room temperature while using the occupancy sensor information. For the commercial building, the BACnet protocol is used for the communication between sensors and the HVAC system. To control and save energy, sensor data are transmitted to the HVAC system wirelessly. Various sensors are used to get the occupancy data, such as temperature sensors, CO2 sensors, and passive infrared sensors (PIR).

3.1. Hardware Emulator 1: Use Raspberry Pi-Based Sensor/Thermostat Interactions for Residential Building

3.1.1. Objective

Some sensors can be used to collect the occupancy data for residential buildings. Motion sensors, such as PIR sensors, use pyroelectric signal pattern changes to count the number of people. On the other hand, image sensors use pattern recognition and image processing techniques to detect humans. Acoustic sensors can use sounds to analyze human activities. All sensor data can be used individually or collectively since they can complement each other.

To save the building energy, a physical or virtual (i.e., software-based) data server is needed to receive, store, and analyze the data. Raspberry Pi is used as the real data server that can perform sensor data fusion. Its board should be programmed to calculate

the occupancy information in real-time so that the temperature and airflow in residential houses can be controlled accordingly.

3.1.2. Difficulties

We still need to deal with some problems with the data server. The first problem is the data format since it is not standardized. In this case, it is not easy to fuse the data. The second problem is that for different types of sensors, there are no existing open-source codes for the stable control logic. The third problem is that it is hard to find any commercial thermostats that can easily change the internal function with sensor data since they are not programmable. Thus, a new 'thermostat' can be built with the Raspberry Pi board and sensors in Figure 2, which would be much cheaper and more convenient. In this case, it is easy to modify the control logic with our new emulated thermostats in a different environment.

3.1.3. Methods

The current temperature and heating/cooling setpoints can be shown on the LCD screen of the thermostat's digital display/control unit. Signals can be sent through its output port to control the heating/cooling/fanning functions. The communication links between occupancy sensors and thermostats can be established in this hardware emulation platform. Therefore, we can use data servers such as Raspberry Pi to collect the sensor information for the control logic. Afterwards, all this data can be transmitted to the HVAC system. Three lights, red, blue, and yellow, are used to represent FAN, Cooling Down, and Heat Up, respectively. A relay bus is employed to mimic the output port of the thermostat.

Figure 3 shows the connections and outputs of the Raspberry Pi, the relay bus, and three different lights. The occupancy sensing results and the room temperature can be used by the Raspberry Pi to send trigger signals. Afterwards, the Pi board can utilize the relay buses to control the status of the lights.

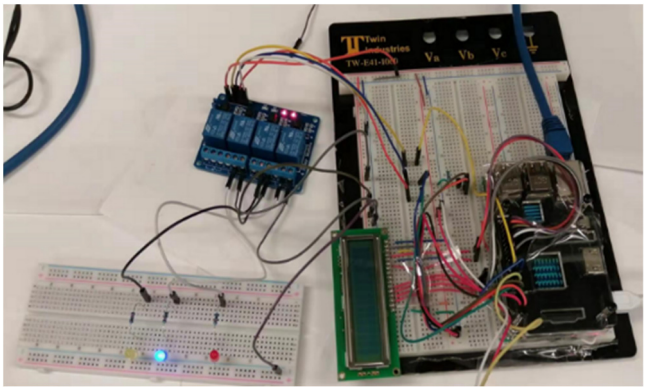

Figure 3. The system setup for mimicking the HVAC control.

In order to calculate the occupancy numbers in a room, Raspberry Pi and PIR sensors are used for this emulator. Whenever there is someone walking through the PIR sensor, which is very sensitive to the motions within ten meters of range, the console will show the current counter result added by 1. If someone walks out of the room, another sensor board will count the number. By using both boards, we can calculate the actual occupant number in the room. By deploying the boards in the proper places, such as near the door entry, we can avoid repeated same-person counting. If no one is in the room, then "0" will be shown in the console. This is a challenging topic still under research. The data in Raspberry Pi are saved in its local memory as a .csv file. Figure 4 is an example of how detection data is saved in the Raspberry Pi board.

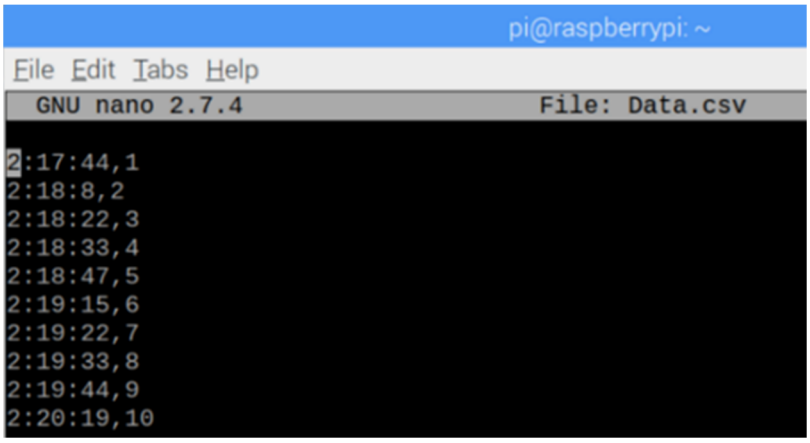

Figure 4. The data file saved locally in Raspberry Pi memory to record the time and amount of people that are detected.

Afterwards, the saved file can be fetched easily by other computers in the same network because the IP address of the current Raspberry Pi is known to them. In our emulator platform, such data sharing is achieved through the BACnet "WhoIs–Iam" handshake process. "WhoIs–Iam" is one of the most important software units for the communication and exchanging of data. In this function, the handshake application with the "bacpypes" package in Python should be initiated at the first step. The second step is to use functions such as request, confirmation, and indication. In the command window, "whoIs" is asked directly to try to gather parameters such as the server IP address. If this is answered with 'Iam', then two devices are connected for parameter sharing. A laptop is used as a server to fetch the data file from the Raspberry Pi every 5 s. In the meantime, SCP (secure copy protocol) is also utilized to establish the communication bridge. However, if we want to make the whole procedure compatible with the BACnet protocol, then the UDP file transfer between the server and client using Python can also be used.

Figure 5 displays the detecting time (x-axis) and the number of occupants (y-axis) and shows that the number of occupants changes with time variance.

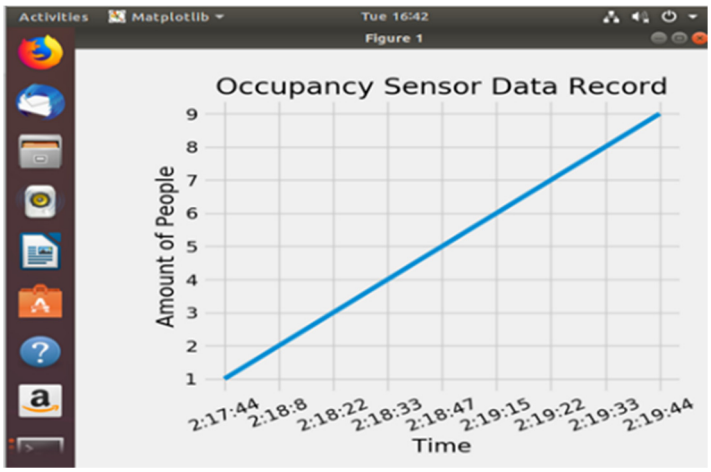

Figure 5. The results in the server (showing the total number of occupants).

3.2. Hardware Emulator 2: Sensor-to-HVAC Communication via BACnet for Commercial Building

3.2.1. Objective

A reliable communication channel between the HVAC control server and sensor data center should be built to send and receive the control signal for the energy-saving system. In the meantime, data transferred between each sensor by means of wired or wireless are also necessary. In order to collect the occupancy data and transmit it to the

HVAC control server, multi-hop data transmission is used instead of single-hop in the large residential building.

3.2.2. Difficulties

The BACnet communication protocol should be followed wired and wirelessly for standard building control. A hardware/software system has been established for building energy-saving control systems based on the BACnet protocol. Afterwards, the data fusion packets are sent back to the data server center. The whole communication and control system has been built from scratch by us since no open-source BACnet communication system can be found in the market now. Moreover, the HVAC control system should be able to send out the correct control signals with the processed sensor data. Additionally, multi-hop data transmissions can be easily added to the emulator platform with the supporting Zigbee-compatible protocol. The multi-hop communication provides a longer transmission delay and higher packet loss rates between data centers compared to the single hop transmission.

3.2.3. Methods

The BACnet protocol-based communication between the HVAC control sever and sensor data center can be shown on this platform. With the recognition of IP addresses and the use of the UDP protocol, the communication process can be built. The file transfer can be easily performed with the provided device's IP addresses and equipment. Moreover, the "WhoIs–Iam" BACnet handshake procedure can be used to obtain the client devices' IP addresses for the server's applications.

Note that all the devices/instruments do not need to be within the same subnet. Routers and gateways are used to deal with communication problems across different backbones or subnets. In order to save hardware resources, only two virtual machines are used. Meanwhile, only Raspberry Pi and a laptop are used for this platform. The virtual machines 1 and 2 are both within Ubuntu 18.04 OS and are installed in two VMware WorkStation 15 Player frames. Moreover, the mode: "Bridged: Connected directly to the physical network" is set for these two virtual machines. Figure 6 shows that the virtual machine can ping the Raspberry Pi device successfully.

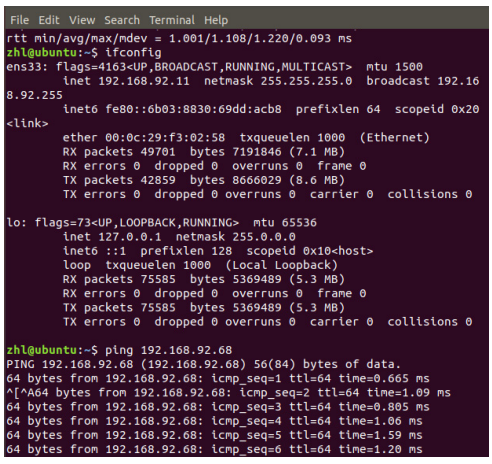

Figure 6. Ping the address of Raspberry Pi via virtual machine.

In order to keep track of the status of each device, occupancy numbers are saved as '.csv' files in both the virtual machine and physical hardware. The server (virtual machine 1) not only plots the data but also saves the data on its own hard disk. Such data will be distributed to other clients for further processing/control. Moreover, an HVAC control machine (virtual machine 2) is also established. The SCP is used to obtain the data from the server (virtual machine 1); afterwards, it uses the data for HVAC control.

The panel's parameters are updated periodically. *PySimpleGui* is used to develop the HVAC control panel.

Figure 7 shows the whole setup of the second emulator platform.

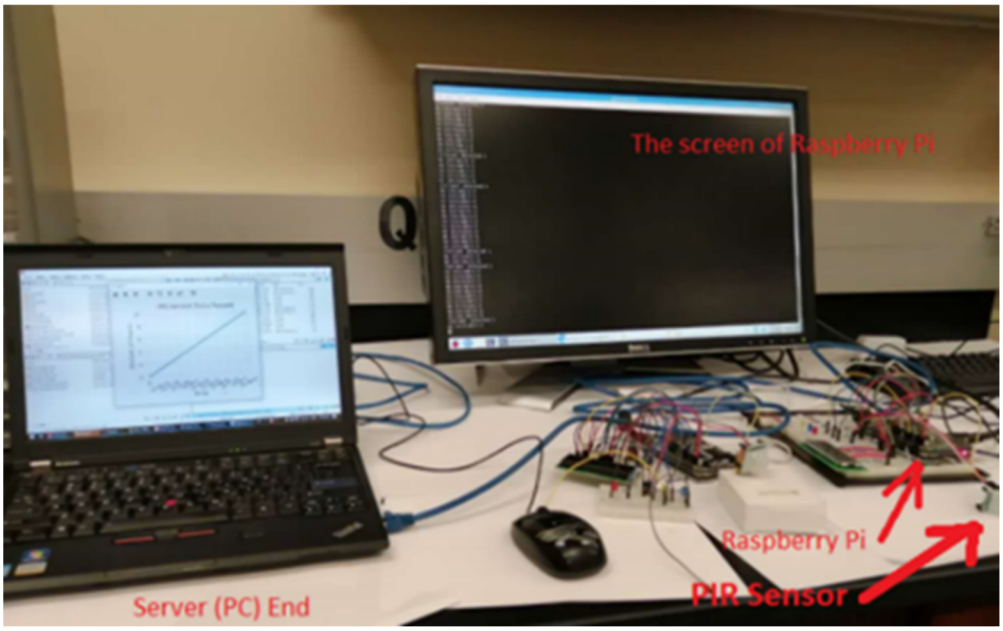

Figure 7. Testbed setting for connecting HVAC virtual machine with server to get sensor data.

4. Testing Platforms and Deep Learning

YABE (Yet Another BACnet Explorer) and WebCTRL are used to display as testing platforms for the emulator system. They are very compatible to serve as the communication 'bridge' between the control server based on the BACnet protocol and the HVAC system. The setpoint (present value field in the YABE property part) can be overwritten by the control server (written in BACpypes code) 'Reading Property' function. Instead of simply using the command window/built-up files to change the property value (setpoint), we write codes to define the whole BACnet device object and properties and then change the value we want.

Moreover, deep learning is also used to predict occupancy numbers. It also introduces the concepts of LSTM models, attention mechanisms and encode-to-decoder structures.

4.1. YABE Platform

YABE (Yet Another BACnet Explorer) in Figure 8 is a graphical window-based program for exploring and navigating BACnet devices. It is also an open-source platform to browse BACnet devices on either Windows or Linux, and it now supports both BACnet IPv4 and IPv6 + BACnet MSTP + BACnet PTP + BACnet Ethernet. Moreover, there are also some basic functions for read, write, read multiple, write multiple, iam, whois, subscribeCOV, notify, WriteFile, ReadFile, etc.

There are several steps to using the YABE program. The first step is to start the BACnet devices, as well as the YABE platform. The second step is that we need to add devices under the 'BACnet/IP over UDP' field. In this case, the communication between YABE and the BACnet devices could be evaluated. Therefore, a UDP connection to the 'Devices' tree will be added to the system. This also triggers three "WHOIS" broadcasts. Therefore, any BACnet/IP devices that can be detected in the network can be displayed in the tree. For instance, it will show up as something similar to "192.168.92.5:57049-389002". It has the IP address, the UDP port, and the device ID of the current device with the YABE program. Moreover, a local endpoint IP address can also be included with more than one Ethernet card. We can either select one from the list or write one on our own without an existing

gateway. Afterwards, the BACnet device will show up in the 'devices tree'. Then, all the nodes of the device will be fetched and displayed in the 'Address Space' in YABE. It is also very important for us to choose the node in the 'address space' tree so as to figure out the given register's properties. In the meantime, the program will also fetch and display all properties in the 'Properties' field.

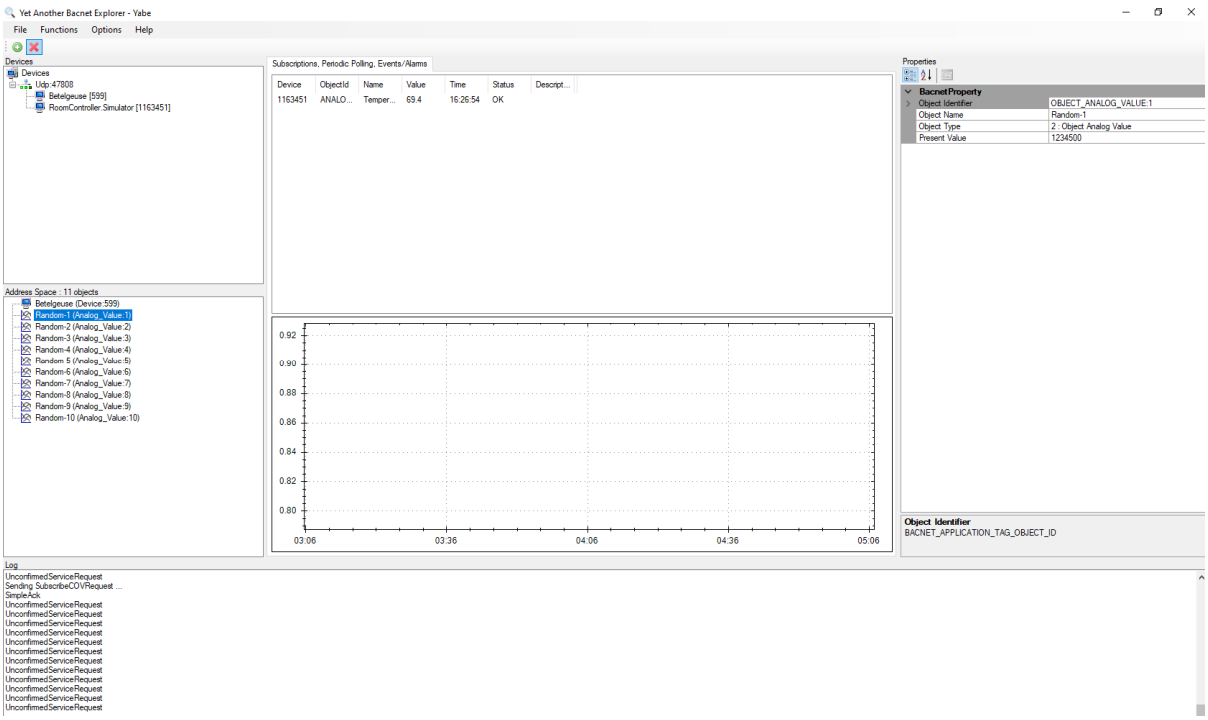

Figure 8. YABE platform with virtual BACnet device and properties value.

However, there is also a problem with multiple UDP clients on the same IP, that is, they only connect to one BACnet UDP port and seek to transmit all their traffic through this port. In order to solve this problem, more than one client should be used at each machine. For example, we can use a virtual machine with a client with various IP addresses. We can also connect two sockets in the program to handle this issue. The BACnet socket at 0xBAC0 should be used for receiving broadcasts and cannot be used to transmit any data or receive any unicast traffic. If we want to transmit a broadcast, another socket (a random port) should be utilized. Additionally, it is also necessary to send the broadcast traffic to the 0XBAC0 port. The random port will be flagged as the source for this broadcast. Therefore, they will transmit back their unicast message to this random port instead of the 0XBAC0 port while the other clients collect the broadcasted message.

4.2. Communication Step

4.2.1. Detect the BACnet Protocol Device

BACnet is a connection-less protocol based on UDP. As long as the address for the communications with another device is known, we can send it a request and wait for the response back. This is different from HTTP or other protocols, in which the request and response are inside a connection. It provides a context for the exchange, as shown in Figure 9.

Every BACnet device has a unique identifier, created by a business process outside of the protocol. Those identifiers are given to devices within a BACnet internet.

In this case, a unicast message 'Who Is' without providing a device range can be sent with the device address. Then, the device could send back an 'I Am' message with its device identifier, as well as some other BACnet communication data. Moreover, it also completes the "binding" of device IDs and IP addresses.

```
def main():
    global this_device
    global this_application

    # parse the command line arguments
    args = ConfigArgumentParser(description=__doc__).parse_args()

    if _debug: _log.debug("initialization")
    if _debug: _log.debug("    - args: %r", args)

    # make a device object
    this_device = LocalDeviceObject(
        objectName=args.ini.objectname,
        objectIdentifier=int(args.ini.objectidentifier),
        maxApduLengthAccepted=int(args.ini.maxapdulengthaccepted),
        segmentationSupported=args.ini.segmentationsupported,
        vendorIdentifier=int(args.ini.vendoridentifier),
        )

    # make a simple application
    this_application = WhoIsIAmApplication(this_device, args.ini.address)

    # make a console
    this_console = WhoIsIAmConsoleCmd()
    if _debug: _log.debug("    - this_console: %r", this_console)

    # enable sleeping will help with threads
    enable_sleeping()
```

Figure 9. Detect the device by UDP communication protocol.

4.2.2. Communication—Send/Receive Request

YABE is necessary to send out *Who-is* and *I-am* messages and display the results that it receives. In this case, we could establish the communications between devices using the BACnet protocol. Additionally, the BACnet protocol's unique and specific identifiers in its configuration are utilized in a large number of applications, so as to figure out whom they communicate with. As long as these identifiers are given to a device, they typically do not change, even as the network topology changes. In this case, the communication between devices can be established, including sending message requests and receiving confirmations. Specifically, BACnet devices use the 'Who-is' requests to translate device identifiers into network addresses, which is similar to a decentralized DNS service. Nevertheless, DNS service names are unsigned integers. The request is broadcasted to the whole network, and the client waits around to listen for 'I-am' messages.

The request from "who-is" is marked as "unconstrained", which means that every device that receives this message will respond with their corresponding "I AM" messages. After broadcasting the requests, the BACnet traffic can be shown on the device's screen with a broadcasted 'I AM' message, at least. Then, the source address of the 'I AM' response is tied to the device identifier and most communications are unicast messages thereafter. The following codes shown in Figure 10 respond to the request broadcasted globally and transmit the IP address and other information back to the broadcasting device via BACnet.

4.2.3. Create Virtual BACnet Device

BACnet devices utilize objects to provide information. A certain number of properties could be read in each device. Therefore, the property identifier, as well as the object identifier, are required to be passed to a 'ReadPropertyRequest' in order to read a single property. Then, this request should be sent to the remote device. In this case, there should be at least one device object provided by the BACnet device. At the same time, by utilizing the 'objectIdentifier()' method of the remote device, it is necessary that a remote device's object be retrieved. The device object contains a list of all other objects provided by the device, which can be retrieved using the code shown in Figure 11.

```
def do_whois(self, args):
    """whois [ <addr> ] [ <lolimit> <hilimit> ]"""
    args = args.split()
    if _debug: WhoIsIAmConsoleCmd._debug("do_whois %r", args)

    try:
        # gather the parameters
        request = WhoIsRequest()
        if (len(args) == 1) or (len(args) == 3):
            addr = Address(args[0])
            del args[0]
        else:
            addr = GlobalBroadcast()

        if len(args) == 2:
            lolimit = int(args[0])
            hilimit = int(args[1])
        else:
            lolimit = hilimit = None

        # code lives in the device service
        this_application.who_is(lolimit, hilimit, addr)

    except Exception as error:
        WhoIsIAmConsoleCmd._exception("exception: %r", error)

def do_iam(self, args):
    """iam"""
    args = args.split()
    if _debug: WhoIsIAmConsoleCmd._debug("do_iam %r", args)

    # code lives in the device service
    this_application.i_am()
```

Figure 10. Send and reponed the request between two devices.

```
def main():

    # parse the command line arguments

    args = ConfigArgumentParser(description=__doc__).parse_args()


    if _debug: _log.debug("initialization")

    if _debug: _log.debug("    - args: %r", args)


    # make a device object

    this_device = LocalDeviceObject(

        objectName=args.ini.objectname,

        objectIdentifier=('device', int(args.ini.objectidentifier)),

        maxApduLengthAccepted=int(args.ini.maxapdulengthaccepted),

        segmentationSupported=args.ini.segmentationsupported,

        vendorIdentifier=int(args.ini.vendoridentifier),

        )


    # make a sample application

    this_application = BIPSimpleApplication(this_device, args.ini.address)
```

Figure 11. Create BACnet device object using Python.

4.2.4. Read/Write Properties of Setpoint

As shown in Figure 4, after creating the object, the 'Read Property' requests are utilized to send the value of Analog Input Object 1. It can also write the analogInput:1 in the sample applications, which is stored as ('analogInput', 1) in the Python code. Instead of a client application, the server is used as the sample, which can reply to all the wire requests. For instance, it can reply to the Who Is and Read Property requests. All these requests are not generated by it, instead, they are initiated by the client. In each object, some values, such as identifier, name, status flags, and present value can be created with analog signals. For this project, we will read the present value, which is the same as the setpoint value in the analysis.

4.2.5. Create Virtual BACnet Device

The communication bridge between the WebCTR system and the sensor data server will be established in the following section. As can be seen in Figure 12, WebCTRL is a BACnet-native, browser-based building automation system through which users can fully access their buildings' schedules, setpoints, trends, alarms, and other control functions from virtually any computer. In this case, personal phones and some other wireless devices can be used as the WebCTRL control system of the buildings.

```
with open('file_occ.csv', 'r') as csv_file:
    new_value = csv_file.read()
    value1 = int(new_value)
csv_file.close()
# make an analog value object
test_av = AnalogValueObject(
    objectIdentifier=("analogValue", 0),
    objectName="1-1-office-setpoint-heating",
    presentValue=int(value1),
    statusFlags=[0, 0, 0, 0],
    covIncrement=1.0,
)
_log.debug("    - test_av: %r", test_av)

# add it to the device
this_application.add_object(test_av)
_log.debug("    - object list: %r", this_device.objectList)
```

Figure 12. Read BACnet device's properties using Python.

In order to provide support to multiple languages at the same time, WebCTRL (Figure 13) has a useful and powerful spreadsheet-based reporting tool. As a native BACnet system, WebCTRL interfaces with LonWorks, Modbus, and many other protocols to provide an integrated solution to building control needs. WebCTRL can be integrated into a BACnet Advanced Workstation Software, which is certified by the BACnet Testing Laboratory. In this case, it could browse the data from any connected BACnet device, including temperature or humidity sensors, etc.

Figure 13. WebCTRL platform.

The WebCTRL server could respond to whatever request is seen on the wire. This control server is able to respond to 'Who Is' requests, as well as the 'Read Property' requests. Specifically, the request is generated by a client instead of by itself.

The server is searching for some BACnet-compatible server devices. The data transmission on WebCTRL can be established based on the 'Who Is–I am' communication through UDP. In this case, the IP address of the data server should be set to a certain range since the WebCTRL has a specific IP number. For instance, in Figure 14, the IP address is 192.168.168.100, and the data server should be set accordingly, to 192.168.168.5.

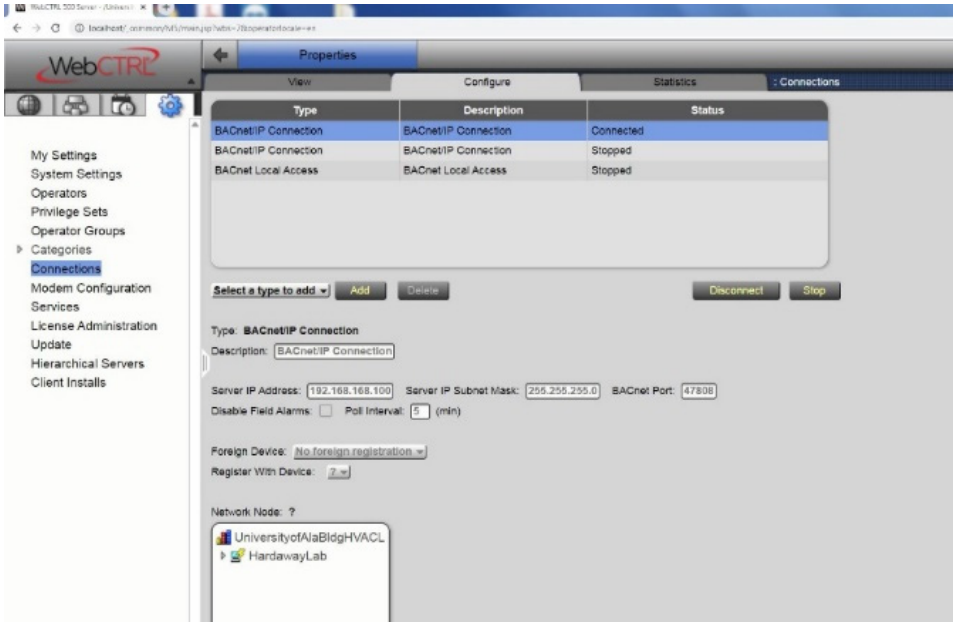

Figure 14. Communication configurations in WebCTRL through UDP.

Figure 15 shows that UDP establishes the communication between the data server and WebCTRL. In this example, the reading data displayed in the YABE browser based on BACnet protocol witnessed successful communication.

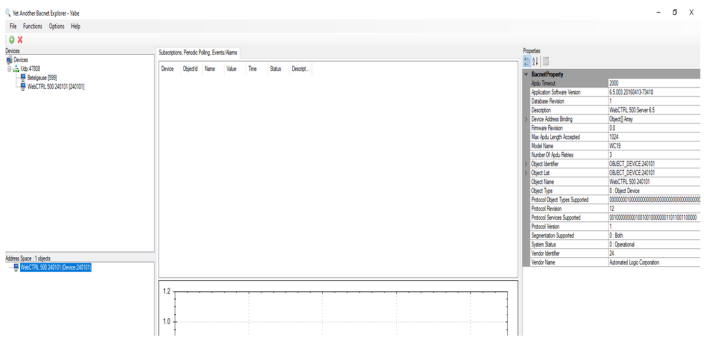

Figure 15. Communication between WebCTRL and our data server.

WebCTRL is a browser that could only read properties from some connected BACnet devices. Some functions, including 'set-point' in BACnet devices, should be provided for WebCTRL to read from. BACnet can be used to let devices exchange data and metadata. Therefore, we could override the previous 'set-point' data with our own code.

5. Platform Extensions with Deep Learning

The Transmission Control Protocol/Internet Protocol (TCP/IP) suite was generated by the Department of Defense (DoD); in this case, we can make sure and preserve the data integrity. It can maintain communications in the event of a catastrophic war. TCP/IP is one of the reliable protocols and can reside at the transport layer of the OSI reference model compared to other transmission protocols. The lost data re-transmission assures the data delivery. Therefore, TCP/IP packages can be collected without any specific order.

The IP layer could route the segments as packets through a subnetwork. Specifically, this step is started right after the transport layers receive data streams from its upper layer. In this case, all these segment packets are transmitted to the Host-to-Host layer protocol on the receiving host side. Additionally, the data stream can be recreated by the host layer protocol. The data will be handed to the upper-layer protocols and applications. TCP/IP generates a reliable session by establishing a virtual TCP connection and includes acknowledgements, sequence numbers, and a flow control window. The three-way handshakes are commonly used to set up TCP communication. The connection is uniquely identified by a combination of sources, as well as its destination IP port number or address.

Energy consumption is one of the biggest contributors to global warming, and most energy use is due to human activities; for example, electrical heating for commercial and residential buildings could consume almost half of the total energy production. Therefore, it is critical to manage energy use efficiently to reduce negative environmental hazards, including carbon dioxide emissions. Energy systems often do not have dramatic changes in load supply, which makes it possible to perform a prediction of future loads with small errors.

Due to the importance of energy scheduling and allocation, load prediction has attracted the attention of researchers. In this context, wireless sensors are widely used to collect energy-related data (such as the number of people in the building) due to their low cost and low power consumption. However, the popular uses of sensors also cause some issues, such as higher wireless transmission delay, poor RF transmission quality, and higher energy consumption. Meanwhile, due to the reduction of the cost of a single sensor, some problems arise, including hardware failure, data transmission failure, missing data, etc. To recover or correct abnormal data, some features should be extracted when the data is being preprocessed. With data pre-processing, the data collection process can be optimized and the correlation between features may also be reduced. Therefore, the energy prediction models should overcome the above data collection issues.

Building energy prediction models includes three main categories: short-term forecasting, mid-term forecasting, and long-term forecasting. The prediction algorithms are typically from the fields of statistics, physics, or machine learning. In this paper, our algorithm is based on machine learning and deep learning models. Some solutions use support vector machines (SVM) and neural networks to estimate energy consumption [16,17]. Due to the large size of the dataset, neural networks have been used to increase the prediction accuracy [18–20]. A one-day-ahead energy prediction scheme is proposed with an ensemble parameter selection model [21]. In [22,23], a Bayesian regularization algorithm is introduced to optimize the neural network prediction scheme. An anomaly detection model based on ensemble neural networks is proposed in [24] to use a random forest to achieve the prediction-based classifiers.

In recent works, recurrent neural networks (RNN) have been used as one of the most popular models to predict energy consumption. RNNs can be used to extract the patterns of time dependencies. Some LSTM-based RNN models are introduced in [25] to forecast the short-term load. Similarly, a deep RNN model based on pooling is also proposed in [26], and a standard LSTM model with a generic algorithm is proposed in [27] for load forecasting in short- and mid-term estimation problems. RNNs is used in [28] to handle multiple input sequences so as to extract the most relevant features among several time slots. Daily load forecasting has also been modeled in [29] using dynamic time warping, together with GRUs. In [30] a cycle-based long-/short-term memory and a time-dependency convolutional neural network are used to increase the prediction performance.

RNNs and LSTMs have been shown to outperform most other predicting models [25,26,29]. The sequence-to-sequence recurrent neural networks are originally used in language translation problems [31]. The standard LSTM and LSTM-based sequence-to-sequence models in [32] are applied to residential building energy prediction. A hybrid algorithm in [33] is proposed to predict energy consumption using LSTM neural networks, empirical mode decomposition, and similar days selection. Two sequence-

to-sequence LSTM-based models [34] are used to predict medium- and long-term energy consumption.

In our paper, an S2S (encoder to decoder) neural network based on the deep LSTM attention model is proposed to predict building energy consumption. The LSTM is used to model time dependencies. The S2S (encoder to decoder) structure strengthens the ability to model time series data. The encoder is utilized to extract the relevant information and perform the prediction task, while the decoder can restore the extracted features. Additionally, the attention mechanism is used to get the relationship between the output and its corresponding input data. It can also build the connection between the encoder and decoder sequence model. Overall, the S2S model based on the deep LSTM attention mechanism outperforms all other models.

5.1. Analyze the Data

Some machine learning and deep learning models can be used as engines to train the data in HVAC systems. Various data can be collected by using different IoT sensors, which are installed in buildings. Afterwards, some deep learning models can be used to analyze these attributes, for example, energy consumptions can also be evaluated and predicted. Therefore, commands can be produced by the deep learning prediction results. Moreover, the temperature of the HVAC units can be controlled with these commands.

There are five different time window lengths used in the proposed sequence model, including 10, 20, 30, 40, and 50 intervals. A five-minute time interval is used to read the meter results. Therefore, the corresponding time is 50 min (~1 h), 100 min (~2 h), 150 min (2.5 h), 200 min (~3 h), and 250 min (~4 h), separately. Meanwhile, all the selected models are trained for nine different epochs, ranging from 1 to 9. These epochs are acceptable levels in terms of algorithm convergence since the model can be well trained within them. The dataset is shown in Figure 5.

5.2. Model Structure

The deep learning model proposed in this paper optimizes the basic S2S (encoder to decoder) model structure. The idea is shown in Figure 16. The encoder could extract different features to train the model in the pretraining steps of the deep neural network (DNN). Afterwards, the model is fine-tuned via supervised learning. Nevertheless, it is not effective to keep many parameters for subsequent supervised learning since it will take a long time to train the model. This could make it unsuitable for online processes. If we see DNN as an optimization problem, it is difficult to achieve the global optimization since the model may easily get trapped into the local optimal.

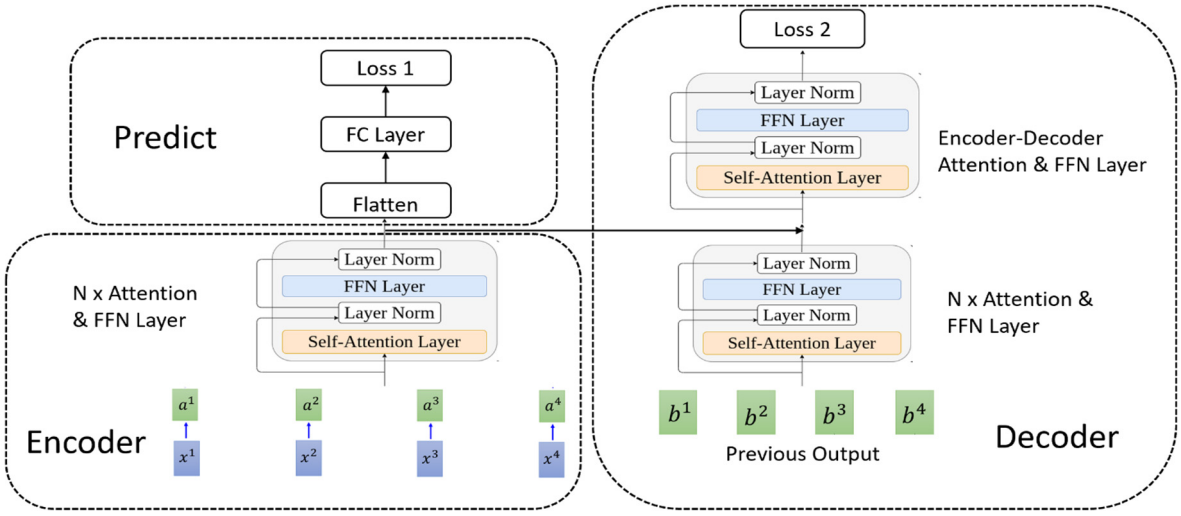

Figure 16. Encoder-Decoder Model Structure.

This generates a reliable session by establishing a virtual TCP connection that includes acknowledgements, sequence numbers, and a flow control window. The three-way handshakes are commonly used to set up TCP communication. The connection is uniquely identified by a combination of sources, as well as its destination IP port number or address.

To deal with the problems illustrated above, it is important to change the training pattern of the S2S model structure. Therefore, the attention mechanism can be used as a constraint added in the middle of the S2S neural network, as shown in Figure 16. In this model, two processes are used to optimize the training loss. Meanwhile, the final loss in the last step is calculated through the sum of these two losses. As can be seen from the encoder, the parameters of the encoder sequence are the same since they all share the same feature extraction layer. The encoder part can learn the most efficient expression of data features. Therefore, the model can be trained with high accuracy.

Additionally, to capture the long-distance relevant information and get better prediction performance, the traditional LSTM models in the DNN are replaced by the attention mechanism. In the proposed model structure, three modules are used for the model construction, i.e., *encoder module*, *decoder module*, and *prediction module*.

Encoder module: the attention mechanism is used to extract the relevant input information. The output of the attention layers in the encoder can be utilized as the input to the decoder sequence to restore previous input data. Moreover, it can also be used to predict the energy consumption in the prediction module. Adding this constraint in the module can prevent the encoder module from copying the input embeddings directly to the output. Instead, the encoder is forced to learn more characteristics of the input embeddings so as to improve the load forecasting performance at the next timestamp. Besides, the key and value matrix generated by the attention mechanism in the encoder module can also be used in the transition layer between the encoder and decoder modules.

Decoder module: the extracted features from the encoder module are restored. In this case, by comparing the differences between the original input data with the restored embeddings, the validity can be calculated and verified.

Predict module: this predicts energy consumption by using the embeddings generated from the encoder module. Instead of directly using the information, it must first be flattened and processed by the fully connected layer. Therefore, the feature dimensions can be modified for the specific prediction task.

Moreover, a multi-step training approach is used in our model for the multi-tasking problems. In our proposed model, the multi-tasking training method is used first, before being split into several single task trainings. In the single training process, the extracted relevant data is used for load forecasting. Specifically, in our proposed method, all relevant feature embeddings can be extracted by using the multi-tasking model. When the model prediction is close to the convergence point, it is necessary to learn some unique information to improve the prediction performance. Therefore, after the multi-tasking process, the encoder can be used to continue the prediction task by using the step-by-step training approach. The prediction performance can eventually be improved with this training method.

6. Results

The proposed approach was implemented using the Tensorflow platform in Python code. Two machines with GPUs are used to train the models, including two NVIDIA GeForce RTX 2080 TI GPU cards and one NVIDIA GeForce GTX 1060 GPU card.

In the following discussions, we will compare and analyze the performance of different models. To prove that the proposed model is effective enough and runs faster than the traditional neural networks, we have conducted the following comparative experiments.

A simple traditional math model (Random Forest) is used as the baseline for comparison. Moreover, CNN, bi-directional LSTM, and attention mechanisms are utilized to extract the relevant features and conduct the training process. The bi-directional LSTM and the attention layers are also utilized as the training layers of the encoder and decoder

to extract the pertinent information and predict energy consumption. While the model has reached convergence, the encoder has been well-trained. Therefore, it can be used to continue the training of the prediction process.

Grid search is often used as the training method for parameter tuning. During the model comparison, the structure of the components is the same, including CNN and bi-directional LSTM.

To maintain the fairness of the results, the same data are provided to each model. As can be seen in Figure 17, MAPE is used as the metric to compare the performance of each model. Compared with the general LSTM model, bi-directional LSTM, deep LSTM, and deep bi-LSTM models with an attention mechanism have a better performance after certain epochs (in this case, approximately five epochs). One of the possible reasons is that LSTM did not extract enough relevant information in a very long time-series data. Specifically, the bi-directional LSTM model with an attention mechanism obtains the best results. According to the different epochs used for each model, since the model with the attention mechanism works in parallel, it can make the model converge faster than others. The single LSTM model can capture sequence characteristics; however, it is trained in serial and thus has a slower training speed. Therefore, the attention mechanism not only speeds up the training process but also improves the prediction performance.

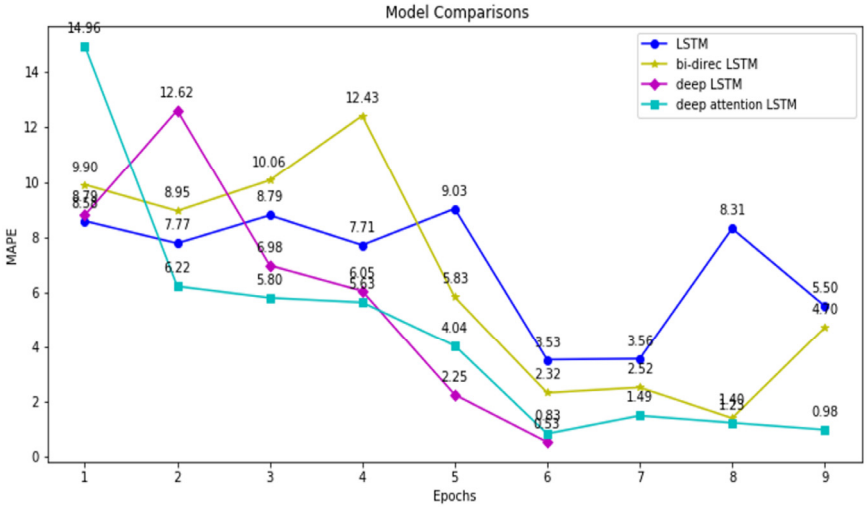

Figure 17. MAPE for different models by various epochs.

In terms of the layers for different models in Figure 18, after eight layers, the bi-directional model has a higher performance than the one-directional model. The reason is that the bi-directional LSTM is a combination of a forward and backward model that share the same gate LSTM unit. In this case, both the future and previous information are considered to predict energy consumption. Then, it can gain higher precision results.

In Figure 19, the prediction performance of different models in various time windows is illustrated. The MAPE of different models can be observed here. Even if the time window changes, the deep bi-directional LSTM model with the attention mechanism has the highest performance. The traditional model is the baseline here since it uses rule-based calculations in a building system. However, since the model structure is relatively simple and cannot effectively extract the relevant feature information, it is not suitable for finding the optimal building energy consumption result. With a higher performance in different time windows, a bi-directional LSTM with an attention mechanism can capture time-series sequence features. Moreover, it can make use of the trained encoder to get efficient feature information while learning some unique features that are beneficial to the prediction process. Therefore, the deep bi-directional LSTM model with an attention mechanism is the best option among all the models.

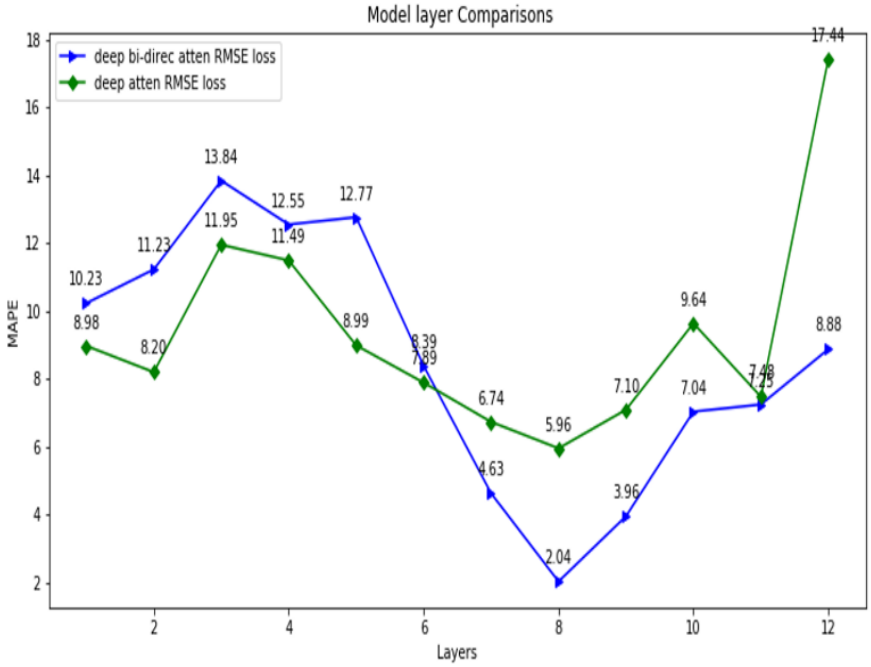

Figure 18. Comparisons between bi-directional and one-direction models with attention mechanism.

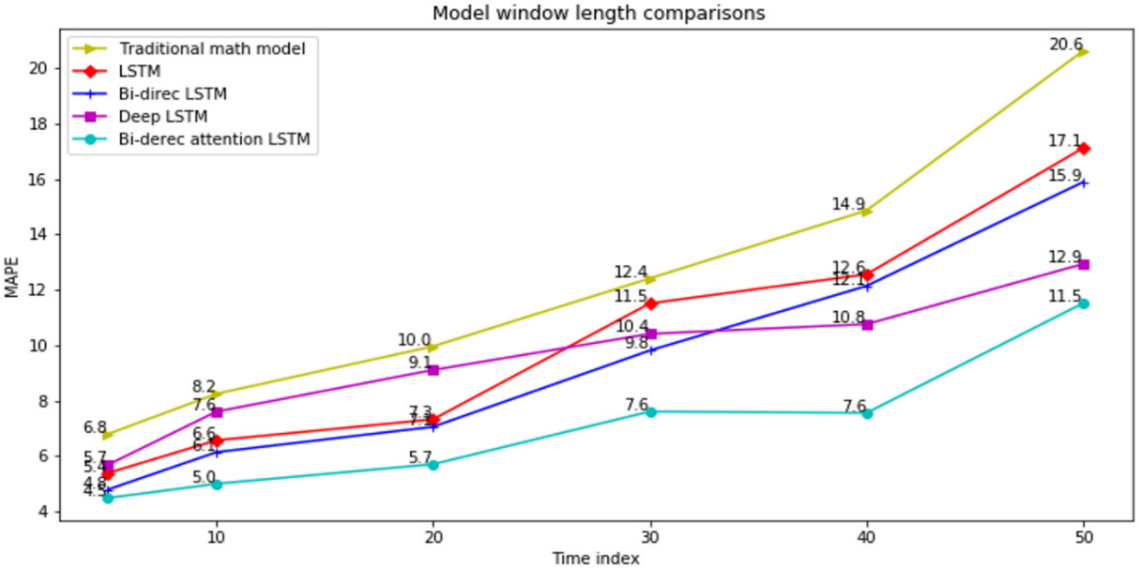

Figure 19. Model comparisons with different time window lengths.

7. Conclusions

In this paper, two hardware emulators that can sense the occupancy data in *residential* or *commercial* buildings have been illustrated. In this case, it is possible to save energy through improved control of the HVAC system. With precise energy consumption predictions, we can further perform optimal production planning and scheduling. Compared with traditional models, the deep neural network has witnessed better predicting performance due to its ability to extract relevant information and model time dependencies.

For the residential building, the system collects various sensor information. It can also generate data fusion in the data server. Afterwards, real-time processing and controlling of the system will be controlled by the Raspberry Pi board. In this case, the airflow is controllable. For the commercial building, the sensor data can be sent among many rooms by the multi-hop communication protocol. Subsequently, the occupancy data can be analyzed with some deep learning models in the data server and the information can be

sent to the HVAC control server. In this case, YABE and WebCTRL can be utilized to test the communication performance.

In this paper, an S2S (encoder to decoder) neural network based on the deep LSTM attention model is proposed to predict building energy consumption. The LSTM is used to model the time dependencies. The S2S (encoder to decoder) structure strengthens the ability to model the time series data. The encoder is utilized to extract the relevant information and perform the prediction task, while the decoder can restore the extracted features. Additionally, the attention mechanism is used to get the relationship between the output and its corresponding input data. It can also build the connection between the encoder and decoder sequence models. Overall, the S2S model based on the deep LSTM attention mechanism outperforms all other models.

For the next-step research, we will draw some new conclusions with more datasets containing various features. It is also necessary to consider and explore some industrial applications. Moreover, we will investigate the optimization of the S2S model structure, such as adding the models to construct a hybrid model for better prediction.

Author Contributions: Funding acquisition, Z.O.; Project administration, F.H.; Writing—original draft, Z.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by United States Department of Energy grant number DE-AR0000936.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work is supported by the U.S. Department of Energy, Advanced Research Projects Agency–Energy directed by Marina Sofos under the SENSOR program through Grant DE-AR0000936—"Quantification of HVAC Energy Savings for Occupancy Sensing in Buildings through an Innovative Testing Methodology".

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

References

1. Energy Forecasting Division, Energy Sector, Natural Resources Canada. *Canada's Energy Outlook, 1996–2020*; Minister of Supply and Services Canada: Ottawa, ON, Canada, 1997.
2. Rathouse, K.; Young, B. Domestic heating: Use of controls. *Defra Mark. Transform. Programme* **2004**, *5*, 24.
3. Center, I.E. Lower energy bills with a set-back thermostat. *Iowa Energy Cent. Des. Moines* **2010**. Available online: http://www.energy.iastate.edu/news/pr/energysavingideas/setbacktherm.htm (accessed on 29 November 2021).
4. Ye, Z.; O'Neill, Z.; Zhang, L.; Hu, F.; Chu, Z. Hardware-Based Emulator for Building Energy Cyber-Physical Control with Occupancy Sensing. In Proceedings of the 17th International Conference on Information Technology—New Generations (ITNG 2020), Las Vegas, NV, USA, 5–8 April 2020; Volume 1134, p. 493.
5. Gao, G.; Whitehouse, K. The self-programming thermostat: Optimizing setback schedules based on home occupancy patterns. In Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, Memphis, Tennessee, 3–6 November 2009; pp. 67–72.
6. Dong, B.; Andrews, B. Sensor-based occupancy behavioral pattern recognition for energy and comfort management in intelligent buildings. In Proceedings of the Building Simulation, Glasgow, Scotland, 27–30 July 2009; pp. 1444–1451.
7. Erickson, V.L.; Lin, Y.; Kamthe, A.; Brahme, R.; Surana, A.; Cerpa, A.E.; Sohn, M.D.; Narayanan, S. Energy efficient building environment control strategies using real-time occupancy measurements. In Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, Berkeley, CA, USA, 3 November 2009; pp. 19–24.
8. Erickson, V.L.; Carreira-Perpiñán, M.Á.; Cerpa, A.E. OBSERVE: Occupancy-based system for efficient reduction of HVAC energy. In Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks, Chicago, IL, USA, 12–14 April 2011; pp. 258–269.
9. Lu, J.; Sookoor, T.; Srinivasan, V.; Gao, G.; Holben, B.; Stankovic, J.; Field, E.; Whitehouse, K. The smart thermostat: Using occupancy sensors to save energy in homes. In Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, Zürich, Switzerland, 3–5 November 2010; pp. 211–224.
10. Beckert, R.; Fuchs, T.; Hardt, W. A Run-Time Scheduling Framework for a Reconfigurable Hardware Emulator. In Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007), Lubeck, Germany, 29–31 August 2007; pp. 147–150.

11. Rosa, V.S.; Gervini, V.I.; Gomes, S.C.; Bampi, S. A hardware DC motor emulator. In Proceedings of the 2010 First IEEE Latin American Symposium on Circuits and Systems (LASCAS), Foz do Iguacu, Brzail, 24–26 February 2010; pp. 45–48.
12. Hasban, A.; Hasif, N.; Khan, Z.; Husin, M.; Rashid, N.; Sharif, K.M.; Zakaria, N. Face recognition for Student Attendance using Raspberry Pi. In Proceedings of the 2019 IEEE Asia-Pacific Conference on Applied Electromagnetics (APACE), Melacca, Malaysia, 25–27 November 2019; pp. 1–5.
13. Mao, J.; Guo, Z.; Geng, H.; Zhang, B.; Cao, Z.; Niu, W. Design of Visual Navigation System of Farmland Tracked Robot Based on Raspberry Pie. In Proceedings of the 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), Xi'an, China, 19–21 June 2019; pp. 573–577.
14. Fatangare, M.; Nimbalkar, A.; Chite, G.; Narkhede, A.; Khilnani, A. An Efficient Temperature Monitoring using Raspberry Pi. In Proceedings of the 2020 International Conference on Inventive Computation Technologies (ICICT), Sydney, Australia, 21–24 April 2020; pp. 1–5.
15. Van, D.P.; Rimal, B.P.; Maier, M.; Valcarenghi, L. Design, analysis, and hardware emulation of a novel energy conservation scheme for sensor enhanced FiWi networks (ECO-SFiWi). *IEEE J. Sel. Areas Commun.* **2016**, *34*, 1645–1662.
16. Grolinger, K.; L'Heureux, A.; Capretz, M.A.; Seewald, L. Energy forecasting for event venues: Big data and prediction accuracy. *Energy Build.* **2016**, *112*, 222–233. [CrossRef]
17. Seyedzadeh, S.; Rahimian, F.P.; Glesk, I.; Roper, M. Machine learning for estimation of building energy consumption and performance: A review. *Vis. Eng.* **2018**, *6*, 1–20. [CrossRef]
18. Hippert, H.S.; Pedreira, C.E.; Souza, R.C. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Syst.* **2001**, *16*, 144–155. [CrossRef]
19. Darbellay, G.A.; Slama, M. Forecasting the short-term demand for electricity: Do neural networks stand a better chance? *Int. J. Forecast.* **2000**, *16*, 71–83. [CrossRef]
20. Li, Y.-C.; Fang, T.-J.; Yu, E.-K. Study of support vector machines for short-term load forecasting. *Proc. CSEE* **2003**, *23*, 55–59.
21. Jetcheva, J.G.; Majidpour, M.; Chen, W.-P. Neural network model ensembles for building-level electricity load forecasts. *Energy Build.* **2014**, *84*, 214–223. [CrossRef]
22. Chae, Y.T.; Horesh, R.; Hwang, Y.; Lee, Y.M. Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings. *Energy Build.* **2016**, *111*, 184–194. [CrossRef]
23. Yuan, C.; Niu, D.; Li, C.; Sun, L.; Xu, L. Electricity consumption prediction model based on Bayesian regularized bp neural network. In Proceedings of the International Conference on Cyber Security Intelligence and Analytics, Shenyang, China, 21–22 February 2019; pp. 528–535.
24. Araya, D.B.; Grolinger, K.; ElYamany, H.F.; Capretz, M.A.; Bitsuamlak, G. An ensemble learning framework for anomaly detection in building energy consumption. *Energy Build.* **2017**, *144*, 191–206. [CrossRef]
25. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Xu, Y.; Zhang, Y. Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Trans. Smart Grid* **2017**, *10*, 841–851. [CrossRef]
26. Shi, H.; Xu, M.; Li, R. Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Trans. Smart Grid* **2017**, *9*, 5271–5280. [CrossRef]
27. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Optimal deep learning lstm model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches. *Energies* **2018**, *11*, 1636. [CrossRef]
28. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Single and multi-sequence deep learning models for short and medium term electric load forecasting. *Energies* **2019**, *12*, 149. [CrossRef]
29. Yu, Z.; Niu, Z.; Tang, W.; Wu, Q. Deep learning for daily peak load forecasting–a novel gated recurrent neural network combining dynamic time warping. *IEEE Access* **2019**, *7*, 17184–17194. [CrossRef]
30. Han, L.; Peng, Y.; Li, Y.; Yong, B.; Zhou, Q.; Shu, L. Enhanced deep networks for short-term and medium-term load forecasting. *IEEE Access* **2018**, *7*, 4045–4055. [CrossRef]
31. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
32. Marino, D.L.; Amarasinghe, K.; Manic, M. Building energy load forecasting using deep neural networks. In Proceedings of the IECON 2016–42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 23–26 October 2016; pp. 7046–7051.
33. Zheng, H.; Yuan, J.; Chen, L. Short-term load forecasting using EMD-LSTM neural networks with a Xgboost algorithm for feature importance evaluation. *Energies* **2017**, *10*, 1168. [CrossRef]
34. Rahman, A.; Srikumar, V.; Smith, A.D. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Appl. Energy* **2018**, *212*, 372–385. [CrossRef]