Establishing an AI Model on Data Sensing and Prediction for Smart Home Environment Control Based on LabVIEW

Kai-Chao Yao, Wei-Tzer Huang, Cheng-Chun Wu, and Teng-Yu Chen

Department of Industrial Education and Technology, National Changhua University of Education, No. 2 Shi-Da Road, Changhua City, Taiwan

Correspondence should be addressed to Kai-Chao Yao; kcyao@cc.ncue.edu.tw and Wei-Tzer Huang; vichuang@cc.ncue.edu.tw

Received 31 May 2021; Accepted 6 July 2021; Published 22 July 2021

In this study, the authors aimed to realize a smart home using an AI model that can be integrated with the Laboratory Virtual Instrument Engineering Workbench (LabVIEW) application to realize environment control. The collected input data were outdoor temperature, indoor temperature, humidity, illumination, and indoor person count. The output control decisions included control of air conditioners, dehumidifiers, power curtains, and lights. An artificial neural network was utilized to process the input data for machine learning for the objective of achieving a comfortable environment. In addition, the control decision predictions made by this AI model were analyzed for model loss and model accuracy. This study implemented the model. Specifically, LabVIEW was used to design the sensing component, data display, and control interface, and Python was used to establish the intelligent model. Moreover, by using the web publishing tool built into LabVIEW, remote sensing and control were fulfilled in this implementation.

1. Introduction

Common household items can be transformed into smart devices through sensors. This idea inspires the use of AI models to overcome the problems associated with controlling various appliances in smart homes. With the development of the social economy and the rapid increase in the variety of people’s needs, many appliances can be found in a typical household. However, it remains difficult to manage and control these appliances, while fulfilling households’ needs for comfort, health, energy reduction, and security, through, for example, automatic temperature control and security over the control of devices. When such appliances use artificial intelligence to realize remote and environmentally aware automated control, user comfort is improved significantly [1–3]. A few scientific works [4, 5] have described the development of “smart homes” through machine learning technologies, and some of the practical implementations of this idea, such as that by Salhi, use machine learning algorithms to realize the early detection of gas leakage and for the control of appliances in smart homes. Casaccia proposed a system composed of domestic sensors and machine learning algorithms to measure users’ well-being.

The Internet of Things uses multiple sensors to detect temperature, light, sound, and motion, all of which act as different data sources. Because massive amounts of data are involved, machine learning can be applied to existing home automation systems to significantly enhance their performance. The deployment of IoT and the utilization of AI models should be advanced to usher in a new era of smart home development [6–8].

The most commonly used AI algorithms are artificial neural networks (ANNs), recurrent neural networks (RNNs), and long short-term memory (LSTM). In [9–11], the ANN was used to analyze sensor data and output control decisions. In [12–14], the RNN was used to analyze the behavior of buildings, such as abnormality detection and energy consumption measurement. LSTM applies to IoT data analysis for living activity sensing system any household appliance control can been in [15–17]. Various AI models are widely used in different smart house or building
2 Mathematical Problems in Engineering

The architecture of a typical ANN includes input, hidden, and output layers. Figure 1 illustrates a type of ANNs comprised of three layers, namely, input, hidden, and output layers. ANNs are the most convenient and effective tools for importing AI algorithms. Bhoi applied Python to design a fire detection system for smart homes, and Nadaf designed a smart mirror by using Raspberry Pi for human monitoring and intrusion detection [19, 20]. In the present study, LabVIEW was used along with Python to build smart devices. These two software packages were used collaboratively to manage the programming aspects of data collection, AI model establishment, and control algorithm development. The proposed deployment can be used to develop AI applications pertaining to data sensing and prediction for controlling smart home environments.

2. Preliminaries

2.1. Smart Space Monitoring and Control Based on IoT and LabVIEW. In [18, 21], the authors aimed to build a smart home system and a museum display cabinet with remote monitoring and control. NI myRIO was used as the control unit to create a control function for smart systems, and LabVIEW was used to realize the wireless transmission function in a program designed to run on NI myRIO. The system was capable of detecting each sensor wirelessly. A shared variable function and a built-in web publishing tool were used to construct the remote control function, which was used to control and monitor data immediately on a local PC or a remote tablet. Moreover, the system was capable of activating various relays for the control of, for example, alerts, air conditioning, lights, and powered windows.

2.2. AI Models. An ANN is a computing system inspired by the human nervous system. It is based on theories of massive interconnections and the parallel processing architecture of the biological system. An ANN model is a data-driven mathematical model that can solve problems through machine learning neurons. One of the advantages of ANNs is their capability to identify complex nonlinear relationships between inputs and outputs without using inputs in the form of direct knowledge or physical processes. The most common type of ANNs comprises three layers, namely, an input layer, a hidden layer, and an output layer. Figure 1 illustrates the architecture of a typical ANN [22].

3. System Structure

Figure 2 shows the system structure of data sensing and control prediction for the smart home environment. LabVIEW is used as the graphical user interface (GUI). Python is used to construct an ANN AI model that is embedded in LabVIEW for data analysis and prediction. LabVIEW is responsible for sensor data acquisition, including data on temperature, illuminance, humidity, and the number of people present indoors. In addition, LabVIEW is used to achieve control ability through an NI DAQ.

Once the system is completed, the features of this system module can be integrated into a patrol robot system, such as the robot constructed in [23] to convert it into an Artificial Intelligence of Things (AIoT) smart home robot. Figure 3 shows the structure of an AIoT patrol robot.

4. Main Results

In this study, the user interface design for integrating an AI data sensing and prediction model for smart home environment control was constructed using LabVIEW. The monitoring and control interface is shown in Figure 4. The human–machine interface design includes a (A) user login block, (B) sensor data block, (C) environmental monitoring block, (D) environmental prediction block, and (E) AI mode selection block. Moreover, the LabVIEW Python node that is embedded in LabVIEW is connected to the AI model that must be trained and tested for predicting actual control commands in the household environment. By following this method, the AI model can be constructed as a complex ANN that can generate predictions and perform intelligent control. Furthermore, the features of LabVIEW can aid in the creation of GUIs for analysis and monitoring. Figure 5 shows a block diagram of the smart home environment control system. Figure 6 shows the function block of the NI Python node.

This proposed system is designed to be able to integrate various AI models. These AI models can be selected in the human–machine interface block E. The AI model used in this study is the ANN. Block E can reproduce the detected data, reset the data, pause detection, select an AI model, and display the predicted sensing values. Figure 7 shows block E in the smart home environment control interface. Figure 8 shows the part of the block diagram pertaining to the integration of the embedded AI model in LabVIEW.

4.1. Sign-In Function. For system security, identity verification is deployed before the operating system, as illustrated in block A of Figure 4. If the login process fails, the monitoring system cannot be entered. Figure 9 and Figure 10 depict images of system login success and failure, respectively.

4.2. Environment Sensing and Control. In the environmental monitoring part, the human–machine interface displays the real-time environmental condition data captured using sensors. Block B in Figure 3 depicts the real-time data captured by the sensors, including the current number of people and the current temperature, humidity, and illuminance. Block C in Figure 3 represents the display data converted into suitable units, as illustrated in Figure 11. Figure 12 presents a partial program diagram of the environmental monitoring and control, and Figure 13 depicts a circuit diagram of the designed sensing data acquisition.
system. The acquired sensing data are sent to the AI model constructed using Python, and the predictions of the AI model are used for home appliance control. On the control device, an NI-DAQmax is used to output the control signals and perform remote relay on-off control.

4.3. Real-Time Data Collection and Prediction. In Section 4.2, the ANN-based AI model constructed using Python was described. This human–machine interface system was designed to facilitate the addition of different AI models for integration. Figure 14 shows the data predicted by the trained ANN model by using the person count, temperature, humidity, and illuminance data captured in Section 4.2. In the human–machine interface in Figure 3, these data are shown in block D. Figure 15 depicts the part of the program diagram for real-time data processing and data prediction.

4.4. AI Model and Learning. In this study, we used Python to construct the AI model depicted in Figure 16, and we used machine learning techniques to predict environmental conditions. Machine learning is a part of artificial intelligence. To facilitate efficient and effective learning by the AI model, the learning process is generally divided into two steps: training and testing. In the training step, historical data, comprising features, and labels are used. After the AI model process testing, it can output its predictions. The AI learning model is analogous to the brain of an artificial intelligence system. If one desires to achieve a more intelligent machine, one must construct a neural network model that is trained under greater complexity. The construction process is depicted in Figure 17. In this study, we used a supervised learning method to establish the ANN and to input standardized features and labels for training and learning. In the process, the AI model is repeatedly trained and adjusted to achieve high speed and high prediction accuracy.

The learning model developed in this study uses an ANN to realize environmental prediction. The ANN contains many neurons, some of which are responsible for receiving data and others for transmitting data. It is an adaptive Internet network, and its structure can be represented using a
simple neuron model, such as the M-P neuron model depicted in Figure 18.

In an M-P model, a certain neuron may receive multiple input signals simultaneously, as shown in Figure 18. Because biological neurons have different synaptic properties and synaptic strengths, they have different effects on neurons. They can be represented by the weights $\omega_1, \omega_2, \ldots, \omega_n$, and their positive and negative values denote prominent
excitement and inhibition in biological neurons, respectively. The magnitudes of prominent excitement and inhibition represent the different strengths of the prominent connections. To express a threshold \( \theta \) (threshold), also called a bias (bias), neurons have two states: excited and inhibited. Under normal circumstances, most neurons are in an inhibited state. However, when a neuron is stimulated and its potential exceeds a threshold \( \theta \), the neuron is activated and excited, and it subsequently transmits chemical substances, that is, information, to other neurons. The associated data processing flow is represented by the mathematical model expressed in the following equation:

\[
y = f \left( \sum_{i=0}^{n} \omega_i x_i - \theta \right).
\]

The M-P model shown in Figure 18 adjusts the weight assigned to each input value during the machine learning process and forms an intelligent model by adjusting the weight \( \omega \) or the threshold \( \theta \). The greater the signal value entering a neuron, the easier it is to trigger neurons and the greater is the influence on the operation of the neural network. Conversely, the smaller the signal value, the weaker is the effect. If the signal is too small, it can even be ignored for computational savings and smaller errors in the output value. The output value is adjusted and converted using an activation function. As depicted in Figure 19, Python is applied to construct the M-P model and the activation function uses a function formulated in terms of \( \tanh \). To improve the accuracy of learning, the neural network is developed as a multilayer perceptron with multiple hidden layers (Figure 20). Figure 21 shows the setting of the activation function during the establishment of the MLP in this study. The rectified linear unit (ReLU) function is used as the activation function after the calculated input value is normalized. Furthermore, the model standardizes its output value ranges and the relationship between the output values. Figure 19 depicts the ANN-MLP model used in this study. Figure 22 depicts the architecture of the ANN-MLP model.

Tables 1 and 2 summarize parts of the training and testing datasets used in the training and testing processes, respectively.

| Session in | Session out |
|------------|-------------|
| Module path | Function name |
| Error in (no error) | Return type |
| Return type | Input parameter |

Figure 5: Block diagram of the smart home environment control system.

Figure 6: Function block of NI Python node.
model does not perform as expected and the weighting parameters must be adjusted to achieve the desired performance. Figure 23 depicts a flowchart of the verification and evaluation steps of the learning model. The steps for verifying the model are as follows:

Step 1: prepare the learning dataset
Step 2: separate the training and test data into input data and output data
Step 3: input the training data into the neural network for training
Step 4: compare the training results of the neural network with the testing output data to observe the loss function for verification
Step 5: feedback the comparison difference in Step 4 and adjust the weight parameters
Step 6: update the neural network parameters (back to Step 3)

After the fourth step, the feedback weighting parameter adjustment module in the machine learning algorithm is executed to adjust the model for obtaining more accurate predictions and improved judgment abilities. The neural network can modify the weighting parameter in terms of the loss function to reduce the error to achieve more accurate predictions. Some commonly used indicators are the mean squared error (MSE) and cross entropy error (CEE). In statistics, the MSE is expressed in terms of an estimated function, \( T \), for an unobservable population number \( \theta \). The following equation expresses the definition of the MSE:

\[
\text{MSE}(T) = E((T - \theta)^2). \tag{2}
\]

The MSE is the expected value of the error squared. The error is the difference between the estimated value and the estimated quantity. MSE satisfies the following equation:

\[
\text{MSE}(T) = \text{var}(T) + (\text{bias}(T))^2. \tag{3}
\]
Among them, the bias \((T)\) is the difference between the expected value of the estimated function and the unobservable population number, similar to the threshold \(\theta\) or the bias of neuron activity, which satisfies the following equation:

\[
\text{bias}(T) = E(T) - \theta.
\]  

(4)

In this study, model loss is verified by using the MSE function. Moreover, the backpropagation method is used to adjust and modify the model. The respective mathematical expressions are given in the following equations:

\[
\text{MSE} = \frac{1}{k} \sum_{i=1}^{k} |y_i - t_i|, \quad (5)
\]

\[
L(E) = \frac{1}{2} \sum_{k} (y_k - t_k)^2. \quad (6)
\]

In equations (5) and (6), \(y\) denotes the output data or the prediction result of the machine learning algorithm, and \(t\) denotes the correct solution data. The loss of the training output function and a dataset with the correct data can be...
used to verify the process. By taking the difference and summing the square values of these differences, we can obtain the mean square error. This value is used as the difference loss judgment of the data, and it is used to implement feedback correction. If the loss is decreasing, a better model can be obtained. Figure 24 depicts a model loss comparison diagram between the model after training and that after testing. In this figure, the training and testing losses
Figure 13: Designed sensing data acquisition circuit.

Figure 14: Comparison of real-time and predicted data.
exhibit decreasing trends, which indicates that the training results are reasonable.

In Figure 24, after 25 training runs, although the model verification loss tends to decrease, database normalization must be considered when processing a large quantity of data. Figures 25 and 26 show whether the data in the database are standardized. The difference graph of the loss function can be obtained after 100 runs of posttraining.

When much data are collected, complex neural networks must use appropriate algorithms for data processing. Gradient descent is one such suitable algorithm. This method can be used to identify the errors in neural networks.

**Figure 15:** The program diagram pertaining to real-time processing and data prediction.

**Figure 16:** Establishment of AI model in Python.

| Layer (type) | Output Shape | Param # |
|--------------|--------------|---------|
| dense (Dense) | (None, 50) | 400     |
| dense_1 (Dense) | (None, 50) | 2550    |
| dense_2 (Dense) | (None, 4) | 204     |

Total params: 3,154
Trainable params: 3,154
Non-trainable params: 0
Figure 27 shows the graph of model loss with 10,000 data points as inputs. Many types of gradient descent methods can be used to adjust the weights. In this study, we used the stochastic gradient descent (SGD) method to determine the gradient of the weighting parameters. This method can update the weight, \( \omega \), in the direction of the gradient. Specifically, \( L \) is the loss function, \( \eta \) is the learning rate, and \( \partial L / \partial \omega \) is the gradient of the loss function with respect to weight. The mathematical equation is as follows:

\[
\begin{align*}
\omega_1 &= \omega_0 - \eta \frac{\partial L}{\partial \omega_0} \\
\omega_2 &= \omega_1 - \eta \frac{\partial L}{\partial \omega_1} \\
&\vdots
\end{align*}
\]

(7)

In the preceding verification model, the training process endows the model with its prediction ability. If perfect
If prediction is excessively pursued in the training process, the obtained weighting parameters will cause overfitting and lead to poor actual predictions. In the simulation, model loss between the training and testing processes can be observed. This is normal. Therefore, regularization is applied to maintain the existing features and reduce the influence of a few unimportant features. The trained model should not overly rely on the weight loss during training. The revised mathematical function is expressed as in the following equation:

$$L(E) = \frac{1}{2} \sum_k (y_k - t_k)^2 + \lambda \sum \omega_i,$$  

(8)

where $\lambda$ represents the choice between the prediction error and the weight term. When $\lambda$ is larger, the model focuses less on the prediction gap. According to equation (8), which is used for regularization, the model loss between training and testing approaches zero, as illustrated in Figures 28 and 29.

After modification of the training model, the accuracy rate can reach 95%, as illustrated in Figure 30. The following

Figure 19: (ANN) M-P model program construction process.

Figure 20: Schematic of the ANN-MLP model.

Figure 21: ANN-MLP model activation function setting.
figures depict the real-time status and prediction status, respectively, of temperature, humidity, illuminance, and person count. In these figures, the actuality and predict lines denote the actual and the forecast situations, respectively. Figure 31 depicts 24 h real-time relative humidity and the forecast state diagram. Figure 32 depicts the real-time relative humidity and forecast state diagram for a week. Figure 33 illustrates the 24 h real-time temperature and predicted state diagram. Figure 34 presents the real-time temperature and predicted state for a week. Figure 35 depicts the real-time illuminance and forecast state diagram for a week. Figure 36 depicts the real-time person count for a week and the forecast state diagram.

![Figure 22: ANN-MLP model architecture.](image)

**Table 1: Part of the training data recorded between 2001 and 2018.**

| Day | Time | Temperature | Humidity | People | Illumination | Time | Temperature | Humidity | People | Illumination |
|-----|------|-------------|----------|--------|--------------|------|-------------|----------|--------|--------------|
| 1/1 | 1:00 | 14          | 60       | 2      | 95           | 2:00 | 16.7        | 81       | 4      | 115          |
| 1/1 | 12:00| 13.7        | 63       | 4      | 115          | 13:00| 16.4        | 81       | 4      | 117          |
| 1/3 | 23:00| 13.4        | 65       | 4      | 117          | 24:00| 15.9        | 82       | 0      | 18           |
| 1/1 | 4:00 | 13.7        | 62       | 0      | 18           | 5:00 | 15.3        | 85       | 0      | 18           |
| 1/2 | 7:00 | 12.4        | 67       | 0      | 18           | 8:00 | 15.6        | 83       | 1      | 87           |
| 1/4 | 9:00 | 11.6        | 71       | 1      | 87           | 10:00| 15.8        | 84       | 1      | 89           |

**Table 2: Part of the testing data recorded between 2019 and 2020.**

| Day | Time | Temperature | Humidity | People | Illumination | Time | Temperature | Humidity | People | Illumination |
|-----|------|-------------|----------|--------|--------------|------|-------------|----------|--------|--------------|
| 1/1 | 1:00 | 16.9        | 80       | 4      | 117          | 2:00 | 16.7        | 81       | 4      | 119          |
| 1/1 | 2:00 | 16.7        | 81       | 4      | 119          | 3:00 | 16.4        | 81       | 4      | 116          |
| 1/1 | 3:00 | 16.4        | 81       | 4      | 116          | 4:00 | 15.9        | 82       | 2      | 95           |
| 1/1 | 4:00 | 15.9        | 82       | 2      | 95           | 5:00 | 15.3        | 85       | 0      | 18           |
| 1/1 | 5:00 | 15.3        | 85       | 0      | 18           | 6:00 | 15.6        | 83       | 0      | 17           |
| 1/1 | 6:00 | 15.6        | 83       | 0      | 17           | 7:00 | 15.8        | 84       | 0      | 17           |
Figure 23: Flowchart of verification and evaluation steps of the learning model.

Figure 24: Comparison of model loss between model training and testing.
Figure 25: Model loss without standard database data.

Figure 26: Model loss with standard database data.

Figure 27: Model loss of training and testing after 10,000 data inputs.

Figure 28: Model loss after normalization $\lambda = 0.001$.

Figure 29: Model loss after $\lambda = 0.01$. 

Figure 30: State diagram of model accuracy for model training and testing.

Figure 31: 24h real-time humidity and forecast state diagram.

Figure 32: Real-time relative humidity and forecast state diagram for a week.
5. Conclusion

In this study, home environment sensing and prediction were successfully achieved to realize the smart control of a home environment. LabVIEW was utilized as the main interface of this smart home system. An ANN AI model constructed using Python was integrated into the LabVIEW system. This combination provided the advantages of both software systems, namely, the excellent monitoring and control interface of LabVIEW and the excellent AI algorithm calculation capabilities of Python. In this study, the sign-in protection function, smart home environment sensing and control, real-time data collection and prediction, and AI model and training were separated, as described in the study. Moreover, different AI models can be integrated into the system, such as the RNN and LSTM. Furthermore, the proposed system can be applied to many different systems to endow them with AI capabilities, such as robots and vehicles.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was funded by the Ministry of Science and Technology, Taiwan (MOST 109-2511-H-018-018-MY3).

References

[1] J. Jaihar, N. Lingayat, P. S. Vijaybhai, G. Venkatesh, and K. P. Upla, "Smart home automation using machine learning algorithms," in Proceedings of the 2020 International
[2] H. Verma, M. Jain, K. Goel, A. Vikram, and G. Verma, "Smart home system based on Internet of "Things," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 11, no. 2, pp. 2073–2075, 2020.

[3] A. Kazarian, V. Teslyuk, I. Tsomits, and M. Mashevskaya, "Units and structure of automated "smart" house control system using machine learning algorithms," in Proceedings of the 2017 14th International Conference the Experience of Designing and Application of CAD Systems in Microelectronics (CADM), pp. 364–366, Lviv, Ukraine, February 2017.

[4] L. Salhi, T. Silverston, T. Yamazaki, and T. Miyoshi, "Early detection system for gas leakage and fire in smart home using machine learning," in Proceedings of the 2019 IEEE International Conference on Consumer Electronics (ICCE), pp. 1–6, Las Vegas, NV, USA, February 2019.

[5] S. Casaccia, L. Romeo, A. Calvaresi et al., "Measurement of users' well-being through domotic sensors and machine learning algorithms," IEEE Sensors Journal, vol. 20, no. 14, pp. 8029–8038, 2020.

[6] A. J. Majumder and J. A. Izaguirre, "A smart IoT security system for smart-home using motion detection and facial recognition," in Proceedings of the 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1065–1071, Madrid, Spain, July 2020.

[7] Z. Zhang, T. He, M. Zhu, Q. Shi, and C. Lee, "Smart tribo-electric socks for enabling Artificial Intelligence of Things (AIoT) based smart home and healthcare," in Proceedings of the 2020 IEEE 33rd International Conference on Micro Electro Mechanical Systems (MEMS), pp. 80–83, Vancouver, BC, Canada, January 2020.

[8] D. Ganesh, G. Seshadri, S. Sokkanarayanan, S. Rajan, and M. Sathiyarayanan, "IoT-based google duplex artificial intelligence solution for elderly care," in Proceedings of the 2019 International Conference on Contemporary Computing and Informatics (ICCI), pp. 234–240, Singapore, December 2019.

[9] Y. Arora, H. Pant, and Banita, "Home automation system with the use of Internet of Things and Artificial intelligence," in Proceedings of the 2019 International Conference on Innovative Sustainable Computational Technologies (CJICT), pp. 1–4, Dehradun, Uttarakhand, India, October 2019.

[10] A. Kazarian and V. Teslyuk, "Optimization of neural network structure for smart house systems," in Proceedings of the 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON), pp. 562–565, Lviv, Ukraine, October 2019.

[11] S. Aggarwal and A. Kumar, "A smart irrigation system to automate irrigation process using IOT and artificial neural network," in Proceedings of the 2019 2nd International Conference on Signal Processing and Communication (ICSPC), Coimbatore, India, March 2019.

[12] H. Zhao et al., "Learning-based occupancy behavior detection for smart buildings," in Proceedings of the 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 954–957, Montreal, QC, August 2016.

[13] A. Legrand, B. Niepceron, A. Cournier, and H. Trannois, "Study of autoencoder neural networks for anomaly detection in connected buildings," in Proceedings of the 2018 IEEE Global Conference on Internet of Things (GCIoT), pp. 1–5, Alexandria, Egypt, December 2018.

[14] A. Zaza, S. Al-Emadi, and S. Kharroub, "Modern QoS solutions in WSAN: an overview of energy aware routing protocols and applications," in Proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), pp. 581–589, Doha, Qatar, February 2020.

[15] R. A. Rashid, L. Chin, M. A. Sarjari, R. Sudirman, and T. Ide, "Machine learning for smart energy monitoring of home appliances using IoT," in Proceedings of the 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), pp. 66–71, Zagreb, Croatia, July 2019.

[16] N. Najari, S. Berlemont, G. Lefebvre, S. Duffner, and C. Garcia, "Network traffic modeling for IoT-device re-identification," in Proceedings of the 2020 International Conference on Omni-Layer Intelligent Systems (COINS), pp. 1–6, Barcelona, Spain, July 2020.

[17] T. Matsui, K. Onishi, S. Misaki, M. Fujimoto, H. Suwa, and K. Yasumoto, "Easy-to-deploy living activity sensing system and data collection in general homes," in Proceedings of the 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 1–6, Austin, TX, USA, March 2020.

[18] K.-C. Yao, W.-T. Huang, L.-C. Hsu, J.-Y. Lai, and C.-K. Yang, "ICIC express letters, part B: applications," An International Journal of Research and Surveys, vol. 7, no. 11, 2016.

[19] S. K. Bhoi, S. Kumar Panda, B. Narayan Padhi et al., "FireDS-IoT: a fire detection system for smart home based on IoT data analytics," in Proceedings of the 2018 International Conference on Information Technology (ICIT), pp. 161–165, Bhubaneswar, India, December 2018.

[20] R. A. Nadaf, M. Rubina, P. Sujata, and V. M. Bonal, "Smart mirror using Raspberry Pi for human monitoring and intrusion detection," in Proceedings of the 2019 1st International Conference on Advances in Information Technology (ICAIT), pp. 116–121, Chikmagalur, India, July 2019.

[21] K.-C. Yao, W.-T. Huang, C.-Y. Lo, L.-C. Hsu, and J.-S. Chiang, "IoT application on remote monitoring and control of museum display cabinet," ICIC Express Letters, vol. 10, no. 9, pp. 2249–2257, 2016.

[22] R. S. Govindaraju, "Artificial neural networks in hydrology. I: preliminary concepts," Journal of Hydrologic Engineering, vol. 5, no. 2, pp. 115–123, 2000.

[23] K.-C. Yao, W.-T. Huang, and L.-C. Hsu, "Evaluation of the established IoT smart home robot teaching model based on embedded thematic-approach strategy," Mathematical Problems in Engineering, vol. 2020, Article ID 6696155, 10 pages, 2020.