Sensors and Materials, Vol. 36, No. 8 (2024) 3491–3517
MYU Tokyo

S & M 3747

Intelligent Factory with Environment Quality Control Based on Fuzzy Method through Deep Reinforcement Learning

Wen-Tsai Sung,[1] Jenny Aryani,[1] and Sung-Jung Hsiao[2*]

[1]Department of Electrical Engineering, National Chin-Yi University of Technology, Zhongshan Rd, Section 2, No. 57, Taichung City 411030, Taiwan
[2]Department of Information Technology, Takming University of Science and Technology, No. 56, Sec. 1, Huanshan Rd., Neihu District, Taipei City 11451, Taiwan

(Received September 15, 2023; accepted May 1, 2024)

Keywords: environment quality system, deep reinforcement learning, fuzzy method, Internet of Things (IoT)

This study is aimed at developing an intelligent factory control system for improving environment quality and reducing electricity consumption. By automating intelligent equipment and leveraging internet networks, the system enables the remote monitoring and management of environmental conditions. We combine the fuzzy method and deep reinforcement learning (DRL) to handle complex factory data and optimize decision-making. The fuzzy method uses fuzzy sets and rules to generate accurate outputs from the data. On the other hand, the DRL system learns optimal policies by interacting with the environment using environment quality, central air conditioner (AC), and alarm data. Hardware implementation uses an ESP32-S microcontroller to send data to Google's Firebase cloud for seamless management and monitoring through a mobile app or website. The study involves developing 36 fuzzy rules and creating 10 models with different combinations of hidden layers, epochs, and learning rate values. Among the fuzzy inference system (FIS)-DRL modeling results, the fourth model stands out as the preferred option to proceed with the experiment, as it achieves the highest accuracy of 91.16%. Note that this model also exhibits a loss value of 1.64% and an incredibly short inference time of only 3 ms. The proposed system offers benefits such as enhanced energy efficiency and reduced costs, making it ideal for intelligent factories. By optimizing resource usage, it will contribute to sustainable development in various industries.

1. Introduction

A factory is an industrial workplace that can be found throughout the world. Factories provide jobs to a large number of employees. However, it is important to ensure that the working environment in the factory meets applicable safety, health, and legal compliance standards.(1) Efforts to protect workers and enforce environmental sustainability are also important focuses in carrying out factory operations in a responsible manner. The most prominent task in knowing whether or not factory conditions are good is ensuring good air condition in the factory. The air

*Corresponding author: e-mail: sungjung@gs.takming.edu.tw
https://doi.org/10.18494/SAM4661

ISSN 0914-4935

condition in the factory is very important to the safety, health, and comfort of workers. Factors that must be considered in relation to the air condition in the factory are, for example, indoor air quality (IAQ), temperature and humidity, ventilation, dust, and smoke control.[2] The factory must comply with applicable occupational safety regulations and standards regarding indoor air conditions and take the necessary precautions to maintain healthy air quality in the work environment. According to the Decree of the Minister of Health of the Republic of Indonesia number 1405/MENKES/SK/XI/2002 Concerning Environmental Health Requirements for Office and Industrial Work, to prevent health problems and environmental pollution in offices and industry, a suitable room temperature is 18–28 ℃ and a suitable humidity is 40–60%. It is also necessary to prohibit smoking in the workspace.[3]

A fuzzy inference system (FIS) for air quality allows the handling of uncertainty and ambiguity associated with air quality measurement. FIS classifies parameters by a reasoning process and integrates them into an air quality index.[4] The data generated by the sensors in a multisensor system is coordinated by the FIS. FIS is used to model human knowledge and address uncertainties in air quality data.[5] Assimakopoulos *et al.*[2] applied the fuzzy logic approach to assess the level of indoor air pollution to determine the comfortable IAQ level.

Deep reinforcement learning (DRL) is an alternative method of selection that can be used to make decisions based on raw data from sensors that measure environmental quality parameters. DRL is a branch of machine learning (ML) that combines two main approaches, namely, reinforcement learning (RL) and deep learning (DL). DRL focuses on teaching agents to take action in a dynamic environment to achieve certain goals and receive positive or negative rewards in accordance with the actions they take.[6] Heo *et al.*[7] used the DRL deep Q-network (DQN) algorithm to design ventilation control systems, where the energy consumption was reduced by up to 14.4% for validation dataset time intervals and IAQ was raised from an unhealthy to acceptable level. Hu *et al.*[8] designed an air scope to monitor IAQ sensing using DRL, where distributed deep Q-learning (DDQL) was employed. The proposed DDQL algorithm performed 8% better than greedy algorithms and 24% better than random strategies. Valladares *et al.*[9] also used DRL in their research to investigate how CO_2 levels IAQ. Their results showed that the proposed agent in their system has a superior predicted mean vote index and that the CO_2 level is 10% lower than in current control systems while consuming about 4–5% less energy.

In this study, the readings from each sensor are transmitted to the microcontroller, which is connected to Wi-Fi. Once connected, the data are transferred to the cloud utilizing Firebase's real-time database as the cloud platform. The system developed in this study combines FIS and DRL as an environment quality classification system in a factory environment. This combination leverages the strengths of DRL in learning from complex data and situations, and FIS in handling uncertainty and simplifying output interpretation. As a result, a more accurate, adaptive, and comprehensible environment quality classification system is achieved.

The multisensor system continuously collects data that are then input into FIS. The collected data are converted into fuzzy variables, and fuzzy rules are established on the basis of these variables. The fuzzy inference process, implemented by the Mamdani method, considers the fuzzy inputs and rules to generate a set of fuzzy outputs. These fuzzy outputs are combined and aggregated to produce an overall environment quality index.

Defuzzification is then performed to convert the fuzzy output set into crisp values, making them easily understandable. This can be achieved by methods such as the centroid or weighted average method. Subsequently, DRL modeling is conducted, providing output to factory workers regarding the observed behavior of the factory objects. The data integrated by FIS is further processed within the DRL framework in the subsequent stage.

In this study, the action taken is the determination of the environment quality conditions and the improvement of air circulation by automatically turning on the central air conditioner (AC), as well as the activation of a buzzer as an alarm when the factory conditions are exceptionally poor. By implementing the DRL stage after the FIS stage, the system can learn and optimize adaptive and intelligent decision-making policies using the observed environmental conditions and FIS results. It dynamically adjusts environmental parameters to achieve the desired environment quality in the factory.

2. Literature Review

In this section, we describe the system literature including explanations of previous research, fuzzy methods, Markov decision processes (MDPs), and DRL in environment quality systems.

2.1 Related work

In related work, Haydari *et al.*[10] used the DRL model to control urban traffic signals in San Francisco. According to their study, there are several factors that affect the controllability of the DRL system, one of which is the reward function. On the other hand, Shahbazi *et al.*[11] used a DRL system combined with a block chain in the development of smart home security. According to their study, DRL is a learning-based algorithm that has the most effective aspects to improve system performance to an accuracy higher than 96.9%, which was previously obtained by them. Next, Kumar *et al.*[12] controlled a dynamic and intelligent traffic light control system using a combination of DRL and FIS models. DRL is dedicated to dynamically generating traffic light phases, whereas FIS is dedicated to analyzing traffic behavior, heterogeneity, and dynamism where DRL is in a deep state. The results proved that using this system results in high efficiency and effectiveness. In this study, DRL combined with FIS was implemented to optimally control and monitor the factory environment quality by examining environmental conditions in terms of temperature, humidity, smoke, and dust.

2.2 Fuzzy method

The fuzzy method in FIS is to use fuzzy logic to make decisions referring to predetermined rules. Fuzzy Mamdani (FM), often called the max–min method, is one of the most commonly used types of FIS.[13] FM is implemented in the development of our system. FM was first developed by Ebrahim Mamdani in 1975.[14] The FIS architecture is shown in Fig. 1; it consists of several stages, namely, fuzzification, fuzzy data rule base, inference, and defuzzification.[15] Fuzzification, the first stage, is a process of transforming crisp input into real fuzzy sets. The

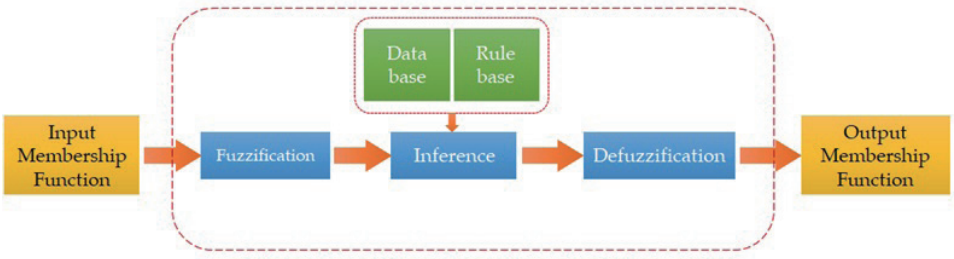


Fig. 1. (Color online) Fuzzy inference system.

second stage involves the rule base data, where the combining function allows the rule application function to be realized by obtaining the maximum rule value from the fuzzy set and applying it to the output using the AND operator. Equation (1) shows the process of combining membership functions using the max operator.

$$\mu_{sf}(x_i) = \max\left(\mu_{sf}(x_i), \mu_{kf}(x_i)\right) \tag{1}$$

The max operator is used to determine the maximum membership degree between $\mu_{sf}(x_i)$ and $\mu_{kf}(x_i)$. $\mu_{sf}(x_i)$ represents the membership function for a particular fuzzy set, and $\mu_{kf}(x_i)$ represents the membership function for another fuzzy set. The third stage is inference, where mapping is performed from the input to the output using fuzzy logic. The last stage in the fuzzy method is the defuzzification of the combined fuzzy set converted into crisp values that can be used as system output. The process of determining the fuzzy center point is expressed by

$$Z = \frac{\int z \cdot z\mu(z)dz}{\int z \cdot \mu(z)dz}. \tag{2}$$

Z represents the result of defuzzification or the center point of the fuzzy area, and $\mu(z)$ represents the membership value resulting from the fuzzification process. dz is used to separate z variables in the integral process The integral of $z\mu(z)dz$ expresses the moments for all regions resulting from composition rules.

2.3 MDP

MDP is an important component in DRL. MDP is a mathematical model used to describe decisions taken in contexts that may change over time.[16] MDP is implemented in this study as a framework for modeling and optimizing action decisions made by learning agents contained in DRL. Theoretically, the MDP formulation refers to the Bellman equation represented by

$$Value(s) = MAx_{action}\left\{\sum_{s',r} P(s', r \mid s, a) \times \left[r + \gamma \times Value(s')\right]\right\}. \tag{3}$$

Value(*s*) represents the value of state *s*, where, in this study, *s* indicates dust particles, smoke content, temperature, or humidity. The maximum possible number of actions is represented by MAx_{action}. In the Bellman equation, $\sum_{s',r} P(s',r \mid s,a) \times \left[r + \gamma \times Value(s') \right]$, which is the sequential probability of changing *s* to *s*′, is integrated through the reward *r* when action *a* is performed. γ states the discount factor generated to balance the direct and future rewards.

2.4 DRL

DRL is a combination of DL and RL and has complex computational dimensions. DRL works on conditions in the environment and acts through actions processed by agents that are integrated with DL.[16] DL or the deep neural network (DNN) is a neural network that mimics the function of neurons in the human brain. DL training uses a neural network layer to collect and process input data and output a specific output layer.[17] In a neural network, there is an input layer and one or more hidden layers that are connected sequentially through connections that have weights.[18] Weight update can be completed using

$$\Delta W_{ij}(t+1) = \Delta W_{ij}(t) + n\frac{\partial C}{\partial W_{ij}}. \tag{4}$$

Here, $\Delta W_{ij}(t + 1)$ is the change in weight between the two networks at time iteration $t + 1$. $\Delta W_{ij}(t)$ is the change in weight that occurred in the previous iteration and is added to *n*, which is the learning rate. The higher the learning rate, the faster the system can adapt to changing conditions or demands in the factory. $\partial C/\partial W_{ij}$ is a partial derivative to measure changes in weight to changes in cost function by providing information on the size of the weight change needed to achieve the system objectives.

In this study, five layers are used: one input layer, three hidden layers, and one output layer. In the input layer, there are four parts, namely, temperature, humidity, smoke, and dust particles. Figure 2 shows the neural network architecture in DL.

Moreover, RL is trained using data originating from the interaction of the agent and the subject's environment,[19] where the agent receives a reward for the best achievement. DQN is an approach in RL, where a neural network models and estimates the *Q* value function in MDP and then updates the *Q* value estimate on the basis of experience. In conclusion, DQN is one of the RL approaches used to solve the MDP problem.

$$Q(s,a) = (1 - \alpha) \times Q(s,a) + \alpha \times (r + \gamma \times \max(Q(s',a'))) \tag{5}$$

In Eq. (5), $Q(s,a)$ is the *Q* function used to estimate the expected cumulative reward value when in state *s* and taking action *a*. The $(1 - \alpha)$ learning rate controls the extent of updating the *Q* value in each iteration. The reward *r* is the reward received when taking action *a* in state *s*. γ is a discount factor comparing how much *r* is obtained now and in the future. max(*Q*) is the maximum expected cumulative reward when in state *s*′ and taking action *a*′, and is based on the new state and action generated after performing *a* and being in state *s*.

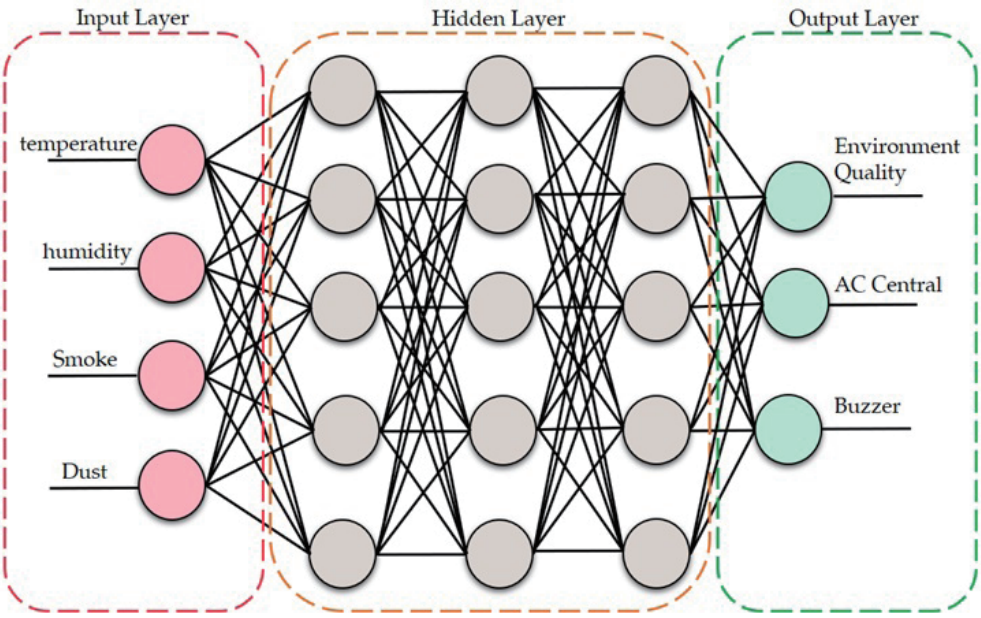


Fig. 2. (Color online) DL architecture implemented in an intelligent factory.

RL can be used to adjust neural network parameters adaptively on the basis of feedback or rewards obtained from the environment. Neural networks can learn interactively and produce better decisions.[20] The stages in DRL in this context involve several elements, namely, the state space represented by *s*, action space *a*, transition function *t*, and reward function *r*. RL architecture can be seen in Fig. 3. DRL will depict the control environment of the factory environment quality as *s*, where each state represents specific environmental conditions such as dust particle level, smoke level, temperature, and humidity. In action space a, the agent can choose actions to control the central AC system. The agent determines not only the condition of AC on and off, but also the conditions of action for environment quality by sending notifications through the mobile application and buzzer as alarms so that the whole is integrated into one complete action. The transition function describes the changing state of the factory environment on the basis of the actions taken by the agent. Changes in factory conditions can be affected by the operation of the control system. The final stage involves the reward function r, which provides feedback to the agent on the basis of the environmental state and actions taken. This reward function is designed to encourage the agent to perform actions that result in good environment quality and minimize pollution in the workspace.

3. Materials and Method

Here, we describe the system architecture, model configuration, hardware and software, and the experimental results that will be applied to this study.

Fig. 3. (Color online) RL architecture.

3.1 System architecture

The system architecture is shown in Fig. 4. There are four layers, namely, the (1) physical layer, (2) network layer, (3) data analysis layer, and (4) application layer. The physical layer comprises the combination of the microcontroller and temperature, humidity, smoke, and dust sensors. The data of the environmental conditions are obtained by the sensors and transmitted to the microcontroller. The results of data processing by the microcontroller are then sent to the network layer via the Wi-Fi module connection. The Wi-Fi module is connected to the internet network to transmit real-time data of factory environmental conditions to Firebase cloud. Data of the real-time conditions are transmitted from the sensors to the data analysis layer. In the data analysis layer, FIS receives and processes the data from sensors by making rules to output decisions in the form of an environment quality index. On the DRL side, the actions to be taken by the agent are determined from the results of FIS decision rules by updating and improving control policies using the feedback received. DRL can optimize control policies on the basis of the rewards received for factory environmental conditions. Furthermore, the output value is displayed by the application layer, namely, a mobile-based application that monitors and controls factory conditions. In addition, since the microcontroller cannot directly control high-voltage electrical appliances, a relay is used. The application layer also includes a relay that controls the central AC turn-on and turn-off operations and a buzzer that provides a notification of any error that may occur in the factory conditions.

3.2 Fuzzy architecture

As explained in the previous section, in this study, we utilize the FM method. In the Mamdani process, it is important to determine the input membership function (IMF) and output membership function (OMF). A membership function is a mathematical function that determines how much each linguistic set contributes to the membership value, where each membership function has a fuzzy variable that is converted into a membership value. IMF converts the fuzzy input values into membership values used in the inference process, while OMF converts the fuzzy output values into membership values used in the defuzzification process to obtain a concrete output value.[13] With the help of IMF and OMF, the defuzzification process is

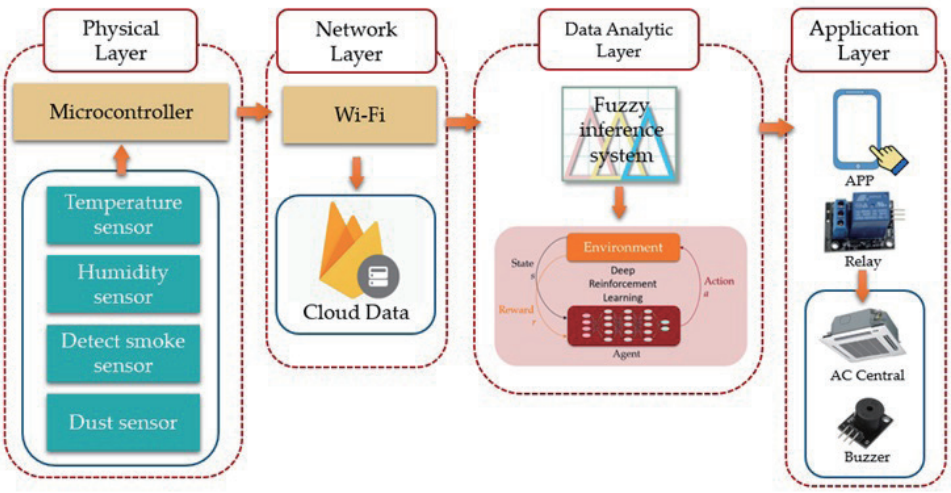


Fig. 4. (Color online) Proposed system architecture.

performed by calculating the weighted average of all linguistic set contributions to an output fuzzy set, resulting in a more concrete and understandable output value. Figure 5 shows the member functions of FIS Mamdani. The Mamdani fuzzy method comprises four IMFs: temperature, humidity, dust particle, and smoke. There are also three OMFs: environment quality, central AC, and buzzer. Each IMF has specific conditions based on its characteristics; the detailed range of each IMF can be found in Table 1, while Table 2 shows the OMF categories.

In the fuzzification stage, the IMFs for temperature, humidity, dust particle, and smoke are transformed into fuzzy sets using membership functions. For instance, the dust particle levels are labeled as normal and risk. After determining the range of each fuzzy set, the subsequent step involves creating fuzzy rules based on Eq. (6), where x represents the IMF, A is the IMF range, and y represents the OMF category B in the fuzzy rules.

$$\text{If } x \text{ is } A, \text{ then } y \text{ is } B. \tag{6}$$

Each fuzzy rule is applied by considering the degree of membership of the input, and the degree of membership of the output is generated. An example of rules in the system we employ are shown in Eq. (7).

$$\text{If temperature is hot, humidity is dry, then central AC is On.} \tag{7}$$

The final stage is defuzzification, where the fuzzy output set is transformed into a crisp value that can be used for decision-making and controlling devices in the environment quality system. Mathematically, a crisp value can be expressed as

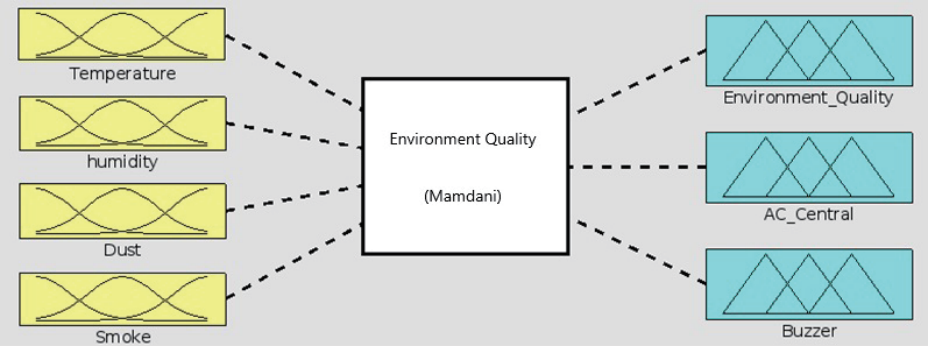


Fig. 5. (Color online) Membership functions of FIS.

Table 1
Ranges of IMFs.

IMF		Range of IMF
Temperature (°C)	Cool	0–20
	Normal	21–28
	Hot	>28
Humidity (%)	Dry	0–45
	Normal	46–65
	Humid	66–100
Dust particle (μg/Nm^3)	Normal	0–10
	Risk	11–40
Smoke (μg/m^3)	Normal	0–500
	Risk	501–10000

Table 2
Categories of OMFs.

OMF	Categories	Range of values
Environment quality	Poor	1–50
	Moderate	50–70
	Good	70–100
Central AC	On	1
	Off	0
Buzzer	On	1
	Off	0

$$Crisp_{value} = \frac{\sum_I u_i \times x_i}{\sum_I u_i}, \tag{8}$$

where u_i represents the membership degree in category i and x_i represents the crisp value for each category i. In environment quality, if the crisp value generated falls within the range of 0–50, it can be categorized as "Poor". If the crisp value falls within the range of 51–70, it is categorized as "Moderate". If the crisp value is above 71, then the environment quality is categorized as "Good". To determine the range, the defuzzification process uses the centroid

method, where category i is determined on the basis of the center of gravity of the curve of each membership category; the resulting value will represent the median value of each category. Then, objective decisions can be made on the basis of the information provided by FIS. The equation of the centroid method is

$$X = \frac{\int abx \times \mu(x)\,dx}{A}. \tag{9}$$

The variable X is a point in the interval $[a, b]$, where a is the maximum crisp value and b the minimum crisp value. $\mu(x)$ is the membership function at point x. A is the area under the membership function curve between the interval values a and b. Using the centroid method in the defuzzification process, FIS can determine the center value of the area under the OMF curve. This results in an output point that represents the value of the output variable expected on the basis of the given input. Figure 6 shows the rule viewer in Matlab. FIS helps determine the center value of the OMF. Thus, FIS can help determine the value of each category in the membership accurately and is not affected by bias. To ensure that there is no bias in the defuzzification process, it is important to ensure that the membership function and defuzzification method used correspond to the predetermined value ranges. For example, in Table 2, the "Poor" category of the OMF Environment Quality has a value in the range of 1–50. In the defuzzification process, if the crisp value is classified as "Poor", then the membership level in the "Poor" category will be high, while the membership levels in the "Moderate" and "Good" categories will be low. A similar effect occurs in the "Moderate" category with a value in the range of 50–70: an increase in membership level in the "Moderate" category will cause a decrease in membership level in other categories. Likewise, in the "Good" category with values in the range of 70–100, an increase in membership level in the "Good" category will result in a

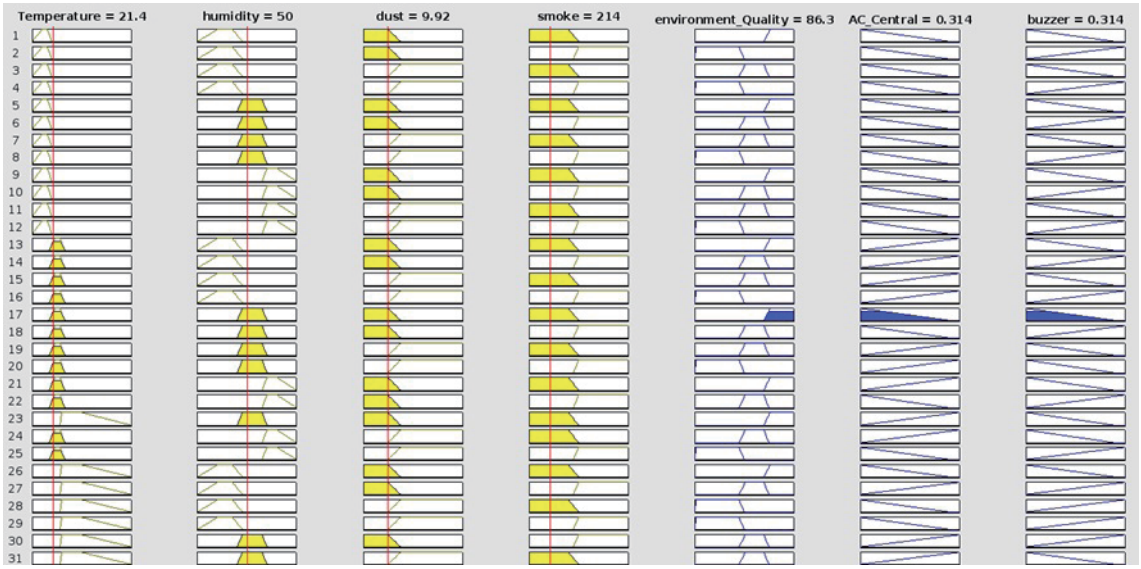


Fig. 6. (Color online) Rule viewer of environment quality system.

decrease in membership level in other categories. Thus, it is important to design membership functions and defuzzification methods carefully in order to minimize potential bias and produce objective results in determining crisp values.

3.3 Hardware system

Figure 7 shows the hardware system integration diagram. It contains BME/BMP280, MQ-2, and GP2Y1010AU0F sensors used to measure temperature, humidity, smoke, and dust in the factory production area. Each sensor is connected to the ESP32 microcontroller board. A microcontroller is a chip in which the central processing unit (CPU), memory, and I/O devices are integrated to control electronic systems or devices that will be programmed into it to run various applications or systems such as that for environmental quality in a factory. The wiring layout for each sensor is as follows: the VCC pins of all sensors are connected to a power source of 3.3 V, and the ground (GND) pin is connected to the GND pin on ESP32. Next, the SDA and SCL pins of the BME/BMP280 sensor must be connected to the GPIO36 and GPIO35 pins. For the MQ-2 sensor, the analog output (AO) is connected to the GPIO05 pin on ESP32. Moreover, for the GP2Y1010AU0F sensor, the analog input is connected to the GPIO08 pin and the output for controlling the LED is connected to the GPIO15 pin. This microcontroller ESP32 is connected to a relay to control the central AC in the factory. Because microcontrollers cannot directly control electrical equipment with high voltage, relays are used. When the indoor environmental quality deteriorates, the buzzer will function as an automatic alarm to warn people if the environment is unhealthy.

3.3.1 ESP32-S

A microcontroller is an Internet of Things (IoT) device that can function as a publisher and subscriber. The publisher will send the data to the server, and the subscriber will look at it and

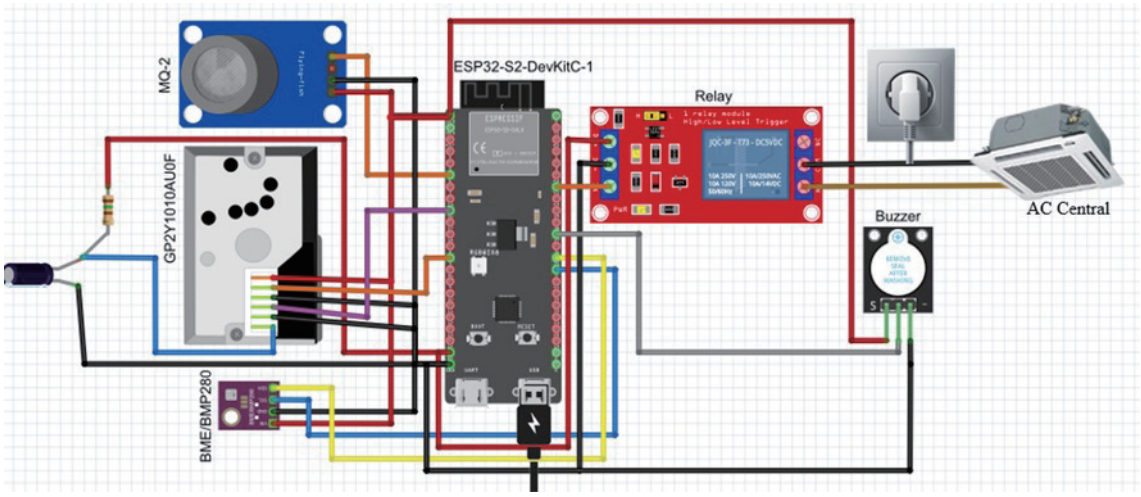


Fig. 7. (Color online) Hardware system integration diagram.

compare the errors and time it takes to transmit and process the entire data received by the microcontroller.[21] In this study, we use ESP32 as the microcontroller as shown in Fig. 8. ESP32 is made by Espressif Systems, a company based in Shanghai, China. TSMC is used as a core producer and is 40 nm in size. ESP32 is an open-source and low-power microcontroller for economic and energy-efficient IoT needs. Equipped with integrated Wi-Fi and dual-mode Bluetooth, there is no need to use additional Wi-Fi or Bluetooth modules. The ESP32 series uses the Tensilica Xtensa LX6 microprocessor as the core, and it is available in both single-core and dual-core modes. ESP32 integrates an antenna switch, an RF balun, a power amplifier, a low-noise receiver amplifier, a filter, and a power management module. It has clock frequencies of 80, 160, and 240 MHz. This microcontroller contains 520 kB of RAM and 4 MB of flash memory for storing data and programs. It is perfect for IoT projects because of its high processing power.

3.3.2 BME/BMP280

Robert Bosch is the world's market leader for pressure sensors in automotive and consumer applications. Bosch provides the BME/BMP280 sensor breakout board featuring temperature, humidity, and barometric pressure sensors. BME/BMP280 is the next-generation sensor from Bosch, an upgraded version of BMP085/BMP180/BMP183 with a low noise height of 0.25 m and the same short transition time. This sensor is tiny, and has low power consumption. BME/BMP280 is built on piezoresistive pressure sensor technology and features high accuracy, linearity, and stability. BME/BMP820 has a temperature sensor rating of ±1.0 hPa and a temperature measurement accuracy of ±1 °C. The sensor dimensions are 2 × 2.5 × 0.95 mm^3. Its operating pressure ranges from 300 to 1100 hPa, while its temperature measurement range extends from −40 to 85 °F. The power voltage required for operation ranges from 1.2 to 3.6 V. BME/BMP280 is shown in Fig. 9.

3.3.3 MQ-2

MQ-2 is a sensor module that can be used to detect flammable smoke or gases, such as liquefied petroleum gas (LPG), hydrogen (H_2), methane (CH_4), carbon monoxide (CO), alcohol,

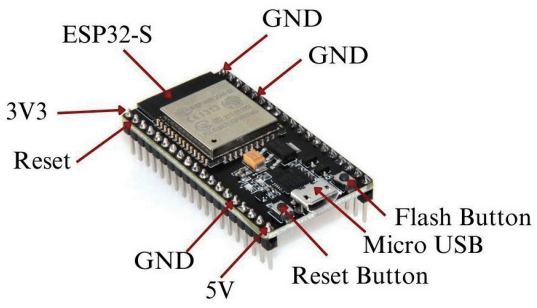


Fig. 8. (Color online) ESP32-S.

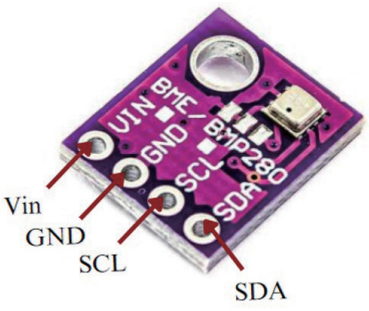


Fig. 9. (Color online) BME/BMP280.

smoke, and propane at concentrations between 200 and 10000 ppm. Moreover, MQ-2 can also be used as a tool for environment quality monitoring. MQ-2 is shown in Fig. 10.

3.3.4 GP2Y1014AU0F

GP2Y1014AU0F is an optical environment quality sensor designed to sense dust particles in air. The internal diagonal installation of infrared light-emitting diodes and a photoelectric crystal allows it to detect the light reflected off dust in air. Even very small particles such as tobacco smoke particles can be detected. It is usually used in air purification systems. The sensor has very low current consumption (20 mA max; 11 mA typical) and can use up to 7 VDC. GP2Y1014AU0F is shown in Fig. 11.

3.3.5 Relay

Figure 12 shows a relay module. Relay modules are the same as relays in general. There is a microcontroller board that allows us to control the relay module using a microcontroller, similarly to the way sensor node researchers use ESP32. Researchers use the relay module

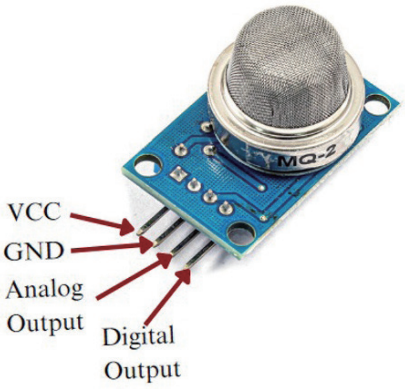


Fig. 10. (Color online) MQ-2.

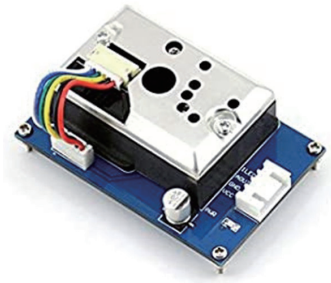


Fig. 11. (Color online) GP2Y1014AU0F.

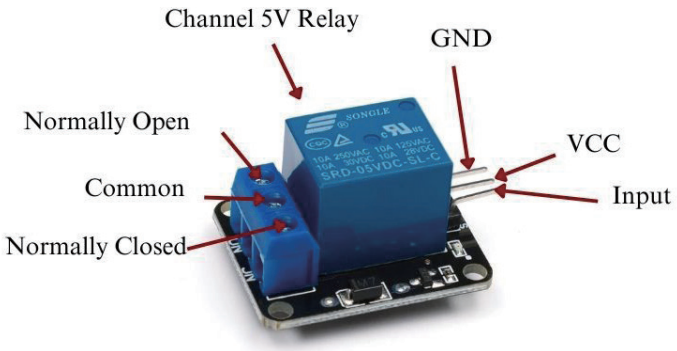


Fig. 12. (Color online) Relay.

because it can regulate a high-voltage electronic circuit using low-voltage signals and protect motors or other components from over-voltage or short circuits.

3.3.6 Central AC

We use central AC in our research experiment. The average industrial plant uses central AC to cool the workspace. Central AC is a system in which air is cooled at a central location and distributed to and from rooms using one or more fans and ductwork. Figure 13 shows a central AC unit.

3.3.7 Buzzer

The buzzer is an electronic component that can convert electrical voltage into sound that acts as a signal on the sensor. The most common uses of buzzers are as alarms, sound indicators, and timers. Figure 14 shows a buzzer.

3.4 Software

3.4.1 Firebase database

Firebase has come a long way since joining Google in 2014. Firebase is undoubtedly one of the most popular backend-as-a-service (BaaS) platforms for app developers. Firebase is a cloud app development platform that supports Android, iOS, and web pages. The purpose of this system is to store a database of information for applications that require control and data. Information from messages can be stored in the local system database. This helps application developers to rapidly build back-end services in the cloud, provides a real-time database, shortens application development time, and helps developers focus more on front-end optimization. Figure 15 depicts the workflow of the Firebase database in this study.

Fig. 13. (Color online) Central AC unit.

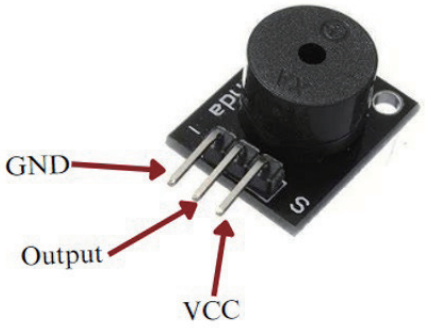


Fig. 14. (Color online) Buzzer.

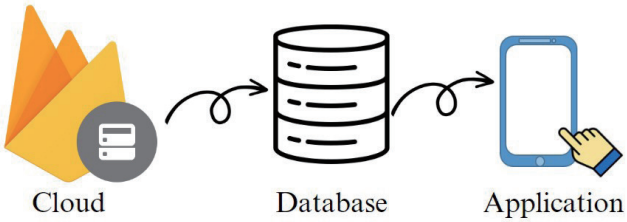


Fig. 15. (Color online) Firebase database.

3.4.2 Android Studio

Android Studio is an integrated development environment (IDE) designed for creating applications on the Android platform. It was released by Google I/O on May 16, 2013, and is freely available for developers to use. Android Studio is used to develop applications for Android smartphones because of its specialized programming capabilities. The intelligent code editor feature in Android Studio enables an easier code analysis, which helps researchers write better, faster, and more productive codes. Android Studio has strong support for integration with Firebase. Using the Firebase SDK, mobile apps can connect with Firebase services to access stored environment quality data, send data to a Firebase database, or receive real-time data from sensors in a factory. Fig. 16 illustrates the workflow of the android studio in this study.

3.4.3 Thonny

Thonny is a software or IDE for Python designed for beginners. It was created by Aivar Annanamaa, an Estonian programmer. Thonny is often recommended for beginners who want to learn or develop skills in Python programming. Thonny supports multiple operating systems, including Windows, mac OS, and Linux. It also supports multiple versions of Python. In this study, it is used as an integrated tool to develop and test the FIS -DRL model for environment quality control in intelligent factories, as shown in Fig. 17.

3.5 Model configuration

As explained in the previous section, we will combine the FIS and DRL functions. The first step is to obtain accurate readings of the temperature, humidity, smoke, and dust sensors in real time as input values. Subsequently, the microcontroller reads and manages the sensor accuracy by conducting tests using the fuzzy method to make decisions on the basis of the sensor accuracy and produce output rules based on predetermined fuzzy rules. The output of the system at this time is used to determine environment quality, control the central AC, and activate the buzzer as an alarm when an error occurs in the system. After creating a rule in FIS Mamdani, it is necessary to create a model system using DRL. The factory workroom is the environment that will be tested in the context of central AC. Next, the state s of the environment is determined in

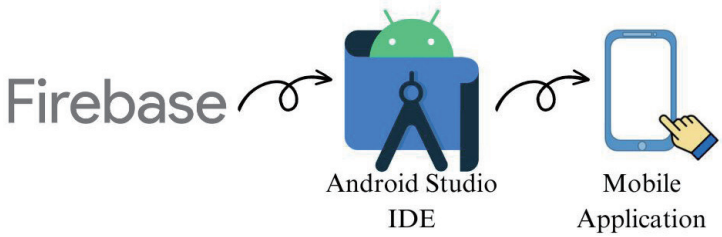


Fig. 16. (Color online) Android Studio.

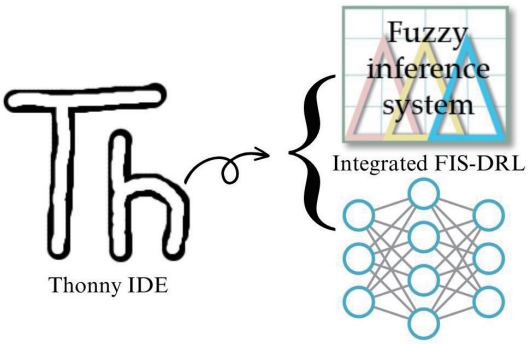


Fig. 17. (Color online) Thonny IDE.

the form of a vector describing environmental conditions in terms of temperature t, humidity h, smoke sm, and particles of dust d. After determining the state values, the next step is to determine action a, which is the determination of the environment quality in the factory environment and the speed of the central AC. Reward r is positive or negative feedback on the agent's decision and becomes a consideration for the agent to determine action a. The DRL model used is a DNN that supports learning and uses the RL method to carry out the training process. The model will interact with the environment and take action based on real-time conditions, receive rewards, and update decision-making strategies in accordance with the events and feedback received.

In the previous section, we discussed the OMF category. The next step involves DNN modeling with multiple class conditions. The dataset in Table 3 represents a class in DNN. The environment quality values are categorized into three conditions: poor, moderate, and good. In the learning model, they are symbolized as 1, 2, and 3, respectively. For the central AC value, turn-on is symbolized as 1 and turn-off as 0. The buzzer value follows the same convention.

During the modeling stage, DRL requires a dataset with optimal results, where, to find the optimal value, the hyperparameter tuning of the DNN architecture layer is carried out. Hyperparameters are tuned using a training dataset of 7200 samples and a testing dataset of 1440 samples. Our study involves a scenario-based approach with 10 different variations of the model, considering factors such as the number of neurons in the three hidden layers, the number of epochs, and the learning rate. Table 4 presents the results of comparing 10 DNN models. The number of neurons or density in the hidden layer is random. Finding the optimal value for the

Table 3
Class dataset in DNN

OMF	Class	Binary class
Environment quality	Poor	1
	Moderate	2
	Good	3
Central AC	On	1
	Off	0
Buzzer	On	1
	Off	0

Table 4
Results for 10 DNN models.

Model name	Hidden layer (neurons)			Learning rate	Epoch
	H1	H2	H3		
Model 1	256	256	256	0.0005	200
Model 2	128	128	128	0.0005	200
Model 3	128	64	32	0.0005	200
Model 4	32	64	128	0.0005	200
Model 5	64	64	64	0.0005	200
Model 6	32	32	32	0.0005	200
Model 7	32	64	128	0.005	300
Model 8	32	64	128	0.0005	300
Model 9	256	128	64	0.005	200
Model 10	128	128	128	0.005	200

number of neurons in the hidden layer is very important for the model to learn the data well. There are several categories, where each category has a certain maximum value. For example, in model 1, all neurons have a maximum value of 256. Model 4 has a combination of the number of neurons from minimum to maximum values, namely, 32, 64, and 128. Moreover, in model 6, the number of neurons in each hidden layer is 32. Several other models have variations in the number of neurons in the hidden layer. From among the models with these various hidden layer variations, we will determine the model that provides optimal results. This hyperparameter tuning process affects the f1 score, how low the loss value is, how much memory is consumed, and the accuracy of the results.

3.6 Experiment setup

Figure 18 shows a work system flowchart for our experiment. The initial stage involves integrating all the components, including temperature, humidity, smoke, and dust sensors, within the factory conditions. Subsequently, FIS is employed to develop fuzzy rules based on the received sensor values. Next, an implementing model is developed, consisting of 8640 datasets divided into a training dataset (83%) and a test dataset (17%). The learning model processing stage involves the analysis of the implemented model's performance. DNN processing is utilized to compare several models and select one with the highest accuracy. If the accuracy falls below 90%, the process returns to the learning model processing stage. If the accuracy reaches or exceeds 90%, the model is integrated into the DRL stage. The FIS and DRL methods are

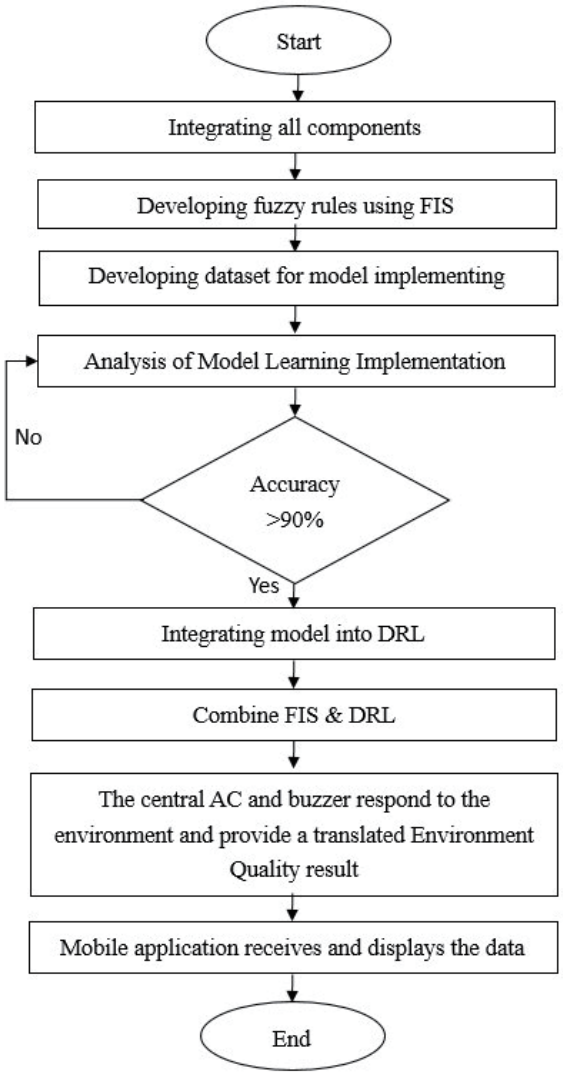


Fig. 18. Flowchart of experimental process.

combined to generate output in the factory environment. The central AC and the buzzer operate in accordance with the actions determined by the optimal agent. Additionally, the system translates the environment quality conditions into categories such as poor, moderate, and good. These categories are displayed through the mobile application.

4. Results and Discussion

4.1 Analysis of fuzzy rule result

In the previous section, the input values were determined on the basis of the factory conditions, and the output rule values were obtained using FIS. To avoid bias in each parameter result, the use of the FIS method can still provide benefits in handling uncertainty and vagueness

in environmental data. Thus, FIS can help determine the value of each category in the membership accurately and unaffected by bias. In this study, there are three OMFs, namely, environment quality, AC control, and buzzer. These three outputs will later become actions in DRL modeling dependent on the state conditions in the environment. This stage represents the implementation of fuzzy inference utilizing the Mamdani method to apply fuzzy rules. The fuzzy rule set consists of 36 rules comprising four IMFs and three OMFs, as shown in Table 5.

4.2 Analysis of model learning implementation

The action determined by DRL can be carried out by agents to determine the best conditions of environment air, AC, and buzzer so that the whole is integrated into one complete action. The DRL model is based on 36 rules generated at the FIS stage and triggered by the state conditions, namely, temperature, humidity, dust, and smoke. Within the agent, conditions related to environmental air, central AC, and the buzzer can be determined to integrate them into a unified action. In Table 6, the results of the trained model architecture evaluation using DNN to control environment quality are presented. The table shows the values of various performance metrics for each model, including accuracy, loss, memory usage, and inferencing time. The accuracy of each model reflects how well the model can produce outputs that match the expected targets under various conditions of, for example, environment quality, central AC, and buzzer. A lower loss percentage indicates higher performance in predicting environment quality control. Moreover, each model uses different levels of memory. Less memory usage implies a more efficient and less resource-intensive operation of the model. Inferencing time, measured in microseconds, indicates how rapidly each model can make decisions, and is a metric to consider in making more responsive and efficient control systems.

Table 5
Membership function rules.

Rule number	IMF				OMF		
	Temperature (°C)	Humidity (%)	Dust (μg/Nm^3)	Smoke (μg/m^3)	Environment quality	Ac control	Buzzer
1	Cool	Normal	Risk	Risk	Poor	Off	On
2	Cool	Humid	Normal	Normal	Moderate	Off	Off
3	Cool	Humid	Risk	Risk	Poor	Off	On
4	Normal	Dry	Normal	Normal	Good	On	Off
5	Normal	Dry	Risk	Risk	Moderate	On	On
6	Normal	Normal	Normal	Normal	Good	Off	Off
7	Normal	Normal	Risk	Risk	Moderate	Off	On
8	Normal	Humid	Normal	Normal	Good	Off	Off
9	Normal	Humid	Risk	Risk	Moderate	Off	On
10	Warm	Dry	Normal	Normal	Good	On	Off
11	Warm	Dry	Risk	Risk	Poor	On	On
12	Warm	Normal	Normal	Normal	Good	On	Off
13	Warm	Normal	Risk	Risk	Poor	On	On
				…			
36	Warm	Humid	Risk	Risk	Poor	On	On

Table 6
Modeling results of 10 DNN models.

Model name	Accuracy (%)			Loss (%)	Memory use (kB)	Inferencing time (ms)
	Environment quality	Central AC	Buzzer			
Model 1	83.1	91.9	98.6	3.55	146.5	33
Model 2	83.3	91.9	98.7	1.56	48.1	8
Model 3	81.5	92.1	98.8	1.85	25.2	3
Model 4	82.7	92.4	98.4	1.64	25.1	3
Model 5	82.3	88.1	98.5	1.35	22.9	3
Model 6	81.3	89.2	98.2	2.50	16.3	1
Model 7	80.3	92.2	98.9	1.66	25.0	3
Model 8	84.0	92.2	96.6	2.81	25.0	4
Model 9	77.8	91.9	98.6	1.24	56.6	11
Model 10	82.2	92.4	98.9	0.96	48.0	8

From Table 6, we can observe that both models 4 and 10 show relatively high accuracies above 90% and low loss rates below 1.70%. However, model 4 outperforms model 10 in terms of memory usage and inference time. On the other hand, model 5 exhibits the shortest inference time of 3 ms and an efficient memory usage of 22.9 kB. Upon further observation, although model 5 has lower loss and memory usage, its average accuracy is much lower than that of model 4. Specifically, model 4 is the best choice in this study, achieving an average accuracy of 91.16%, while model 5 only attains an average accuracy of 89.63%. Therefore, model 4 is highly suitable for controlling environment quality in intelligent factories. The differences in the top three model levels are shown in Table 7.

In data visualization using model 4, there are three outputs: environment quality, central AC, and buzzer. In the data visualization for environment quality modeling, three different shades of green are used to represent correct conditions, while three different shades of red represent incorrect conditions in the modeling process. The errors in the environment quality conditions are more prominent because the accuracy in modeling environment quality only reaches 82.7%, as can be seen in Fig. 19.

The visualization of the data from central AC modeling can be seen in Fig. 20. We can observe that data with a value of 0 are depicted in yellow, indicating the condition of central AC as turn-off, while data with a value of 1 are shown in green, indicating the condition of central AC as turn-on. The red and orange colors indicate inconsistencies or errors in the modeling process.

The data visualization for the buzzer is shown in Fig. 21 and follows the same pattern as that for central AC. Data with a value of 0, representing the turn-off condition, are indicated in yellow, while data with a value of 1, indicating the turn-on condition, are indicated in green. However, similar to the central AC data visualization, there are also red and orange colors, which indicate inconsistencies or errors in the modeling process.

Table 7
The top three ranking models.

Model name	Average accuracy (%)	Loss (%)	Memory use (kB)	Inferencing time (ms)
Model 4	91.17	1.64	25.1	3
Model 5	89.63	1.35	22.9	3
Model 10	91.17	0.96	48	8

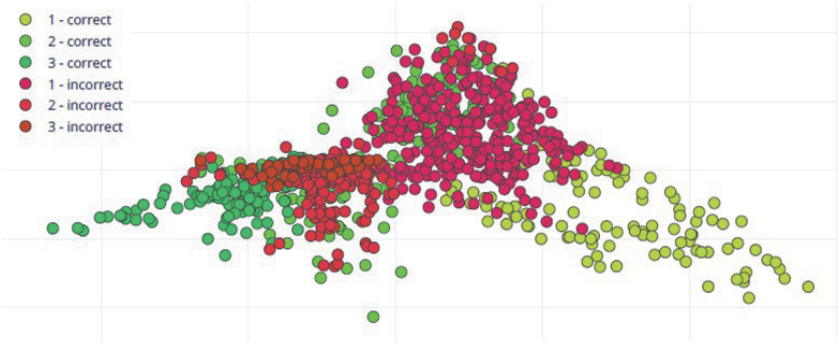


Fig. 19. (Color online) Data visualization in modelling of environment quality.

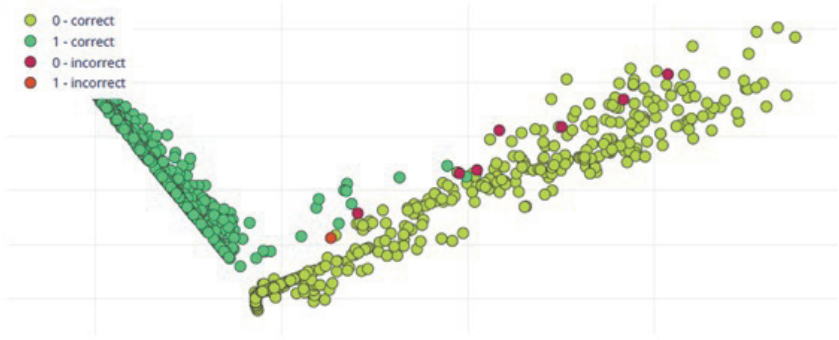


Fig. 20. (Color online) Data visualization in modelling of central AC.

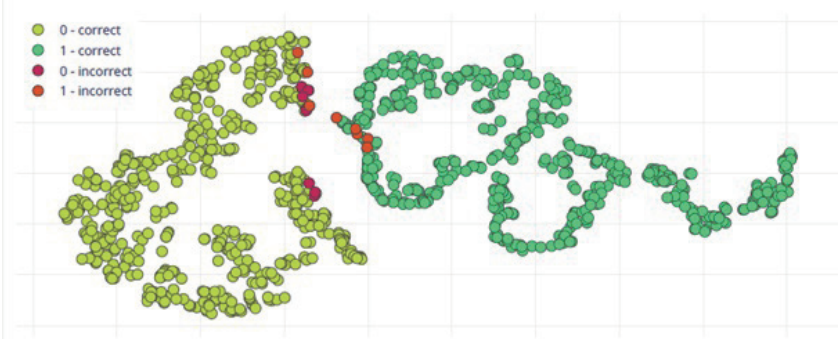


Fig. 21. (Color online) Data visualization in modelling of buzzer.

4.3 Comparison of FIS-DRL and conventional system

After modeling the FIS-DRL system with model 4, the next step is to test the system to evaluate the performance of the developed model. Thirty scenarios are tested to compare the FIS-DRL system with conventional systems for environment quality assessment, as shown in Table 8. The test dataset consists of 13 records indicating a good environment quality situation, 14 records representing a moderate environment quality situation, and three records reflecting a bad environment quality situation. The FIS-DRL modeling we conducted resulted in an accuracy of 91%. These records were collected randomly under various room conditions of temperature, humidity, dust, and smoke.

Figure 22 shows the reward results for environment quality, AC settings, buzzer, and mean reward. From these reward results, it can be concluded that the agent is starting to learn the environment that it is supposed to control. In Fig. 22(a), we can see that although the agent learns

Table 8
Testing scenarios of FIS-DRL implemented for environment quality.

Number	Parameter				FIS-DRL system	Expected value
	Temperature (°C)	Humidity (%)	Dust (μg/Nm^3)	Smoke (μg/m^3)		
1	24	67	10	467	Good	Good
2	26	38	35	396	Good	Good
3	25	68	9	286	Good	Good
4	23	45	38	371	Good	Good
5	22	49	17	7618	Good	Moderate
6	25	87	1	23	Good	Good
7	28	34	37	94	Good	Good
8	28	76	7	58	Good	Good
9	22	38	25	105	Good	Good
10	24	0	31	266	Good	Good
11	25	45	38	105	Good	Good
12	22	14	23	193	Good	Good
13	28	33	40	396	Good	Good
14	27	82	5	81	Moderate	Moderate
15	28	50	3	88	Moderate	Moderate
16	34	59	5	319	Moderate	Moderate
17	27	75	2	358	Moderate	Moderate
18	28	56	5	9106	Moderate	Poor
19	41	55	7	821	Moderate	Moderate
20	28	58	4	1607	Moderate	Moderate
21	38	65	7	1695	Moderate	Moderate
22	24	47	25	965	Moderate	Poor
23	24	73	2	357	Moderate	Moderate
24	27	49	0	2154	Moderate	Moderate
25	33	60	0	4311	Moderate	Moderate
26	28	54	10	679	Moderate	Moderate
27	30	59	6	8819	Moderate	Moderate
28	27	60	34	679	Poor	Poor
29	26	62	30	5999	Poor	Poor
30	22	61	12	2228	Poor	Poor

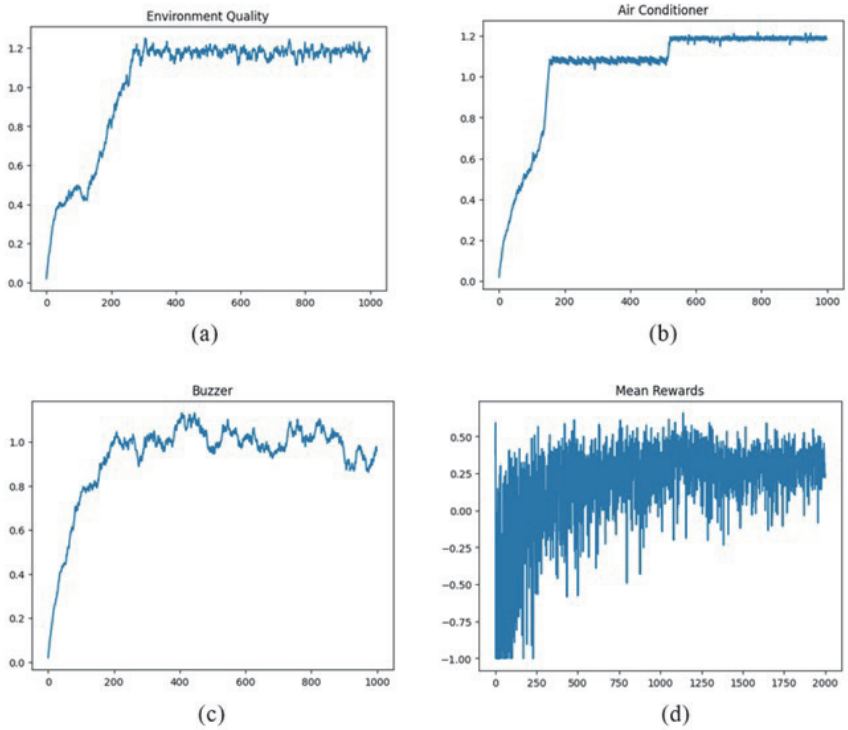
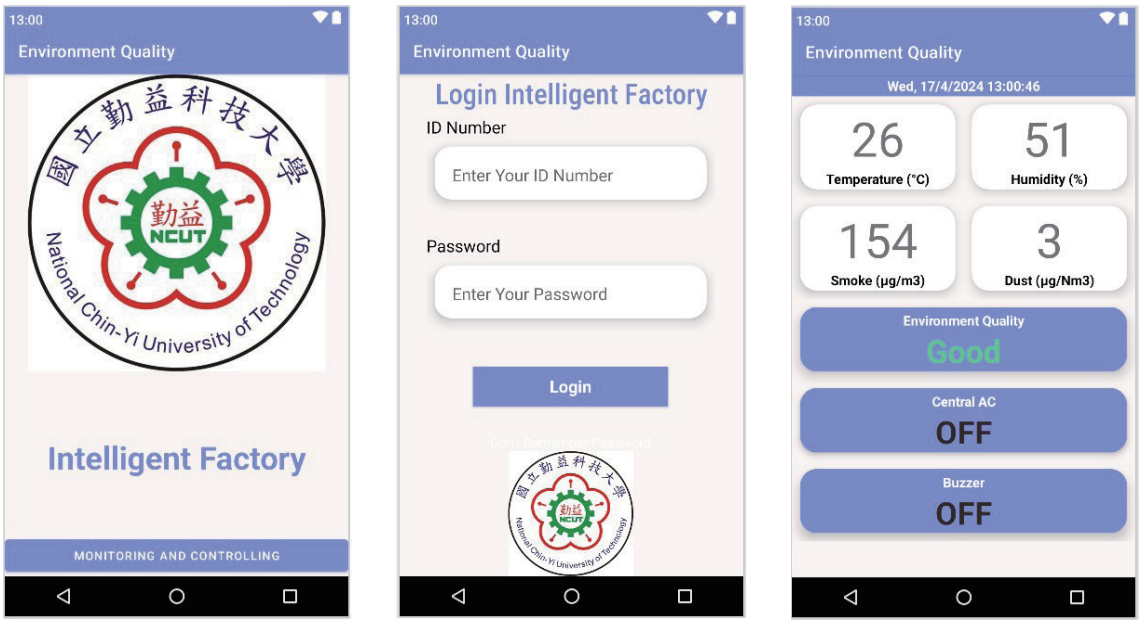


Fig. 22. (Color online) Rewards of environment quality in factory. (a) Environment quality. (b) AC. (c) Buzzer. (d) Mean reward.

rapidly to recognize and respond to the buzzer, the final reward obtained is not higher than those of other devices. This suggests that, although the agent learns rapidly, its ability to understand and respond to emergency conditions may not be higher than those of other devices. On the other hand, for environment quality, as seen in Fig. 22(b), although the agent learns more slowly, the final reward obtained is higher. This suggests that the agent may need more time to understand the environment quality conditions in the environment, but is then able to take more effective actions based on that understanding. In Fig. 22(c), for the AC setting, the agent learns gradually, where in some episodes, it may stagnate in its learning. However, in more advanced episodes, the agent starts to learn better and the reward increases from that of the previous episode. This shows that the agent can learn from experience and improve its performance over time. Regarding the mean reward seen in Fig. 22(d), although there is a decrease when the buzzer has a learning decline, the system as a whole can still run well. This shows that, despite variations in the individual performance of the system components, the overall system can still achieve good performance.

Furthermore, an experiment is conducted to detect smoke by deliberately lighting a fire in the room for a certain duration. The aim of this experiment is to assess the system's capability to detect and respond to smoke in real-world scenarios.

The mobile application is developed using Android Studio IDE. Figure 23(a) illustrates the main menu design, while Fig. 23(b) shows the login menu design. To ensure system security and protect factory privacy, the application design requires users to enter an ID number and password



(a) (b) (c)

Fig. 23. (Color online) Mobile device application. (a) Main menu. (b) Login menu. (c) Monitoring and controlling menu.

to log in, as depicted in Fig. 23(b). The subsequent menu shown in Fig. 23(c) allows users to monitor various factory conditions, including temperature, humidity, smoke, dust, environment quality, central AC status, and an alarm represented by a buzzer.

4.4 Cost analysis of FIS-DRL in intelligent factory system

The implementation of an intelligent factory control system that utilizes a combination of FIS-DRL technology has a significant impact on factory workers. It offers efficiency and ease in managing intelligent factories at a low cost. An analysis was conducted to compare the intelligent factory control system based on FIS-DRL technology with the utilization of human labor. This analysis was performed under typical conditions in Taiwan, where the electricity cost is NTD 7.03 per kWh. The FIS-DRL device has a maximum power usage of 20 W and operates continuously for 24 h a day. On the other hand, the cost of human labor is NTD 1000 per day, with a working duration ranging from 3 to 5 h. In Taiwan, a maintenance fee of NTD 1500 per month is charged to ensure that the system is running properly and to detect any errors or issues that may arise during its operation. The maintenance fee covers regular checks and necessary repairs to maintain the system's functionality and efficiency.

The FIS-DRL and labor costs are shown in Table 9. It is evident that the FIS-DRL method incurs lower daily and monthly costs (NTD 6561.6) than labor costs (NTD 30000). When converted to USD, the FIS-DRL method costs USD 209.69, while the labor cost is USD 958.70. Therefore, implementing the FIS-DRL method appears to be more cost-effective and economical than the labor-intensive approach in this particular scenario.

Table 9
Analysis of FIS-DRL in intelligent factory system and labor costs.

Method	Component	Daily cost (NTD)	Monthly cost (30 days; NTD)	Maintenance cost (NTD)	Total (NTD)	in USD (1 USD = 31.29 NTD)
FIS-DRL	20 W; NTD 7.03 per kWh	168.720	5061.6	1500	6561.6	209.69
Labor cost	Labor cost = NTD 1000	1000	30000	—	30000	958.70

5. Conclusions

In this study, the combination of FIS and DRL using DNN modeling in model 4 was proven to be highly suitable for implementing environment quality control or monitoring in intelligent factories. The model achieved satisfactory accuracies for each output, namely, 82.7% for environment quality, 92.4% for central AC, and an impressive 98.8% for the buzzer. Additionally, the memory usage for this model was low, specifically 25.1 kB. This implementation successfully demonstrated the potential to optimize resource usage, reduce labor costs, and enhance efficiency in controlling environment quality in factory environments.

FIS was utilized to model and handle the uncertainty of factory environment data by employing sets and rules to generate output based on the given data. FIS used fuzzy sets and created 33 fuzzy rules to provide output based on the existing IMF.

Through the integration of FIS and DRL, we obtained promising results in addressing the complex challenges of managing environment quality in intelligent factories. The resulting system improved efficiency, reduced operational costs, and enhanced safety and comfort levels in the factory environment. The results of this study significantly contribute to the development of environment quality control and monitoring systems in intelligent factories, with potential positive impacts across various industrial fields and work environments.

The obtained results serve as a basis for further development in utilizing FIS and DRL technology to enhance overall system efficiency and performance, which will benefit numerous industries and work settings.

Acknowledgments

The authors would like to thank the Department of Electronic Engineering of National Chin-Yi University of Technology for supporting this research.

References

1 Y. Wu, H. N. Dai, H. Wang, Z. Xiong, and S. Guo: IEEE Commun. Surv. Tutorials **24** (2022) 2. https://doi.org/10.1109/COMST.2022.3158270
2 M. N. Assimakopoulos, A. Dounis, A. Spanou, and M. Santamouris: Sci. Total Environ. **449** (2013) 461. https://doi.org/10.1016/j.scitotenv.2012.12.043
3 A. Nur Afifah and K. Candraning Diyanah: Indones. J. Public Heal. **16** (2021) 3. https://doi.org/10.20473/ijph.vl16il.2021.386-396
4 J. J. Carbajal-Hernández, L. P. Sánchez-Fernández, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad: Atmos. Environ. **60** (2012). https://doi.org/10.1016/j.atmosenv.2012.06.004

5 R. Bao, Y. Zhou, and W. Jiang: Proc. 2022 2nd Int. Electr. Eng. Control Sci. Conf. (IC2ECS, 2022) 986–989. https://doi.org/10.1109/IC2ECS57645.2022.10088050.
6 Y. Li, X. Hu, Y. Zhuang, Z. Gao, P. Zhang, and N. El-Sheimy: IEEE Internet Things J. **7** (2020) 7. https://doi.org/10.1109/JIOT.2019.2957778
7 S. K. Heo, K. J. Nam, J. Loy-Benitez, Q. Li, S. C. Lee, and C. K. Yoo: Energy Build. **202** (2019) 109440. https://doi.org/10.1016/j.enbuild.2019.109440
8 Z. Hu, S. Cong, T. Song, K. Bian, and L. Song: IEEE Internet Things J. **7** (2022) 9. https://doi.org/10.1109/JIOT.2020.3004339
9 W. Valladares, M. Galindo, J. Gutiérrez, W. C. Wu, K. K. Liao, J. C. Liao, K. C. Lu, and C. C. Wang: Build. Environ. **155** (2019) 105. https://doi.org/10.1016/j.buildenv.2019.03.038
10 A. Haydari, M. Zhang, C. N. Chuah, and D. Ghosal: Proc. 2021 IEEE Veh. Technol. Conf. (IEEE, 2021) 3–4. https://doi.org/10.1109/VTC2021-Spring51267.2021.9448639.
11 Z. Shahbazi, Y. C. Byun, and H. Y. Kwak: Processes **9** (2021) 9. https://doi.org/10.3390/pr9091593
12 N. Kumar, S. S. Rahman, and N. Dhakad: IEEE Trans. Intell. Transp. Syst. **22** (2021) 8. https://doi.org/10.1109/TITS.2020.2984033
13 M. Muñoz and E. Miranda: Proc. 2016 IEEE Fuzzy System Conf. (FUZZ-IEEE, 2016) 888. https://doi.org/10.1109/FUZZ-IEEE.2016.7737782
14 A. Pamuji: Sci. J. Informatics **4** (2017) 1. https://doi.org/10.15294/sji.v4i1.7082
15 N. Alavi: J. Saudi Soc. Agric. Sci. **12** (2012) 2. https://doi.org/10.1016/j.jssas.2012.10.001
16 M. Sewak, S. K. Sahay, and H. Rathore: Proc. 2022 IEEE 5th Int. Conf. Comput. Commun. Autom. (ICCCA, 2020) 379. https://doi.org/10.1109/ICCCA49541.2020.9250787
17 D. P. F. Möller: Adv. Inf. Secur. **103** (2023) 347. https://doi.org/10.1007/978-3-031-26845-8_8
18 S. Dong, P. Wang, and K. Abbas: Comput. Sci. Rev. **40** (2021) 100379. https://doi.org/10.1016/j.cosrev.2021.100379
19 M. Kaloev and G. Krastev: Proc. 2021 5th Int. Symp. Multidiscip. Stud. Innov. Technol. (ISMSIT, 2021) 351–352. https://doi.org/10.1109/ISMSIT52890.2021.9604690
20 C. Li, P. Zheng, Y. Yin, B. Wang, and L. Wang: CIRP J. Manuf. Sci. Technol. **40** (2023) 75. https://doi.org/10.1016/j.cirpj.2022.11.003
21 R. Niranjana, S. Arvind, M. Vignesh, and S. Vishaal: Proc. 2022 Int. Conf. Autom. Comput. Renew. Syst. (ICACRS, 2022) 1. https://doi.org/10.1109/ICACRS55517.2022.10028992

About the Authors

Wen-Tsai Sung is working with the Department of Electrical Engineering, National Chin-Yi University of Technology as a professor and Dean of Research and Development. He received his MS and PhD degrees from the Department of Electrical Engineering, National Central University, Taiwan in 2000 and 2007, respectively. He has won the 2009 JMBE Best Annual Excellent Paper Award and the Dragon Thesis Award sponsored by Acer Foundation. His research interests include artificial intelligence Internet of Things (AIoT) and wireless sensors networks. (songchen@ncut.edu.tw)

Jenny Aryani obtained her bachelor's degree in electrical engineering from Taiwan National Chin-Yi University of Technology and continued on to a master's course at the same university in electrical engineering. Her research interests are in artificial intelligence Internet of Things related applications and sensor applications. (4b112132@gm.student.ncut.edu.tw)

Sung-Jung Hsiao is working with the Department of Information Technology, Takming University of Science and Technology, as an associate professor. He received his BS degree in electrical engineering from the National Taipei University of Technology, Taiwan, in 1996 and MS degree in computer science and information engineering from National Central University, Taiwan, in 2003. He received his Ph.D. degree from the Department of Electrical Engineering, National Taipei University of Technology, Taiwan, in 2014. He has work experience in research and design at the renowned computer company, Acer Universal Computer Co., Mitsubishi, and FIC.
(sungjung@gs.takming.edu.tw)