International Journal of Internet, Broadcasting and Communication Vol.15 No.2 144-156 (2023)
http://dx.doi.org/10.7236/IJIBC.2023.15.2.144

IJIBC 23-2-16

A Study on the Development of Artificial Intelligence Crop Environment Control Framework

Guangzhi Zhao

Ph.D. Candidate., Dept. of Computer Science and Engineering, Jeonbuk National University
frankzgz1234@gmail.com

Abstract

Smart agriculture is a rapidly growing field that seeks to optimize crop yields and reduce risk through the use of advanced technology. A key challenge in this field is the need to create a comprehensive smart farm system that can effectively monitor and control the growth environment of crops, particularly when cultivating new varieties. This is where fuzzy theory comes in, enabling the collection and analysis of external environmental factors to generate a rule-based system that considers the specific needs of each crop variety. By doing so, the system can easily set the optimal growth environment, reducing trial and error and the user's risk burden. This is in contrast to existing systems where parameters need to be changed for each breed and various factors considered. Additionally, the type of house used affects the environmental control factors for crops, making it necessary to adapt the system accordingly. While developing such a framework requires a significant investment of labour and time, the benefits are numerous and can lead to increased productivity and profitability in the field of smart agriculture. We developed an AI platform for optimal control of facility houses by integrating data from mushroom crops and environmental factors, and analysing the correlation between optimal control conditions and yield. Our experiments demonstrated significant performance improvement compared to the existing system.

Keywords: *Throat Mushroom, IoT (Internet of Things), Fuzzy Theory, Control Environment*

1. Introduction

Although many smart farm technologies have been developed and implemented in South Korea, they currently only support general functions such as opening and closing devices for houses, as well as monitoring and adjusting temperature and humidity [1]. They do not provide detailed technical elements for specific crops. Therefore, these technologies may provide convenience for existing farmers, but they may not have the immediate ability to respond to crop problems for new or inexperienced farmers with limited experience in agriculture.

To overcome these limitations, a smart AI farm that utilizes AI technology to develop an intelligent precision farming support system is needed [2]. This system aims to improve productivity and quality while minimizing

Manuscript Received: March. 18, 2023 / Revised: March. 22, 2023 / Accepted: March. 26, 2023
Corresponding Author: frankzgz1234@gmail.com
Tel:*** - **** - ****
Ph.D. Candidate, Dept. of Computer Science and Engineering, Jeonbuk National University, Korea

the input of labor, energy, and nutrients.

First-generation models that have been widely distributed in South Korea typically only support functions such as opening and closing devices for houses, as well as monitoring and adjusting temperature and humidity. However, they lack the practicality and optimal control technology of third-generation models used in advanced countries for complex energy management.

Existing smart farms only consider simple environmental factors for facility control systems, and there is a need for a smart AI farm that incorporates AI technology to adapt appropriately to environmental factors [3]. The previous systems are simple smart farm models, and automation has not been achieved to a significant extent, leading to a lack of information for building these systems.

Fuzzy theory can be used to generate rule-based inferences and quantify uncertain external environmental factors during data collection and analysis [4].

The system proposed in this study is a comprehensive smart farm system that considers the optimal growth environment setting values using AI techniques and big data analysis [5, 6]. This system takes into account the type of crops and their growth environment settings, allowing for easy adjustment of the cultivation environment for new varieties. This reduces the risk of user errors in cultivation environment settings and decreases the user's burden of risk. The existing system does not consider the variety of crops or their cultivation environment, whereas the proposed system requires parameter value changes depending on the type of crop and considers multiple factors such as the size of the house, control devices, and equipment type [7].

Section 2 provides an explanation of the IoT-based facility, while Section 3 describes the development of the crop control environment framework [8]. Section 4 presents the experimental results, and Section 5 concludes the study [9].

2. IoT-based Platform Environment

In this study, we aim to develop an IoT-based artificial intelligence software platform for optimal control of greenhouse facilities. First, we measure and collect data on mushroom crops grown in greenhouse facilities in the Jinan region of Jeollabuk-do Province in order to analyze the correlation between optimal control conditions and crop yield, as well as environmental conditions and crop growth data [10]. Based on this analysis, we will build an artificial intelligence software platform for optimal control of greenhouse facilities that is more accurate.

The following is an environmental system for building a facility greenhouse control device and collecting crop growth environment data.

Figure 1 shows house control device wind direction, wind speed, precipitation

Figure 1. House control device wind direction, wind speed, precipitation

Figure 2 shows external weather sensor: temperature, humidity, pressure measurement

Figure 2. External weather sensor: temperature, humidity, pressure measurement

Figure 3 shows internal sensors: temperature, humidity measurement

Figure 3. Internal sensors: temperature, humidity measurement

Figure 4 shows Observation camera: capable of pan, tilt, zoom, and night vision.

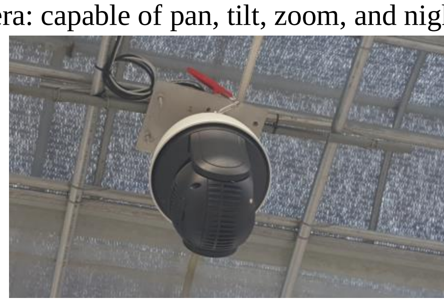


Figure 4. Observation camera: capable of pan, tilt, zoom, and night vision.

The following describes the details of house monitoring, collection, and control:

① Data Collection

i. Collect and monitor indoor temperature and humidity data for each house
ii. Collect and monitor external environmental data for each farm
iii. Collect and monitor control records for each house

② Temperature Control: Remotely control indoor temperature through the analysis of crop-specific optimal control goals in the data analysis system.

Data and control monitoring details are shown in Figure 5.

Figure 5. Data and control monitoring

3. Control Environment Framework

3.1 Data Collection and Analysis

For the environment of research and data collection, which is divided into three categories, the first step is to develop and build an environment control device and environmental sensor for mushroom cultivation facilities to collect data. Secondly, data will be collected through facility environment control devices and environmental sensors, and finally, a system will be developed to collect optimal environmental data for the cultivation of mushroom. Development process of control framework is shown in Figure 6.

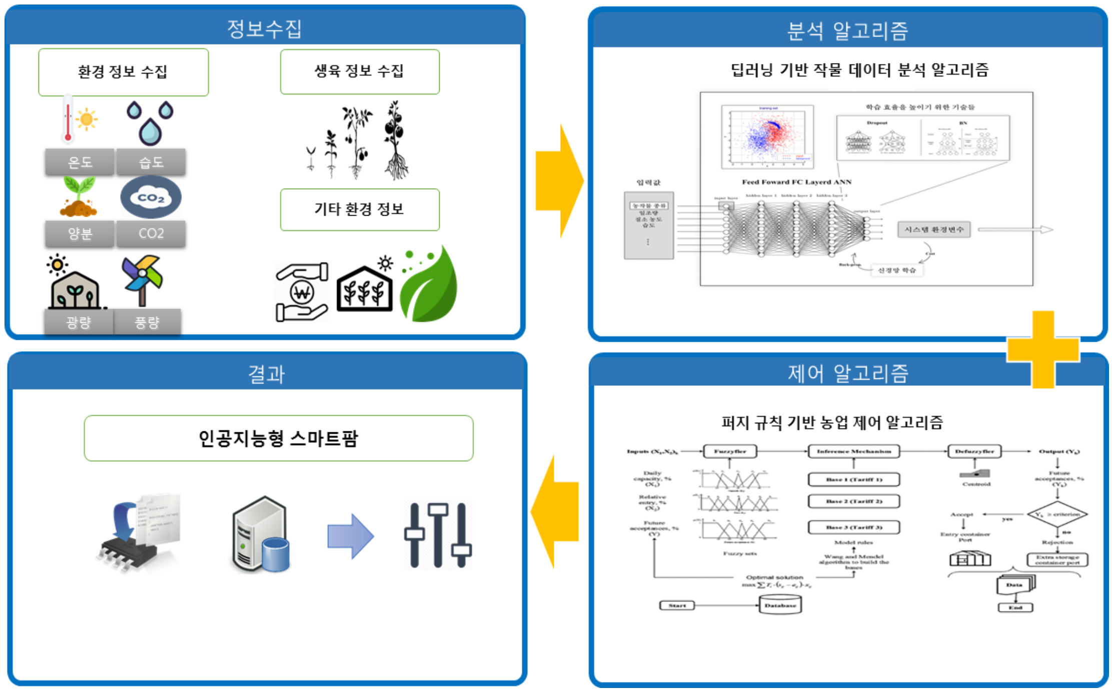


Figure 6. Development process of control framework

3.2 Understanding the Correlation between Data

The following Figure 7. describes the development of a learning algorithm for optimal control of the facility environment.

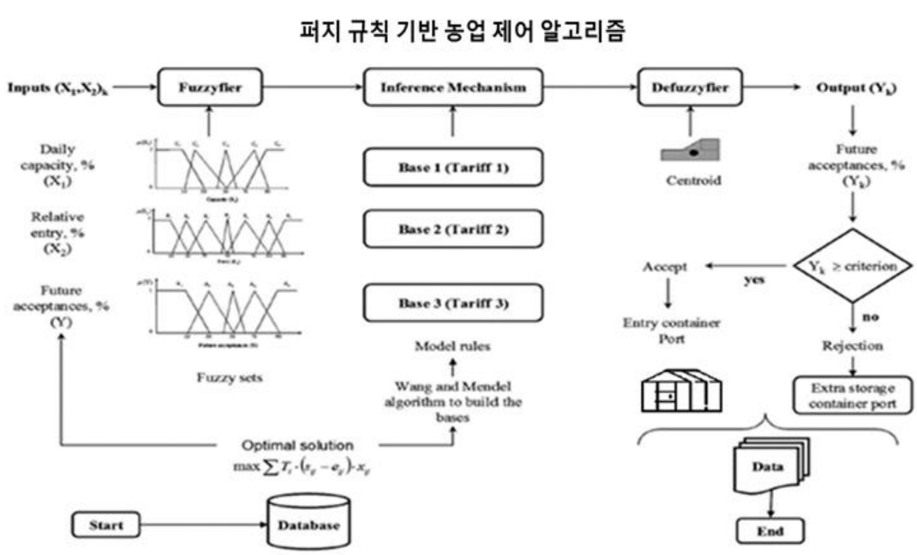


Figure 7. Research on Artificial Intelligence algorithms for optimal control of facility environment.

Framework development through cultivation environment control for Pholiota nameko mushroom - Automatic control, manual control, remote control system development [11].

Analysis of collected data for optimal crop growth and research on artificial intelligence algorithms Application of optimal growth data to crop environment control framework Research on data collection methods for various crop varieties through literature review and research Development of improved embedded systems for automatic control in case of network disruption Modification and improvement of framework through feedback. Testing and feedback through application to mushroom cultivation facilities

Data analysis algorithm is shown in Figure 8.

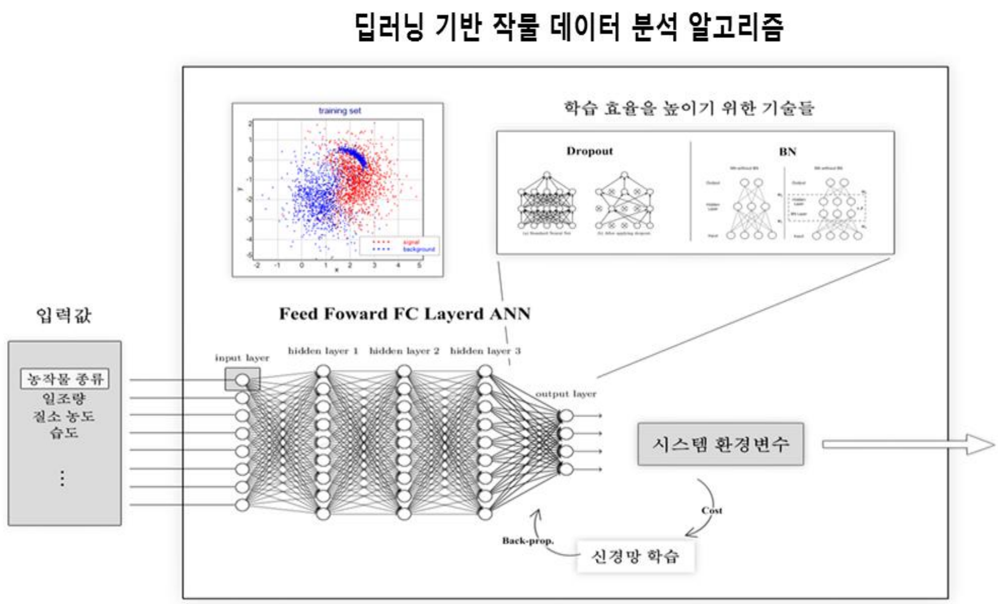


Figure 8. Data analysis algorithm.

Review of improvements and implementation in the system is shown in Figure 9.

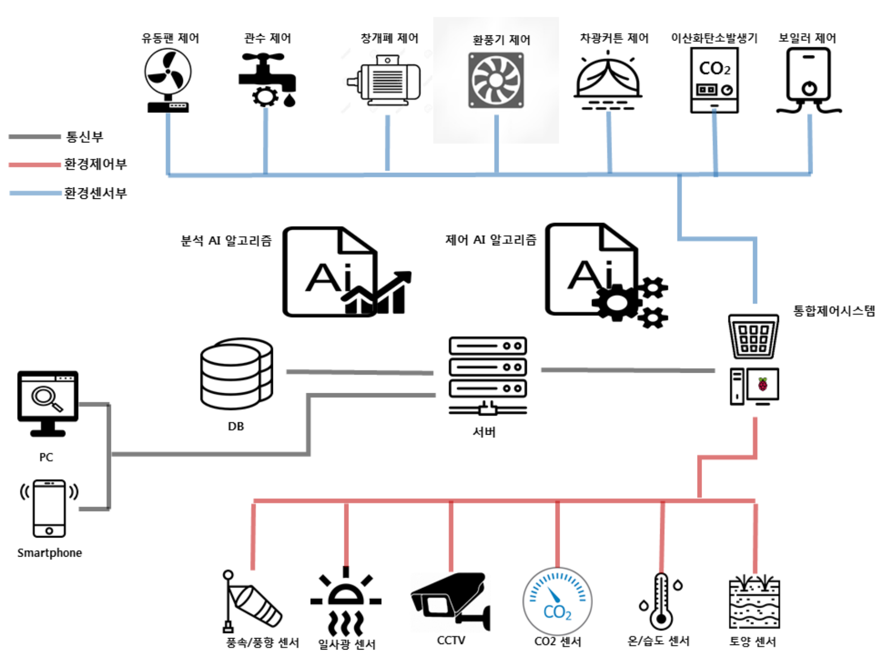
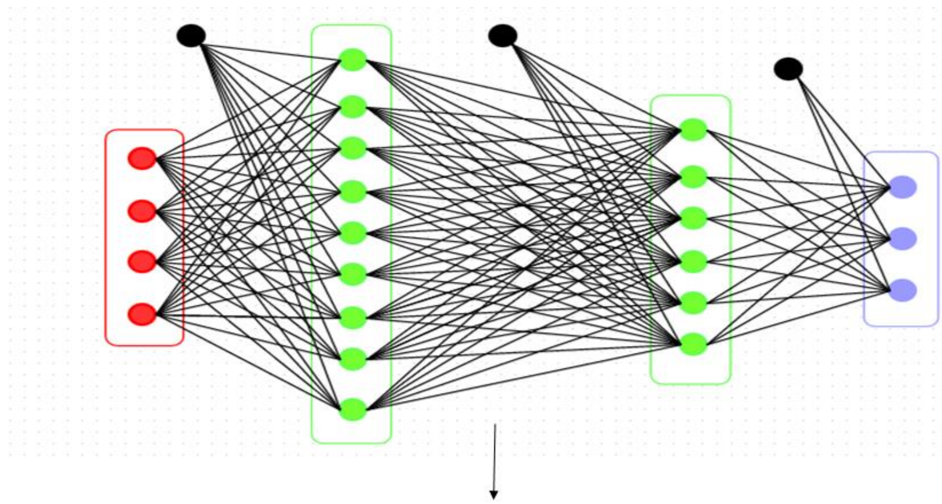


Figure 9. Setting up a testbed environment.

The following Figure 10. is an explanation for building a deep learning analysis model. Deep learning analysis model.

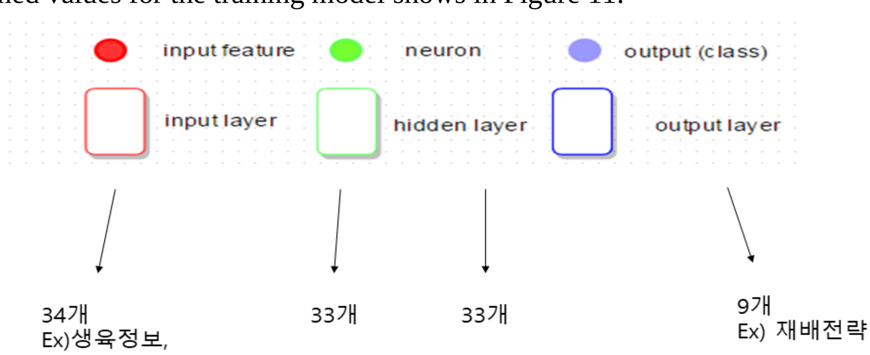


3-layers Fully Connected Neural Network(DNN)

```
# Define the neural network
def neural_net(x_dict):
    # TF Estimator input is a dict, in case of multiple inputs
    x = x_dict['growth_data']
    # Hidden fully connected layer with 256 neurons
    layer_1 = tf.layers.dense(x, n_hidden_1)
    # Hidden fully connected layer with 256 neurons
    layer_2 = tf.layers.dense(layer_1, n_hidden_2)
    # Output fully connected layer with a neuron for each class
    out_layer = tf.layers.dense(layer_2, num_classes)
    return out_layer
```

Figure 10. Deep learning architecture

User-defined values for the training model shows in Figure 11.




```
# Network Parameters
n_hidden_1 = 33 # 1st layer number of neurons
n_hidden_2 = 33 # 2nd layer number of neurons
num_input = 34
num_classes = 9
```

Figure 11. Value of hyperparameters.

Details of the learning model shows in Figure 12.

Update: Adam optimizer
Loss function: SoftMax
Minibatch size :100
Learning rate: 0.1

```
import tensorflow as tf

# Parameters
learning_rate = 0.1
num_steps = 1000
batch_size = 100
display_epoch = 1
```

```
# Define the model function (following TF Estimator Template)
def model_fn(features, labels, mode):

    # Build the neural network
    logits = neural_net(features)

    # Predictions
    pred_classes = tf.argmax(logits, axis=1)
    pred_probas = tf.nn.softmax(logits)

    # If prediction mode, early return
    if mode == tf.estimator.ModeKeys.PREDICT:
        return tf.estimator.EstimatorSpec(mode, predictions=pred_classes)

    # Define loss and optimizer
    loss_op = tf.reduce_mean(tf.nn.sparse_softmax_cross_entropy_with_logits(
        logits=logits, labels=tf.cast(labels, dtype=tf.int32)))
    #optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
    optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate)
    train_op = optimizer.minimize(loss_op, global_step=tf.train.get_global_step())

    # Evaluate the accuracy of the model
    acc_op = tf.metrics.accuracy(labels=labels, predictions=pred_classes)
```

Figure 12. Details of the training model

4. Experiment and Results

4.1 Learning of Growth Information Data

A total of 323 growth data sets (demo data) from the shiitake mushroom farm were used as input and output data for learning shows in Figure 13.

idGrowth	id_device_block	id_svc_org	date	week	growth_length	flower_cluster_height	stem_dia	blossoming_number	fruit_set_number	fruit_number	circumference	internode_length	growth_type	length_growth_rate	x_coordinate	intercalary_rate	intercalary_rate_variation	vigor	y_coordinate	step	retail_trend	recommend
1249	1	1	2017-12-04	2	19.21	27.04	10.29	9.61	6.45	23	3.23106	5	nutrition growth	13	108.75	64.6212	64.6212	vigor strong	88.447	2	46.6038	1
1248	1	1	2017-12-03	1	17	24.63	10.2	6.27	4.11	20	3.2028	3.2028	nutrition growth	0	125	100	0	vigor sustenance	250	2	46.6038	6
1247	1	1	2016-03-28	34	21.38	27.13	9.68	21.77	21.06	25	3.03952	4	nutrition growth	-15.3267	144.158	75.988	-31.2953	vigor weak	328.238	2	17.7191	2
1246	1	1	2016-03-21	33	25.25	23.75	10.25	20.71	20.28	25	3.2185	3	nutrition growth	14.9818	106.273	107.283	36.0053	vigor strong	159.987	2	39.969	1
1245	1	1	2016-03-14	32	21.96	22.38	9.08	20	19.48	25	2.85112	4	reproduction growth	-6.07357	382.592	71.278	22.0323	vigor strong	194.919	2	36.3016	1
1244	1	1	2016-03-07	31	23.38	27.92	9.41	19.32	18.6	24	2.95474	6	reproduction growth	16.9	353.875	49.2457	5.01652	vigor strong	237.459	2	14.3854	1
1243	1	1	2016-02-29	30	20	29.38	9.86	18.57	17.91	24	3.09604	7	nutrition growth	11.1111	111.111	44.2291	-5.53986	vigor weak	263.85	2	12.8576	1
1242	1	1	2016-02-22	29	18	23.42	9.51	17.97	17.3	23	2.98614	6	no growth	-22.8461	250	49.769	-27.7105	vigor weak	319.276	2	35.9214	1
1241	1	1	2016-02-15	28	23.33	27.17	9.87	17.34	16.66	23	3.09918	4	nutrition growth	20.1339	99.8326	77.4795	-30.2225	vigor weak	325.556	2	46.6038	1
1240	1	1	2016-02-08	27	19.42	27.04	10.29	16.51	15.97	21	3.23106	3	nutrition growth	14.2353	107.206	107.702	43.646	vigor strong	140.885	2	51.7999	1
1239	1	1	2016-02-01	26	17	23.88	10.2	15.76	15.11	19	3.2028	5	reproduction growth	8.48756	364.391	64.056	-16.014	vigor weak	290.035	2	39.1525	1
1238	1	1	2016-01-25	25	15.67	28.17	10.2	15.04	14.41	18	3.2028	4	reproduction growth	2.7541	371.557	80.07	-22.8173	vigor weak	307.043	2	14.5244	1
1237	1	1	2016-01-18	24	15.25	28.25	9.83	14.28	13.7	16	3.08662	3	reproduction growth	-14.0845	392.606	102.887	58.5685	vigor strong	103.579	2	12.9168	1
1236	1	1	2016-01-11	23	17.75	31.88	9.88	13.78	13.17	15	3.10232	7	nutrition growth	10.9375	111.328	44.3189	-27.7441	vigor weak	319.36	2	27.8353	1
1235	1	1	2016-01-04	22	16	27.88	9.18	13.19	12.55	15	2.88252	4	nutrition growth	5.75017	117.812	72.063	-21.0903	vigor weak	302.726	2	45.977	1
1234	1	1	2015-12-28	21	15.13	27.75	8.9	12.53	11.9	13	2.7946	3	reproduction growth	21.4286	348.214	93.1533	46.315	vigor strong	134.212	2	47.4801	1
1233	1	1	2015-12-21	20	12.46	29.79	8.95	11.71	11.28	12	2.8103	6	reproduction growth	-14.069	392.586	46.8383	-20.7502	vigor weak	301.875	2	51.0258	1
1232	1	1	2015-12-14	19	14.5	34.04	8.61	11.35	11	13	2.70354	4	nutrition growth	9.76533	112.793	67.5885	22.3202	vigor strong	194.2	2	44.041	1
1231	1	1	2015-12-07	18	13.21	29.58	8.65	11.15	10.66	14	2.7161	6	reproduction growth	-8.1363	385.17	45.2683	0.523333	vigor strong	248.692	2	21.493	1
1230	1	1	2015-11-30	17	14.38	35.63	8.55	10.58	10.05	17	2.6847	6	nutrition growth	-25.1432	156.429	44.745	6.16786	vigor strong	234.58	2	7.45241	1
1229	1	1	2015-11-23	16	19.21	34.83	8.6	10.13	9.59	16	2.7004	7	reproduction growth	-12.9982	391.248	38.5771	-31.4449	vigor weak	328.612	2	0.579344	1
1228	1	1	2015-11-16	15	22.08	36.33	8.92	9.61	8.97	18	2.80088	4	nutrition growth	0.914077	123.857	70.022	-2.512	vigor weak	256.28	2	-7.71155	1
1227	1	1	2015-11-09	14	21.88	31.46	9.24	9.13	8.59	19	2.90136	4	no growth	-16.9009	250	72.534	21.98	vigor strong	195.05	2	-25.6105	1
1226	1	1	2015-11-02	13	26.33	33.79	9.66	8.48	7.88	20	3.03324	6	reproduction growth	-7.74352	384.679	50.554	-1.099	vigor weak	252.747	2	-40.0402	1

Figure 13. Details of the training model

- Training set: A data set for learning is randomly extracted 80% of the entire data shows in Figure 14.

```
In [5]: data = getData2.Get_data()
        #for x in data:
        #    print(x)

        shuffle_data = np.random.permutation(data)
        length = len(shuffle_data)
        train = int(length * 0.8)
        test = int(length * 0.2)
        train_data = shuffle_data[:train]
        test_data = shuffle_data[train+1:]

        print(train_data)

[[ list([40, 109.375, 14.5, 2, 2300, 2100, 1900, 1800, 1700, 1900, 1700, 1900, 1900, 2100, 2000, 2000, 2000, 18
00, 1800, 1800, 1800, 1800, 1800, 1800, 1900, 1900, 1900, 1900, 2000, 2000, 1800, 1600, 1500, 1500])
  0]
 [ list([2, 250.0, 212.068, 1, 1500, 1500, 1500, 1400, 1400, 1500, 1800, 1900, 2000, 2000, 2000, 2000, 2000, 18
00, 1900, 1800, 1800, 1800, 1800, 1800, 1800, 1800, 1500, 1500, 1600, 1600, 1400, 1500, 1700, 1700])
  6]
 [ list([10, 117.604, 286.385, 2, 2500, 2300, 2500, 2700, 2800, 3000, 3300, 3700, 3900, 4100, 4000, 4200, 4000,
4000, 4000, 4200, 4200, 4200, 4000, 4000, 4000, 3000, 2800, 2800, 3000, 2800, 2800, 2700, 2700, 2700])
  1]
 ...,
 [ list([17, 0.0, 216.834, 2, 2700, 2700, 2900, 2800, 3000, 3200, 3400, 3500, 3300, 3500, 3600, 3600, 3600, 370
0, 3700, 3500, 3300, 3400, 3300, 3300, 3300, 3200, 3000, 3000, 3000, 3000, 3000, 3000, 3300, 3500])
  1]
 [ list([32, 250.0, 119.167, 2, 3600, 3600, 3400, 3200, 3500, 3400, 3300, 3100, 3300, 3500, 3500, 3300, 3200, 3
100, 3100, 3000, 2900, 2900, 3000, 2800, 2800, 2800, 2800, 2800, 2800, 2800, 2700, 2800, 2900, 2700])
  0]
 [ list([1, 375.0, 297.347, 1, 2300, 2000, 1800, 2000, 2000, 2000, 2000, 1800, 1800, 1800, 1800, 1800, 1900, 19
00, 1800, 1800, 1800, 1600, 1600, 1500, 1600, 1500, 1500, 1500, 1700, 1600, 1600, 1600, 1800, 1600])
  6]]
```

Figure 14. Training set

Test set: The data set for validation is selected by 20% of the total data shows in Figure 15.

```
In [8]: data = getData2.Get_data()
        #for x in data:
        #    print(x)

        shuffle_data = np.random.permutation(data)
        length = len(shuffle_data)
        train = int(length * 0.8)
        test = int(length * 0.2)
        train_data = shuffle_data[:train]
        test_data = shuffle_data[train+1:]


        print(test_data)

[[ list([1, 250.0, 157.174, 2, 1800, 1800, 1900, 1900, 1800, 1800, 1800, 1600, 1600, 1500, 1600, 1500, 1500,
1500, 1700, 1600, 1600, 1600, 1800, 1600, 1500, 1700, 1700, 1700, 1700, 1700, 1800, 1800, 1800, 1700])
  1]
 [ list([13, 250.0, 341.867, 1, 2600, 2600, 2600, 2800, 2800, 2800, 2600, 2600, 2800, 2500, 2400, 2500, 2500,
2700, 2500, 2600, 2700, 2700, 2900, 2800, 3000, 3200, 3400, 3500, 3300, 3500, 3600, 3600, 3600, 3700])
  6]
 [ list([39, 408.333, 253.925, 2, 2800, 2800, 2800, 2800, 2800, 2600, 2500, 2300, 2100, 1900, 1800, 1700, 190
0, 1700, 1900, 1900, 2100, 2000, 2000, 2000, 1800, 1800, 1800, 1800, 1800, 1800, 1800, 1900, 1900, 1900])
  0]
 [ list([6, 125.0, 173.986, 1, 2700, 2700, 2900, 2800, 3000, 3200, 3400, 3500, 3300, 3500, 3600, 3600, 3600,
3700, 3700, 3500, 3300, 3400, 3300, 3300, 3300, 3200, 3000, 3000, 3000, 3000, 3000, 3000, 3300, 3500])
  6]
 [ list([6, 434.211, 290.08, 2, 1000, 1000, 1100, 1100, 1100, 1100, 1100, 1100, 1400, 1500, 1600, 1600, 1700,
1600, 1600, 1500, 1500, 1500, 1500, 1700, 1600, 1800, 1900, 2000, 1800, 1600, 1800, 1800, 1800, 2000])
  1]
 [ list([45, 363.636, 341.583, 2, 2000, 2000, 1800, 1600, 1500, 1500, 1500, 1500, 1500, 1500, 1400, 1300, 150
0, 1700, 1700, 1700, 1600, 1600, 1700, 1800, 1800, 1600, 1700, 1700, 1600, 1600, 1400, 1200, 1200, 1200])
```

Figure 15. Test set

Correlation analysis using learned data Analysis of Best and Worst based on temperature, humidity, water, and sunlight.

Training results and training progress are shown in Figure 16 and Figure 17 respectively.

```
#generate comparison table
columns = ['temp', 'humi', 'Co2', 'insolation']
index = ['stage_1', 'stage_2', 'stage_3']

best_stage_1 = best_season_data[:4]
best_stage_1_ave = np.mean(best_stage_1, axis = 0)

best_stage_2 = best_season_data[4:8]
best_stage_2_ave = np.mean(best_stage_2, axis = 0)

best_stage_3 = best_season_data[8:12]
best_stage_3_ave = np.mean(best_stage_3, axis = 0)


best_stages_data = np.column_stack((best_stage_1_ave,
                                    best_stage_2_ave,
                                    best_stage_3_ave))
best_stages_data = np.transpose(best_stages_data)


df_1 = pd.DataFrame(best_stages_data, columns = columns, index = index)
print('The best season environment information: ')
print(df_1)

worst_stage_1 = worst_season_data[:4]
worst_stage_1_ave = np.mean(worst_stage_1, axis = 0)

worst_stage_2 = worst_season_data[4:8]
worst_stage_2_ave = np.mean(worst_stage_2, axis = 0)

worst_stage_3 = worst_season_data[8:12]
worst_stage_3_ave = np.mean(worst_stage_3, axis = 0)


worst_stages_data = np.column_stack((worst_stage_1_ave,
                                     worst_stage_2_ave,
                                     worst_stage_3_ave))
worst_stages_data = np.transpose(worst_stages_data)

df_2 = pd.DataFrame(worst_stages_data, columns = columns, index = index)
print('The worst season environment information: ')
print(df_2)
```

Figure 16. The result of training

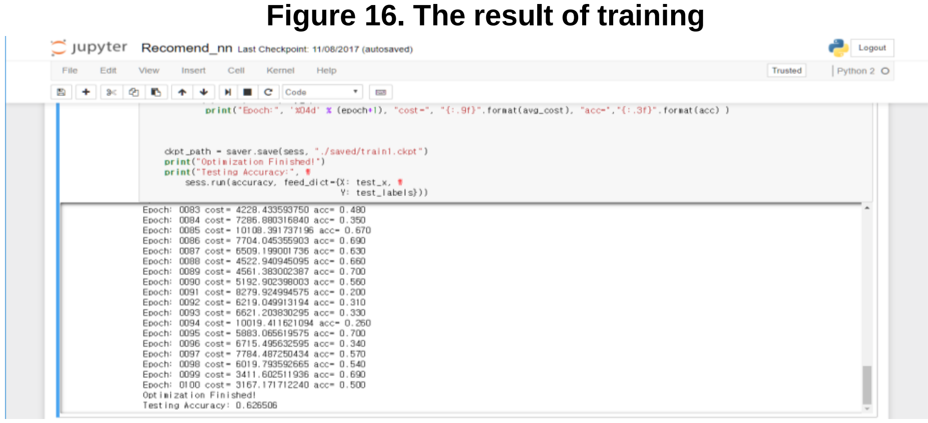


Figure 17. Training progress

Training Status Graph The graph below visualizes the accuracy of the learning model during the process of creating the model. It shows that the accuracy increases as the weight is updated. The X -axis on the graph represents the number of iterations (the number of times the weight is updated), and the Y-axis represents the accuracy in percentage. As the iteration progresses, it can be observed that the accuracy gradually increases. The Figure 18 shows training monitoring.

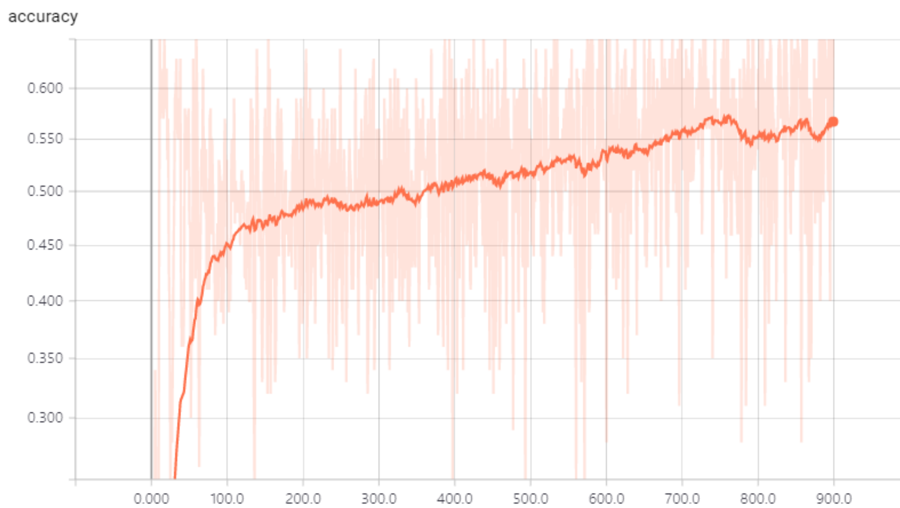


Figure 18. Training monitoring

Cross-entropy graph Figure 19 below is a graph that shows the error rate in terms of CE (Cross-entropy) as learning progresses [12]. There are three methods for calculating the error rate, which are CE, SSE (Sum Square Error), and MSE (Mean Square Error), but CE, which is commonly used recently, was used for the calculation in this project. In the CE graph, the X-axis represents the number of iterations as above, and the Y-axis represents the CE value. Also, it can be seen that the error rate gradually decreases as the iteration progresses.

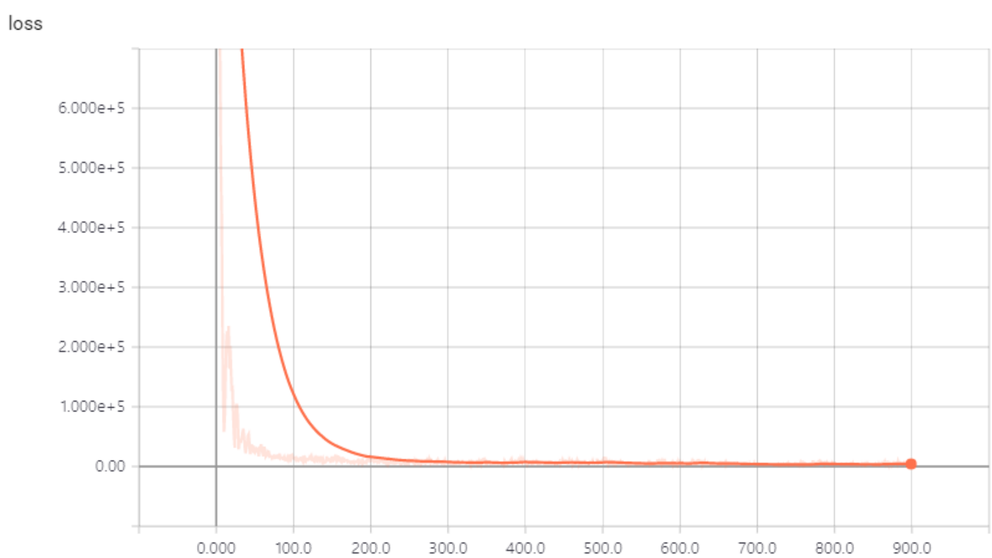


Figure 19. Cross Entropy graph

Recommend cultivation strategies based on the prediction results of the trained decision-making system, using the input of growth information to suggest the strategy with the highest probability value from the trained model. Figure 20 shows Extracting environmental information for the best and worst times.

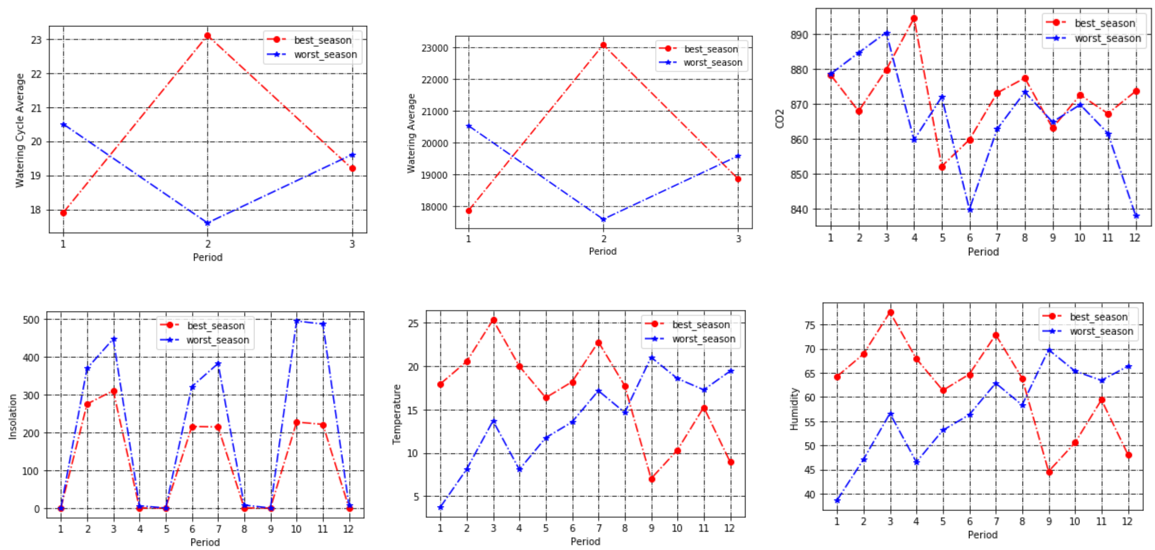


Figure 20. Extracting environmental information for the best and worst times.

Big data visualization Analyze data correlations through big data processing Visualize the analy zed results show in Figure 21.

```
The best season environment information:
              temp       humi        Co2   insolation
stage_1  20.951675  69.65665  880.10000   146.116750
stage_2  18.771675  65.71835  865.63325   107.383167
stage_3  10.394600  50.66585  869.15450   111.937500
The worst season environment information:
              temp       humi        Co2   insolation
stage_1   8.409992  47.210025  878.33325  205.266582
stage_2  14.276675  57.656675  862.03325  177.483249
stage_3  19.072100  66.240400  858.60425  246.424917
```

Figure 21. Extraction of best & worst environmental information.

5. Conclusion

In recent years, the use of AI techniques in agriculture has become increasingly popular due to their potential to optimize the yield and quality of crops while minimizing resources and costs. In this study, we developed an IoT-based AI software platform for optimal control of facility houses, which is an essential tool for precise cultivation management. By collecting and analyzing growth data from mushroom crops and integrating it with internal and external environmental data, we built a more accurate AI platform for facility house control. The platform enables the optimization of environmental control factors, such as temperature, humidity, and CO2 concentration, for each crop variety to achieve optimal growth conditions and higher yield. The experimental results on a test bed showed that our AI platform significantly outperformed the existing system, reducing the user's risk burden in setting the growth environment and increasing overall efficiency. The proposed framework has the potential to enhance the sustainability and productivity of the agriculture industry, making it an important step towards smart farming.

Acknowledgement

This work was supported by project for Joint Demand Technology R&D of Regional SMEs funded by Korea Ministry of SMEs and Startups in 2023.(Project No. RS-2023-00207672)

References

[1] O'Shaughnessy, Susan A., et al. "Towards smart farming solutions in the US and South Korea: A comparison of the current status." Geography and Sustainability 2021.
https://doi.org/10.1016/j.geosus.2021.12.002

[2] De Vries, Albert, Nikolay Bliznyuk, and Pablo Pinedo. "Invited Review: Examples and opportunities for artificial intelligence (AI) in dairy farms." Applied Animal Science 39.1, 2023: 14-22.
https://doi.org/10.15232/aas.2022-02345

[3] Minsky, Marvin. "Future of AI technology." 1992.

[4] Zimmermann, H-J. "Fuzzy set theory." Wiley interdisciplinary reviews: computational statistics 2.3, 2010: 317-332.
https://doi.org/10.1002/wics.82

[5] Fan, Jianqing, Fang Han, and Han Liu. "Challenges of big data analysis." National science review 1.2, 2014: 293-314.
https://doi.org/10.1093/nsr/nwt032

[6] Boursianis, Achilles D., et al. "Internet of things (IoT) and agricultural unmanned aerial vehicles (UAVs) in smart farming: a comprehensive review." Internet of Things 18, 2022: 100187.
https://doi.org/10.1016/j.iot.2020.100187

[7] Malrey Lee, "Final Report for Small and Medium Business Administration: IOT-based artificial intelligence development of a framework for environmental control of wood ear mushrooms," Oct. 2019.

[8] S. H. Jang, K. W. Nam, and Y. G. Jung, "Smart Building Block Toys using Internet of Things Technology," International Journal of Advanced Culture Technology, vol. 4, no. 2, pp. 34–37, Jun. 2016.
https://doi.org/10.17703/IJACT.2016.4.2.34

[9] Madakam, Somayya, et al. "Internet of Things (IoT): A literature review." Journal of Computer and Communications 3.05, 2015: 164.
https://doi.org/10.4236/jcc.2015.35021

[10]Jo, Jae-Yeong. "Water quality of agricultural groundwater in Western Coast area and Eastern Mountain Area of Jeollabuk-do." Journal of Applied Biological Chemistry 54.3, 2011: 218-224.
https://doi.org/10.3839/jabc.2011.036

[11] Arita, I. K. U. O. Pholiota nameko. New York, NY: Academic Press, 1978.

[12] De Boer, Pieter-Tjerk, et al. "A tutorial on the cross-entropy method." Annals of operations research 134, 2005: 19-67.
https://doi.org/10.1007/s10479-005-5724-z