Jurnal Teknik Informatika Vol. 15 No. 2, 2022 (137-146)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v15i2.26762

JURNAL TEKNIK INFORMATIKA

Homepage : http://journal.uinjkt.ac.id/index.php/ti

Action Recommendation Model Development for Hydromon Application Using Deep Neural Network (DNN) Method

Meida Cahyo Untoro[1], Eko Dwi Nugroho[2], Mugi Praseptiawan[3*], Aidil Afriansyah[4], Muhammad Nadhif Athalla[5]

[1,2,3,4,5]Informatics Engineering, Production Technology and Industry Department, Institute Technology of Sumatera
[1,2,3,4,5]Terusan Ryacudu, Way Hui, Jati Agung, Lampung Selatan, 35365
E-mail: [1]cahyo.untoro@if.itera.ac.id, [2]eko.nugroho@if.itera.ac.id, [3]mugi.praseptiawan@if.itera.ac.id, [4]aidil.afriansyah@if.itera.ac.id, [5]muhammad.119140209@student.itera.ac.id

ABSTRACT

Article:
Accepted: July 16, 2022
Revised: May 24, 2022
Issued: November 15, 2022

***Correspondence Address:**
mugi.praseptiawan@if.itera.ac.id

Controlling hydroponic plants, which is currently being carried out manually, can be said to be less effective because it still involves the hard work of farmers to continuously monitor the condition of the hydroponic plants. Therefore, the general objective of this research is to develop a model that can be used as a recommendation system for actions that farmers need to take based on hydroponic crop conditions. The model formed with this machine learning method will then be used in the Hydromon application which allows farmers to manage and monitor the condition of hydroponic plants and take action based on the recommendations given. This model was developed using a deep neural network algorithm consisting of five layers with the help of the TensorFlow framework. The results show that the model is accurate with an accuracy value of 96.47% on the test data to classify plant conditions so that it can be used in the Hydromon application.

Keywords: *Hydroponic, Recommendation, Machine Learning, TensorFlow, Hydromon*

I. INTRODUCTION

This Agriculture is the foundation of human civilization and is responsible for producing high-quality food for the inhabitants of the world. Along with population growth and infrastructure development that can lead to a lack of agricultural land, good farming methods need to be used so as to guarantee the availability of cheap and quality products to meet food needs. Based on these problems, farming activities using the hydroponic method are expected to overcome these problems.

Hydroponics is a method of growing crops without the use of soil, using water as a solvent according to the nutrients needed by plants. However, plants developed by the hydroponic method are quite vulnerable so their conditions need to be continuously controlled so that their growth can run optimally. Currently, monitoring the condition of hydroponic plants is still done manually so it takes hard work involving hydroponic plant farmers to continuously monitor the condition of the plants and provide the necessary actions for the plants directly. This is considered ineffective because it requires more time and energy to do so, but it needs to be done to ensure the plants grow in good conditions.

Hydroponics is a method of growing crops without the use of soil. This method uses water as the solvent according to the nutrients that the plant needs. The advantage of the hydroponic method is that it can provide more optimal conditions for plant growth[1].

However, hydroponic farming methods have a higher level of complexity compared to conventional farming methods. This method is said to be complicated because it involves various equipment, such as containers, pumps, lamps, nutrients, and so on at a fairly high cost[2]. In addition, hydroponic plants are plants that are also fairly vulnerable so their condition needs to be continuously monitored and controlled.

This problem of monitoring and controlling hydroponic plants can be solved with the help of an Internet of Things (IoT) tool that allows farmers to automate the monitoring methods of such hydroponic plants. IoT is a development of technological devices that have the ability to receive data, process data, and send information to users with the help of software that is directly integrated with the internet[3].

Thus, various kinds of sensors and control systems can be used to automate agricultural systems with the hydroponic method. This automation process is also supported by the use of a recommendation system built with a machine learning model in the application to inform the actions that need to be taken by farmers based on the condition of the hydroponic plant.

Artificial intelligence (AI) or artificial intelligence is a data processing process based on analytical models that produce predictions, rules, answers, recommendations, or other similar results. This analytical model is built using programming concepts that explicitly know the relationship between the procedure and decision logic in a system that has intelligence[4].

Machine learning is a branch of science part of artificial intelligence built with programming languages that allow computers to think like humans and can improve their understanding through experience automatically. Machine learning has a focus on developing systems that are able to learn on their own to decide things without having to be programmed by humans through learning models built on the data owned[5].

There are 4 learning processes in machine learning, namely Supervised Learning, Unsupervised Learning, Semi-supervised Learning, and Reinforcement Learning[6]. The machine learning model developed in this study aims to create a system of action recommendations that need to be carried out by hydroponic plant farmers based on the condition of the hydroponic plant.

Deep learning is part of machine learning whose algorithms are inspired by the structure of the human brain. This algorithm is able to adapt to large amounts of data to solve various problems that are difficult to solve with other machine learning algorithms. Deep learning has gone beyond advanced approaches in computer vision and natural language processing to achieve previously unimaginable advances for challenging problems[7].

Deep learning is a model that can be used effectively on various devices such as IoT or mobile devices that have limited resources[8]. Deep learning has three types of parameters, namely weights and biases (model parameters, hyperparameters, and task parameters[9].

Weights and biases are linear parameters or tensors in the mock conditional network set during the training period. Hyperparameters are initial global variables that are manually set and affect the behavior of the function, its training algorithm, and its neural network architecture. Assignment parameters are parameters defined for a particular case. The parameter lies in the requirements of the problem, which need to be met and cannot be changed.

Neural Network is one of the machine learning algorithms inspired by how the human brain works. The machine learning algorithm consists of many neurons that are interconnected with each other. Meanwhile, deep neural networks are part of deep learning which is a development of the method by adding more layers to the architecture.

In this study, the algorithm architecture of deep neural networks was built with the help of the TensorFlow framework. The algorithm consists of several layers and the use of hyperparameter values adapted to the aim of building a model of recommendations for actions that farmers can perform based on the condition of hydroponic plants monitored in real-time[10].

Unlike conventional methods, deep neural networks trade domain expertise in feature engineering and model training with automated feature learning that exists on the availability of large amounts of data[7]. While on the one hand, the large number of network parameters allows greater flexibility and allows deep neural networks to model more complex relationships, training deep neural networks take a lot of time.

For example, a deep neural network may require several weeks or months of training time, running on a cluster of high-performance computers or application-specific integrated circuits. This is in contrast to traditional machine learning algorithms which only take a few hours at most on a laptop computer. On the other hand, deep neural network's phase inference is very fast, so it can run in real-time on small, heavy, and powerful hardware, such as embedded processors[7].

The most important concept in deep neural network is their weight[11]. An example of a deep neural network can be seen in Figure 1. In Figure 1 there are four input units and two output units. In a deep neural network, it is necessary to find one or more patterns from the input layer, so that these patterns can be used to classify one output layer. To do so, it is necessary to design a number of hidden layers with an activation function Figure 1 consists of three hidden layers with each layer consisting of seven hidden units.

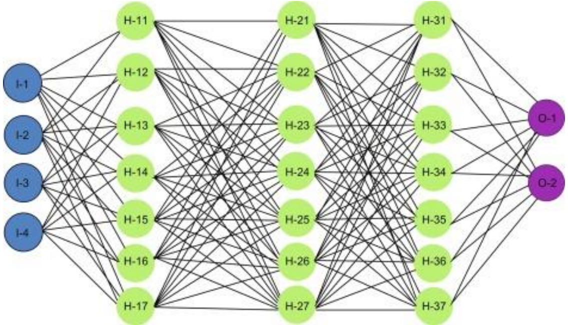


Figure 1. Deep Neural Network dengan tiga lapisan tersembunyi[11]

Deep neural networks excel at finding hierarchical representations that accomplish complex tasks through large data sets and have been demonstrated in studying many object classes that play an important role in classifying classes[12]. Deep neural networks have become a standard tool for solving various computer vision problems[13].

Deep neural networks typically use non-linear activation functions that allow to create complex non-linear mappings between the given input and the resulting output, which is important for studying and modeling complex data, such as images, videos, audio, and in data sets. general where the relationship between the desired input and output is not linear[14]. The calculation for the hidden layer can be seen in formula 1[15].

$$f(x)=f[a^{(L+1)}(h^{(L)}(a^{(L)}(\ldots(h^{(2)}(a^{(2)}(h^{(1)}(a^{(1)}(x)))))))))] \quad (1)$$

With hidden layer output $h^{(l)}(x)$. Each preactivation function $a^{(l)}(x)$ is usually a linear operation with a matrix $W^{(l)}$ and bias $b^{(l)}$, which can be combined into a parameter Θ. The notation "hat" $\hat{x}$ indicates that 1 has been added to the vector x. The hidden layer activation function $h^{(l)}(x)$ often has the same form at each level, but this is not a requirement.

$$\begin{aligned}
&a^{(l)}(x) = W^{(l)}x+b^{(l)}, \\
&a^{(l)}(\hat{x}) = \Theta^{(l)}\hat{x} && , l = 1 \\
&a^{(l)}(\hat{h}^{(l-1)} = \Theta^{(l)}\hat{h}^{(l-1)} && , l > 1 \qquad (2)
\end{aligned}$$

Tensorflow is an open-source deep-learning framework used for machine learning applications. This framework can facilitate the machine learning process in the fields of speech recognition, image classification, object detection, text classification, etc. TensorFlow framework can be used to train deep learning models to perform object detection and classification in real-time[16]. lite is an open-source deep learning framework that can be used on devices to assist with machine learning modeling on those devices[17].

TensorFlow was created by researchers at Google and is the most popular among most deep learning libraries. TensorFlow greatly facilitates and speeds up the research and application of neural network models and applies its concepts to build and train convolutional neural network models for classification[18].

Based on these problems and potential solutions, in general, this study aims to develop a model that allows farmers to manage and monitor hydroponic crop conditions and take action based on the recommendations given. The model created covers the entire process and is integrated into a mobile application called Hydromon to assist in the process of cultivating hydroponic plants. In particular, this study aims to develop a system of recommendations for actions that need to be carried out by farmers based on the conditions of the hydroponic plants.

The model made later can be made into a tool which is an Internet of Things, so that farmers can really feel the benefits. This research will use methods, namely Deep Neural Network and also TensorFlow, so this research is included in Artificial Intelligence and also the Internet of Things. Deep Neural Network was chosen because the authors considered that they were able to represent the problem of this research after studying and comparing it with other learning methods.

II. METHODOLOGY

This research was developed using several stages, namely business process understanding, data understanding, data preparation, modeling, evaluation, and deployment. The stages carried out are as follows.

2.1 Business Understanding

The business understanding stage is the stage of defining the problem and the purpose of a study. The purpose of this study is to build a model that can provide recommendations for actions that need to be carried out by farmers based on the condition of the hydroponic plant. The model is built using a deep neural network architecture by training data that is formed independently based on three attribute conditions of hydroponic plants, namely pH, TDS, and light intensity. The limitations of this study are the condition of training and validation data that is not obtained from valid sources and the number of hydroponic plant attributes that are controlled is still relatively minimal.

2.2 Data Understanding

The data understanding stage is the stage of introducing the data used for the data processing process so that the data used in the research also needs to be adjusted to the purpose of the study. Therefore, to meet the research objectives, data with three attributes are needed that describe the condition of hydroponic plants, namely the pH, TDS, and light intensity attributes. Based on these three attributes, 18 labels or actions can be formed by farmers. The training data and validation data are then formed independently according to the number of labels obtained. That is, if there are 18 labels, then each label is formed with 1000 data that reference the various possible conditions of the hydroponic plant. Meanwhile, the test data is obtained directly by taking data from sensors that are integrated with hydroponic plants.

2.3 Data Preparation

The data preparation stage is the stage of processing raw data into data that is ready to be used for data processing. Some of the stages of data preparation carried out are data retrieval based on the three data attributes to be used, randomization of data with the shuffling method, adjustment of data type and size, sharing of training data and validation data, and normalizing data with the MinMaxScaler method.

2.4 Modeling

The modeling stage is the stage for implementing algorithms for the data processing process. This stage needs to be adjusted to the data owned so that the results of data processing can be in accordance with their purpose. In this study, a deep neural network algorithm with 5 layers in it was formed with the aim of classifying multiclass labels. The first modeling process carried out is the creation of a checkpoint function with the help of the TensorFlow library to store temporary training results in personal storage, as well as a training stop function with the help of the TensorFlow library which is used to stop training when the missing value of validation data has reached the desired value. The next process is the formation of a deep neural network model with five layers consisting of one input layer, three hidden layers, and one output layer in it.

This model is also formed with the help of the TensorFlow library, then the formation of a compiler to run the model using three parameters, namely optimizer, loss, and metrics. The next process is to execute the training process with an adjusted amount of training, this training process also applies some processes that have previously been carried out. After the training process is complete and the value of accuracy and validation can be observed, the next process is to form a form of visualization that can describe the accuracy of the training and validation process.

2.5 Evaluation

The evaluation stage is carried out to see the quality of the model that has been formed. This stage is carried out by looking at the evaluation value of the model, then testing it using data taken directly from sensors integrated with hydroponic plants. After the data is obtained, the next process is the definition of several action recommendations that farmers can do, of course, the action recommendations that are defined manually are adjusted to the labeling process that has been carried out at the data understanding stage. The next process is testing the model with the data that has been taken before, then the results of the test will be visualized.

2.6 Deployment

The deployment stage carried out is to document the results of the research in the form of making research reports. In addition, another deployment process is the process of converting the model into a TensorFlow lite that can be used in IoT tools and Hydromon applications.

III. RESULTS AND DISCUSSION

The business understanding stage of this study is to build a model with a deep neural network architecture by training data that is formed independently based on three hydroponic plant conditions, namely pH, TDS, and light intensity. The model can provide recommendations for actions that farmers need to take based on the condition of the hydroponic plant.

Table 1. Business understanding

Application	Hydromon Application (Hydroponic plant monitoring and control application)
Output Ideal	Recommend actions based on real-time conditions
Objectives	Predicting control actions based on parameters collected from hydroponic plants

The next stage is the data understanding process to obtain data that can be used in accordance with the research objectives. Thus, data was formed with three attributes that describe the condition of hydroponic plants, namely the pH, TDS, and light intensity attributes. Based on these three attributes, 18 labels or actions can be formed by farmers. The training data and validation data are then formed independently according to the number of labels obtained. That is, if there are 18 labels, then each label is formed with 1000 data that reference the various possible conditions of the hydroponic plant. Meanwhile, the test data is obtained directly by taking data from sensors that are integrated with hydroponic plants.

Table 2. Multiclass classification

Output Model	Multiple Classification (18 action categories)
	0 No Action
	1 Pump Water
	2 Light On
	3 Pump Nutrient
	4 pH up
	5 pH down
	6 Pump water + light on
	7 Pump water + ph up
	8 pump water + ph down
	9 pump water + light on + ph up
	10 pump water + light on + ph down
	11 light on + pump nutrient
	12 light on + ph up
	13 light on + ph down
	14 light on + pump nutrient + ph up
	15 light on + pump nutrient + ph down
	16 pump nutrient + ph up
	17 pump nutrient + ph down

Table 3. Data to be processed

No	...	TDS	Light Intensity	pH	...	Label
0	...	993	689	6268	...	0
1	...	735	462	6044	...	0
2	...	750	778	5599	...	0
3	...	934	491	6090	...	0
...	...	...	...	...	...	...
997	...	1219	969	4728	...	9
998	...	1019	866	5272	...	9
999	...	1053	907	5326	...	9

Table 4. Data distribution by class

Class	Sum	Percentage
0	1000	5.56%
1	1000	5.56%
2	1000	5.56%
3	1000	5.56%
4	1000	5.56%
5	1000	5.56%
6	1000	5.56%

Class	Sum	Percentage
7	1000	5.56%
8	1000	5.56%
9	1000	5.56%
10	1000	5.56%
11	1000	5.56%
12	1000	5.56%
13	1000	5.56%
14	1000	5.56%
15	1000	5.56%
16	1000	5.56%
17	1000	5.56%
Total	18000	100.00%

The next stage is the preparation of data before the model creation process is carried out. This stage implements several stages of data preprocessing. The first stage is data retrieval according to three data attributes that will be used for the training process, namely pH, TDS, and light intensity. The data will then be scrambled by the data shuffling method.

Table 5. Data after going through the shuffling process

No	TDS	Light Intensity	pH	Label
0	1139	962	6268	6
1	684	995	6044	11
2	1151	985	5599	10
3	1028	575	6090	7
…	…	…	…	
17997	1388	856	4728	6
17998	1121	951	5272	9
17999	459	920	5326	11

The next stage is to adjust all the data with a similar data type, namely the float32 data type. The next stage is the division of all the data into training data which amounts to 80% (14,400 data) and validation data as much as 20% (3600 data) of the total data. The next stage is the normalization of the training data and validation data with the MinMaxScaler method.

Table 6. Train data after going through the normalization process

No	pH	TDS	Light Intensity
0	0.490833	0.861667	0.364933
1	0.239167	0.683333	0.024012
2	0.017500	0.411667	0.091296
3	0.655000	0.820000	0.165082
…	…	…	…
3597	0.282500	0.173333	0.886943
3598	0.310000	0.963333	0.884442
3599	0.908333	0.601667	0.281641

Table 7. Test data after going through the normalization process

No	pH	TDS	Light Intensity
0	0.556667	0.458333	0.40550
1	0.399167	0.140000	0.44850
2	0.196667	0.711667	0.41475
3	0.554167	0.545000	0.58200
…	…	…	…
14397	0.307500	0.408333	0.79875
14398	0.408333	0.696667	0.43200
14399	0.983333	0.645000	0.06625

The next stage is modeling which is the stage for implementing algorithms for the data processing process. In this study, an algorithm with a deep neural network with 5 layers in it was formed with the aim of classifying multiclass labels. The first modeling process carried out is the creation of a checkpoint function with the help of the TensorFlow library to store temporary training results in personal storage, as well as a training stop function with the help of the TensorFlow library which is used to dismiss training when the validation data loss value has reached the specified value.

The next process is the formation of a deep neural network model with the help of the TensorFlow library. The architecture of the model with five layers consists of one input layer with 128 neurons and a relu activation function to enter training data, three hidden layers which each have neurons totaling 64, 32, and 16 with a relu activation function, and one output layer with 18 neurons according to the number of labels you want to classify and the softmax activation function.

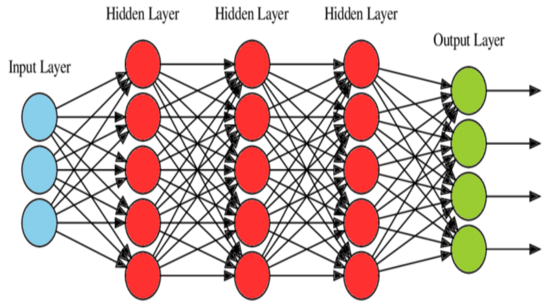
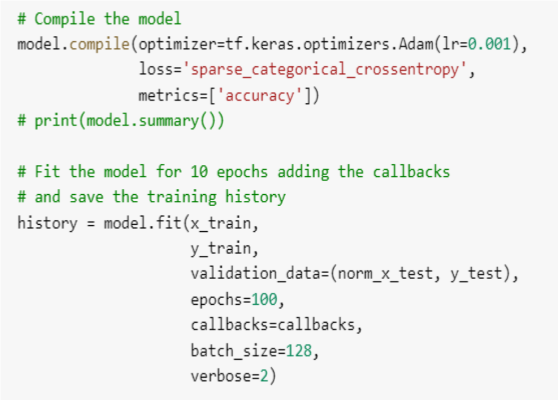


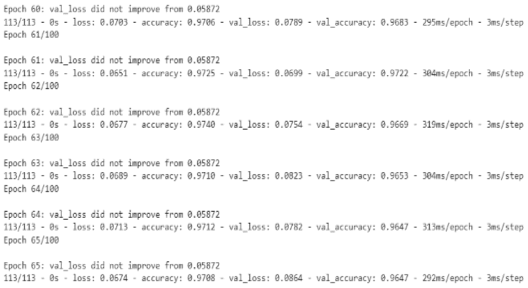
Figure 2. Visualization of deep neural network architecture with customized hyperparameter tuning

The next stage is the formation of a compiler to run the model using three parameters, namely adam optimizer, sparse categorical crossentropy loss, and the use of accuracy values as metrics. The next process is to execute the training process with 100 epochs, callbacks, batch sizes totaling 128, verbose worth 2, as well as training data and customized validation data.



```python
# Compile the model
model.compile(optimizer=tf.keras.optimizers.Adam(lr=0.001),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
# print(model.summary())

# Fit the model for 10 epochs adding the callbacks
# and save the training history
history = model.fit(x_train,
                    y_train,
                    validation_data=(norm_x_test, y_test),
                    epochs=100,
                    callbacks=callbacks,
                    batch_size=128,
                    verbose=2)
```

Figure 3. The process of compiling and fitting the model against the data



```
Epoch 60: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0703 - accuracy: 0.9706 - val_loss: 0.0789 - val_accuracy: 0.9683 - 295ms/epoch - 3ms/step
Epoch 61/100

Epoch 61: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0651 - accuracy: 0.9725 - val_loss: 0.0699 - val_accuracy: 0.9722 - 304ms/epoch - 3ms/step
Epoch 62/100

Epoch 62: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0677 - accuracy: 0.9740 - val_loss: 0.0754 - val_accuracy: 0.9669 - 319ms/epoch - 3ms/step
Epoch 63/100

Epoch 63: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0689 - accuracy: 0.9710 - val_loss: 0.0823 - val_accuracy: 0.9653 - 304ms/epoch - 3ms/step
Epoch 64/100

Epoch 64: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0713 - accuracy: 0.9712 - val_loss: 0.0782 - val_accuracy: 0.9647 - 313ms/epoch - 3ms/step
Epoch 65/100

Epoch 65: val_loss did not improve from 0.05872
113/113 - 0s - loss: 0.0674 - accuracy: 0.9708 - val_loss: 0.0864 - val_accuracy: 0.9647 - 292ms/epoch - 3ms/step
```

Figure 4. Data training process

After the training process is complete and the value of accuracy and validation can be observed, the next process is to form a form of visualization that can describe the accuracy of the training and validation process.

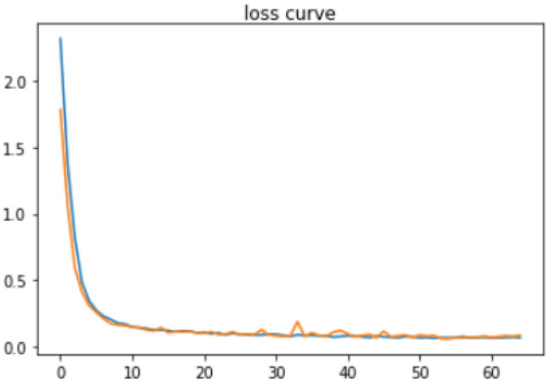


Figure 5. Loss curve after data training process

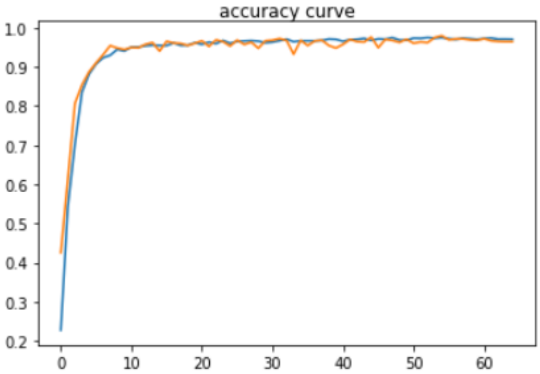


Figure 6. Accuracy curve after data training process

The next stage is an evaluation carried out to see the quality of the model that has been formed. The first stage that is carried out is to look at the loss value and accuracy of the model that has been trained and validated. The results in the training data showed a loss value of 7.19% and an accuracy value of 97.02%, while the test data showed a loss value of 8.64% and an accuracy value of 96.47%.

Training Scores

```python
[ ] # evaluate the model
    # train_loss, train_acc = model.evaluate(norm_x_train, y_train, verbose=2)
    # test_lost, test_acc = model.evaluate(norm_x_test, y_test, verbose=2)
    # # loss, acc = model.evaluate(X_test, y_test, verbose=2)
    # print('Train %s: %.3f' % model.metrics_names[1])
    # print('Test accuracy: %.3f' % test_acc)
    train_scores = model.evaluate(norm_x_train, y_train, verbose=0)
    print("%s: %.2f%%" % (model.metrics_names[0], train_scores[0]*100))
    print("%s: %.2f%%" % (model.metrics_names[1], train_scores[1]*100))

    loss: 7.19%
    accuracy: 97.02%
```

Testing Scores

```python
[ ] test_scores = model.evaluate(norm_x_test, y_test, verbose=0)
    print("%s: %.2f%%" % (model.metrics_names[0], test_scores[0]*100))
    print("%s: %.2f%%" % (model.metrics_names[1], test_scores[1]*100))

    loss: 8.64%
    accuracy: 96.47%
```

Figure 7. Score and training data validation and test data

Next is the stage of testing directly using data taken from sensors integrated with hydroponic plants.

Table 8. TDS, light intensity, pH, and label

No	TDS	Light Intensity	pH	Label
0	379	713	6033	3
1	425	643	6040	3
2	993	659	4716	4
3	1067	834	5982	6
…	…	…	…	…
17997	771	738	4002	4
17998	934	920	6980	13
17999	484	743	4720	16

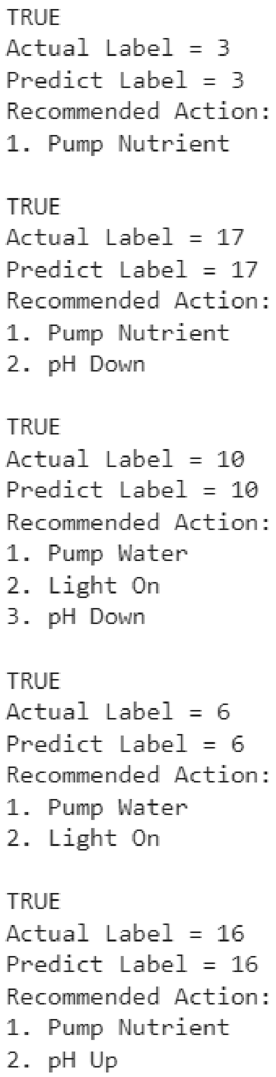
After the data is obtained, the next process is the definition of several action recommendations that farmers can do, of course, the action recommendations that are defined manually are adjusted to the labeling process that has been carried out at the data understanding stage.

```
# action yang akan dilakukan oleh actuator
action = {
    '0' : 'No Action',
    '1' : 'Pump Water',
    '2' : 'Light On',
    '3' : 'Pump Nutrient',
    '4' : 'pH Up',
    '5' : 'pH Down'
}
# recommendation berupa action yg dapat dilakukan
labels = {
    '0' : {
        'recommendation' : [0]
    },
    '1' : {
        'recommendation' : [1]
    },
    '2' : {
        'recommendation' : [2]
    },
```

Figure 8. Defined action recommendations

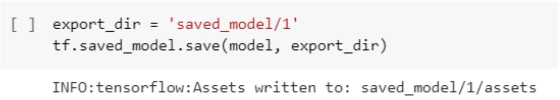
The next process is testing the model with the data that has been taken and the recommendations that have been defined, then the results of the test will be visualized. It can be seen in the picture, from several iterations carried out to test the model, excellent results were obtained.

This indicates that the model formed is fairly accurate to provide recommendations for actions that need to be carried out by farmers of hydroponic plants. After the evaluated model has shown good results, then the next step is the deployment stage by converting the model into a TensorFlow lite form so that the model can be used in IoT tools and Hydromon applications.



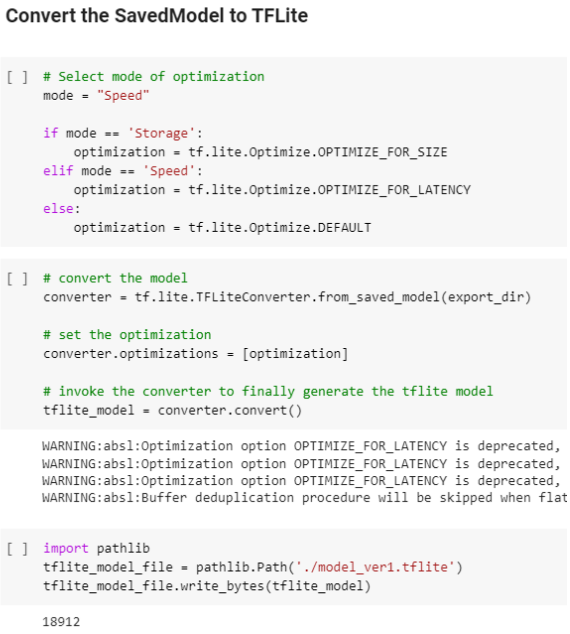
```
TRUE
Actual Label = 3
Predict Label = 3
Recommended Action:
1. Pump Nutrient

TRUE
Actual Label = 17
Predict Label = 17
Recommended Action:
1. Pump Nutrient
2. pH Down

TRUE
Actual Label = 10
Predict Label = 10
Recommended Action:
1. Pump Water
2. Light On
3. pH Down

TRUE
Actual Label = 6
Predict Label = 6
Recommended Action:
1. Pump Water
2. Light On

TRUE
Actual Label = 16
Predict Label = 16
Recommended Action:
1. Pump Nutrient
2. pH Up
```

Figure 9. Visualization of test results



```
[ ] export_dir = 'saved_model/1'
    tf.saved_model.save(model, export_dir)

    INFO:tensorflow:Assets written to: saved_model/1/assets
```

Figure 10. Model storage process



Convert the SavedModel to TFLite

```
[ ]  # Select mode of optimization
     mode = "Speed"

     if mode == 'Storage':
         optimization = tf.lite.Optimize.OPTIMIZE_FOR_SIZE
     elif mode == 'Speed':
         optimization = tf.lite.Optimize.OPTIMIZE_FOR_LATENCY
     else:
         optimization = tf.lite.Optimize.DEFAULT
```

```
[ ]  # convert the model
     converter = tf.lite.TFLiteConverter.from_saved_model(export_dir)

     # set the optimization
     converter.optimizations = [optimization]

     # invoke the converter to finally generate the tflite model
     tflite_model = converter.convert()
```

```
WARNING:absl:Optimization option OPTIMIZE_FOR_LATENCY is deprecated,
WARNING:absl:Optimization option OPTIMIZE_FOR_LATENCY is deprecated,
WARNING:absl:Optimization option OPTIMIZE_FOR_LATENCY is deprecated,
WARNING:absl:Buffer deduplication procedure will be skipped when flat
```

```
[ ]  import pathlib
     tflite_model_file = pathlib.Path('./model_ver1.tflite')
     tflite_model_file.write_bytes(tflite_model)
```

```
18912
```

Figure 11. The process of converting models into TensorFlow lite forms

IV. CONCLUSION

The results of the model development that the author carried out together with the machine learning model development team showed that the developed model was good enough to support the action recommendation system with an accuracy value of 96.47% on the test data. The results of this model will then be placed on IoT tools and Hydromon applications as a recommendation system for actions that farmers need to do based on the condition of the hydroponic plant. In the future, advanced development of the model can be carried out using a more diverse number of layers and hyperparameters, as well as the development that allows the system to provide recommendations for more diverse attributes.

BIBLIOGRAPHY

[1] T. Hariono, C. Putra, and K. A. W. Hasbullah, "Data Acquisition for Monitoring IoT-Based Hydroponic Automation System Using ESP8266," *Newt. Netw. Inf. Technol.*, vol. 1, no. 1, pp. 1–7, 2021, [Online]. Available: https://ejournal.unwaha.ac.id/index.php/newton/article/view/1534.

[2] S. Bhatt, D. Mody, R. Rao, and P. T. Vijayetha, "IMPLEMENTATION OF IoT AND MACHINE LEARNING IN HYDROPONICS : A REVIEW," *J. Emerg. Technol. Innov. Res.*, vol. 5, no. 11, pp. 387–391, 2018.

[3] M. Ridwan and K. M. Sari, "Penerapan IoT dalam Sistem Otomatisasi Kontrol Suhu, Kelembaban, dan Tingkat Keasaman Hidroponik Application of IoT for Automated Controlling System of Temperature, Humidity, and Acidity in Hydroponics," *J. Tek. Pertan. Lampung*, vol. 10, no. 4, pp. 481–487, 2021, [Online]. Available: http://dx.doi.org/10.23960/jtep-l.v10.i4.481-487.

[4] J. Díaz-Ramírez, "Machine Learning and Deep Learning," *Ingeniare*, vol. 29, no. 2, pp. 182–183, 2021, doi: 10.4067/S0718-33052021000200180.

[5] E. Retnoningsih and R. Pramudita, "Mengenal Machine Learning Dengan Teknik Supervised Dan Unsupervised Learning Menggunakan Python," *Bina Insa. Ict J.*, vol. 7, no. 2, p. 156, 2020, doi: 10.51211/biict.v7i2.1422.

[6] A. U. Zailani, A. Perdananto, N. Nurjaya, and S. Sholihin, "Pengenalan Sejak Dini Siswa SMP tentang Machine Learning untuk Klasifikasi Gambar dalam Menghadapi Revolusi 4.0.," *KOMMAS J. Pengabdi. Kpd. Masy.*, vol. 1, no. 1, 2020.

[7] S. Z. Gurbuz, *Deep Neural Network Design for Radar Applications*. London: SciTech Publishing, 2021.

[8] A. Musa, M. Hassan, M. Hamada, and F. Aliyu, "Low-Power Deep Learning Model for Plant Disease Detection for Smart-Hydroponics Using Knowledge Distillation Techniques," *J. Low Power Electron. Appl.*, vol. 12, no. 2, 2022, doi: 10.3390/jlpea12020024.

[9] I. Gridin, *Automated Deep Learning Using Neural Network Intelligence: Develop and Design PyTorch and TensorFlow Models Using Python*. New York: Apress Media, 2022.

[10] M. Mehra, S. Saxena, S. Sankaranarayanan, R. J. Tom, and M. Veeramanikandan, "IoT based hydroponics system using Deep Neural Networks," *Comput. Electron. Agric.*, vol. 155, no. October, pp. 473–486, 2018, doi: 10.1016/j.compag.2018.10.015.

[11] N. Dey, *Intelligent Speech Signal Processing*. Kolkata: Academic Press, 2019.

[12] D. Bau, J. Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 117, no. 48, pp. 30071–30078, 2020, doi: 10.1073/pnas.1907375117.

[13] H. (ed. . Kennedy, *An Historical Atlas of Islam*. Leiden, Boston, Köln: Brill, 2002.

[14] A. Iosifidis and A. Tefas, *Deep Learning for Robot Perception and Cognition*. London: Academic Press, 2022.

[15] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*. New York: Morgan Kaufmann, 2017.

[16] B. Pang, E. Nijkamp, and Y. N. Wu, "Deep Learning With TensorFlow: A Review," *J. Educ. Behav. Stat.*, vol. 45, no. 2, pp. 227–248, 2020, doi: 10.3102/1076998619872761.

[17] T. J. Sheng *et al.*, "An Internet of Things Based Smart Waste Management System Using LoRa and Tensorflow Deep Learning Model," *IEEE Access*, vol. 8, pp. 148793–148811, 2020, doi: 10.1109/ACCESS.2020.3016255.

[18] U. Fadlilah, A. K. Mahamad, and B. Handaga, "The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study," *J. Phys. Conf. Ser.*, vol. 1858, no. 1, 2021, doi: 10.1088/1742-6596/1858/1/012085.