

# Improved Mango Disease Detection Model for Energy-efficient Edge Computing Device

Wen-Tsai Sung,<sup>1</sup> Indra Griha Tofik Isa,<sup>2</sup> and Sung-Jung Hsiao<sup>3\*</sup>

<sup>1</sup>Department of Electrical Engineering, National Chin-Yi University of Technology,  
Zhongshan Rd, Section 2, No. 57, Taichung City 411030, Taiwan

<sup>2</sup>Graduate Institute, Prospective Technology of Electrical Engineering and Computer Science,  
National Chin-Yi University of Technology, Zhongshan Rd, Section 2, No. 57, Taichung City 411030, Taiwan

<sup>3</sup>Department of Information Technology, Takming University of Science and Technology,  
Taipei City 11451, Taiwan

(Received November 19, 2024; accepted March 17, 2025)

**Keywords:** mango disease detection, vision sensing, YOLOv7-Tiny, intelligent agriculture, edge computing

Mango is a fruit that has high economic value for several countries, so it is widely cultivated through technological engineering, including the implementation of artificial intelligence through vision sensing based on object detection methods. In this study, we proposed a detection model by integrating a You-Only-Look-Once version 7 Tiny (YOLOv7-Tiny) detection model and U-Net segmentation to detect plant growth through mango leaf diseases. The dataset development was carried out through image collection from several resources and image augmentation, which resulted in a total of 13004 images consisting of eight classes, namely, anthracnose, bacterial canker, cutting weevil, die back, gall midge, sooty mold, powdery mildew, and normal. The training, testing, and validation data were set to 70, 20, and 10%, respectively. The comparative experiment involved other models, namely, YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, and YOLOv8n. The experimental results showed that the proposed model has outstanding performance compared with the other models with a mean average precision of 90.22%, while precision, recall, and *F1*-score have the percentages of 88.42, 87.93, and 88.17%, respectively. In practical applications, the proposed model has significant results in detecting mango leaf diseases. Moreover, the proposed model has good performance in terms of energy efficiency, which is represented by the model size and inference time generated by the model.

## 1. Introduction

Mango cultivation has high economic value for several countries and high demand in the world market. Several methods have been developed to maintain the supply and demand of mangoes, such as the use of pesticides, counseling and training, agricultural engineering, infrastructure development, and technology implementation. However, one of the main

---

\*Corresponding author: e-mail: [sungjung@gs.takming.edu.tw](mailto:sungjung@gs.takming.edu.tw)  
<https://doi.org/10.18494/SAM5414>

challenges in optimizing mango production is the limited land area. Thus, the alternatives that can be implemented in increasing the productivity of mango cultivation, in terms of both quantity and quality, include the implementation of artificial intelligence (AI)-based technologies. Several AI-based technologies that can be integrated with the agricultural sector include IoT-based monitoring systems, integrated big data, and intelligent video surveillance for precision agriculture.<sup>(1–3)</sup>

Mango plant visualization is a representative indicator of how the growth of mango plants can be optimized. An intelligent model is embedded into vision sensing to perform detection and classification tasks in determining the condition of mango plants, whether they are in optimal condition or not, based on the condition of the leaves, fruit, and tree trunk. The results of intelligent model detection can be recommendations for users or triggers on actuators to carry out certain instructions. Several researchers have developed an intelligent mango detection model based on vision sensing, such as machine vision to detect the fruit count and fruit size of mango plantations. The proposed model is MangoYOLO, which achieves a mean average precision (mAP50) of 98.8%.<sup>(4)</sup> Wu *et al.* developed a mango detection and monitoring system for greenhouses.<sup>(5)</sup> The data sources used were the AWS database and a deep learning model developed to detect pests in mangoes. The experimental results showed an accuracy of 90% and an *F1*-score of 95%.<sup>(5)</sup> Huang and Wu constructed GCS-YOLO4-Tiny, which is a group convolution aimed at detecting the multiple maturity stages of a mango fruit.<sup>(6)</sup> The model modification was carried out by adding a squeeze-and-excitation module to increase accuracy and involve group convolution in order to reduce the computational cost. The evaluation metrics of the model's performance, including *mAP*, recall, *F1*-score, and precision, reached 93.42, 91.00, 90.80, and 90.80%, respectively.<sup>(6)</sup> On the other hand, Roy and Bhaduri developed the real-time growth stage detection model by combining DenseNet and YOLOv4. The evaluation metric of experimental results indicated an *mAP* of 96.20% and an *F1*-score of 93.61%.<sup>(7)</sup>

The above-mentioned works only focus on the performance of the model; however, in the context of a monitoring system based on vision sensing, it is necessary to consider the computational cost of the constructed model, especially when it is implemented in practical applications such as an edge computing system. The edge computing itself is an intelligent computing system that has limitations such as memory storage, but has reliability in data transmission, real-time detection, reduced latency, and increased privacy and security.<sup>(8,9)</sup> Edge computing can increase productivity in the agricultural sector, provide real-time data to the cultivator, and generate integrated and comprehensive data.<sup>(10)</sup> Therefore, the intelligent model to be implemented must have good adaptability in terms of memory consumption, which will ultimately support energy efficiency sustainably.<sup>(11)</sup>

In this study, a mango leaf disease detection model based on YOLOv7-Tiny was developed. This model has compatibility with limited resources and less memory consumption so as to support energy efficiency. This model was integrated with a segmentation technique that increases the precision at the pixel level of each instance object, which will increase model performance to detect mango leaf diseases. The practical application of lightweight mango object detection was implemented on edge computing devices through multiple evaluation metrics including model evaluation metrics, such as accuracy, precision, and recall, and energy-

efficient analysis metrics, such as model size and inference time. The developed device can provide convenient technology in mango cultivation monitoring with low computational cost and high accuracy.

## 2. Materials and Methods

This section consists of several subsections, including (1) the proposed system, which represents the main components integrated into the system; (2) the architecture of the proposed lightweight mango detection model that explains the original architecture of YOLOv7-Tiny, the segmentation technique, and the modification of YOLOv7-Tiny, which implements the segmentation technique; (3) the experimental setup that describes the dataset used in this study, experimental area, and evaluation metrics; and (4) the hardware and software used in this study.

### 2.1 Proposed system

Figure 1 shows the proposed system containing several elements that are connected. The input image in the form of a mango leaf is captured by the vision-sensing device to be processed and transmitted to the edge computing system, which is integrated with the cloud system. The proposed model is integrated into an edge computing system that combines YOLOv7-Tiny and the segmentation method to classify the input image. The classification task was conducted to classify the image source on the basis of eight classes, namely, anthracnose, bacterial canker, cutting weevil, die back, gall midge, sooty mold, powdery mildew, and normal. The proposed system provides real-time plant disease monitoring and diagnosis. With edge computing, the system can process images locally, lowering latency and allowing for rapid actions. Furthermore, the capacity to categorize various diseases and evaluate overall plant health is useful for agricultural cultivators.

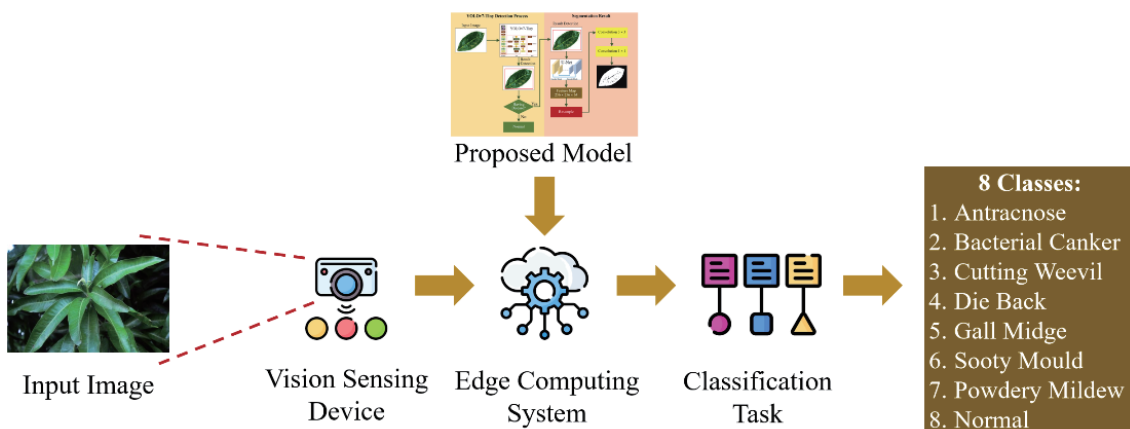


Fig. 1. (Color online) Proposed system.

### 2.2 State-of-the-art YOLOv7-Tiny

The foundation model that will be employed in this study is the state-of-the-art YOLOv7-Tiny as depicted in Fig. 2. The initial You Only Look Once (YOLO) model was designed in 2016, which is a significant breakthrough in the object detection model, providing more accurate and faster detection.<sup>(12)</sup> Significant developments are found in YOLOv4, where the concepts of backbone, neck, and head as the structure of the YOLO architecture are introduced. The subsequent versions continued to undergo modifications and improvements until YOLOv7 was developed, where there is a key innovation with the presence of an extended efficient layer aggregation (E-LAN) network and the integration of several modules in the head structure such as SPPCSP, MCB, and CBL in order to achieve multiscale feature fusion.<sup>(13)</sup> The light version of

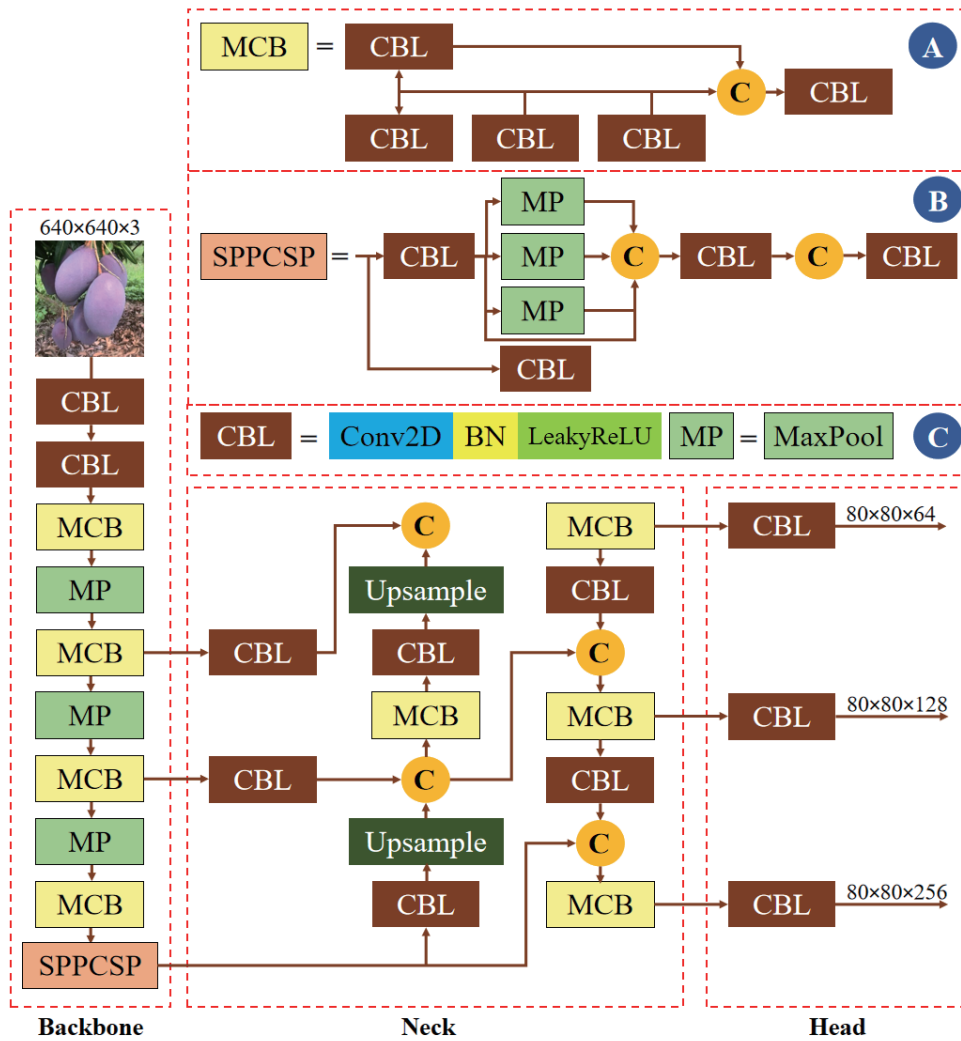


Fig. 2. (Color online) Architecture and detailed modules of YOLOv7-Tiny.<sup>(13)</sup>

YOLOv7 is YOLOv7-Tiny, which has the advantage of faster detection and can adapt to limited memory capacity devices.<sup>(14)</sup>

The structure of YOLOv7-Tiny consists of the backbone, neck, and head network. The backbone network consists of several modules such as CBL, MP, and MCB. The detailed CBL modules are shown in Fig. 2(C), which consists of a convolutional 2D layer, batch normalization, LeakyRelu, and MP that indicates the MaxPool layer. The MCB module has two branches, which are used to gain network efficiency by extracting more features and having the robustness of the network. Several CBL modules are involved within the MCB module, which has the final CBL module processed as superimposed, which is represented in Fig. 2(B). In the first backbone network, there are two CBL modules followed by the combination of MCB and MP. The last module in the backbone structure is SPPCSP, which is connected to CBL at the neck structure. The neck network consists of CBL, MCB, and Upsample modules. In addition, concatenation integrates several modules. The second and third MCB modules in the backbone network are connected to the CBL modules at the neck structure, which has a kernel size of 1 and a step size of 1 for each CBL. The first Upsampling module is used to balance the feature image after feature extraction operation is conducted by the CBL module; then, it is concatenated and forwarded into the MCB module. On the other hand, the head network is an integration between the MCB in the neck network and the CBL in the head network. The three CBL modules have a kernel size of 3 and a step size of 1, where the top, middle, and bottom of the CBL modules have dimensions of  $80 \times 80 \times 64$ ,  $80 \times 80 \times 128$ , and  $80 \times 80 \times 256$ , respectively. CBL modules aim to handle feature extraction and integrate the number of channels generated by MCB modules that have extracted the network's features in different sizes.

### 2.3 Overview of U-Net segmentation method

U-Net segmentation refers to the U-shape architecture that consists of two parts, namely, an encoder and a decoder, as depicted in Fig. 3 on the left and right sides, respectively. The encoder acts as the contracting path, whereas the decoder acts as the expanding path.<sup>(15)</sup> The feature extraction is conducted in the encoder part of the input image. There are several layers such as a series of convolutional layers followed by max pooling layers. Convolutional layers are used to identify the image's features and patterns, and max pooling layers can reduce the feature maps' spatial dimensions. Therefore, the network focuses only on the important information features. Moreover, the decoder part handles the upsampling of the feature map and reconstructs the image segmentation. The decoder consists of upsampling layers that aim to enhance the feature map's spatial dimensions and convolutional layers that handle the details of the output image by refining the segmentation. U-Net has three main features, namely, symmetric architecture, skip connection, and the efficiency number of parameters.<sup>(16)</sup> Symmetric architecture is represented by having mirror images of the encoder and decoder parts to enable capturing the context and details of a fine grain. Skip connection is the implementation of concatenation between the feature maps in the encoder part and the upsampling feature map in the decoder part. The efficiency number of parameters indicates the ability of the architecture to employ a limited number of datasets because it has an efficient and small network compared with other related architectures.

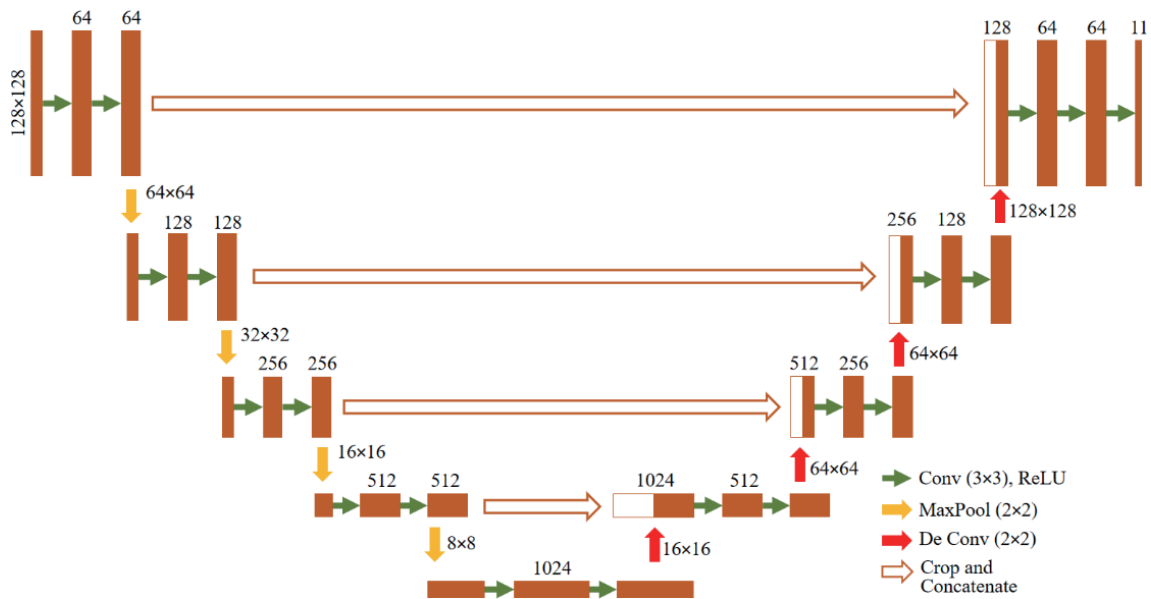


Fig. 3. (Color online) U-Net segmentation architecture.<sup>(15)</sup>

## 2.4 Architecture of improved mango disease detection model

Figure 4 shows the improved mango disease detection model, which is the integration of the lightweight YOLOv7-Tiny model with the U-Net segmentation method. In the initial stage, the detection is carried out using YOLOv7-Tiny to determine whether the mango leaf has a disease or not. If the leaf does not have a disease, then the leaf is considered normal; otherwise, the segmentation will be carried out using the U-Net model. The encoder and decoder mechanisms in U-Net are used for feature extraction and upsampling in the segmentation stage. After passing the segmentation stage, the next step is conducting the feature map operation with dimensions of row, column, and channel of  $256 \times 256 \times 64$ , respectively. The feature map can capture more abstract and semantic information of the processed image to enhance the accuracy of the model in performing the disease classification. Then, resampling is applied to align the processed image after feature map processing. This aims to ensure that both input and output feature maps have proportional dimensions. The two convolutional blocks with sizes of  $3 \times 3$  and  $1 \times 1$  are adopted to refine the feature extraction based on the resampling stage. Therefore, the model has outstanding performance in defining in detail the diseases of the mango plant, especially mango leaf diseases.

## 2.5 Dataset

The data sources used in developing the dataset come from public and private repositories. Public repositories come from Google Image and Kaggle, whereas private repositories are home-made datasets captured by a Canon PowerShot S95 digital camera from mango plantations

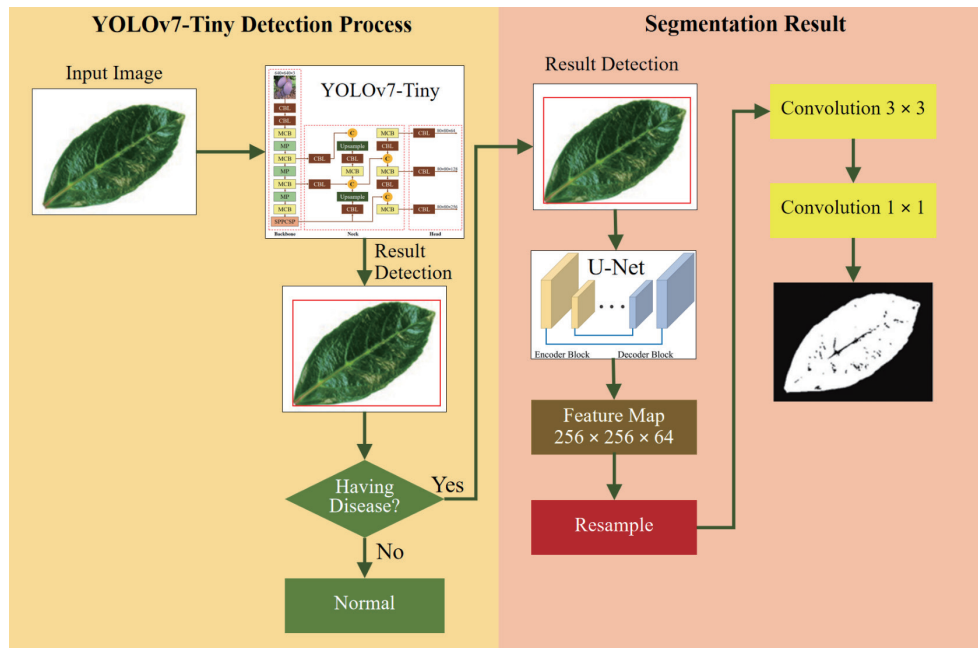


Fig. 4. (Color online) Architecture of proposed model.

located at coordinates of  $24^{\circ} 08' 48.0''$  (north–south position) and  $120^{\circ} 43' 54.3''$  (east–west position) of Taiping District, Taichung City, Taiwan. The initial number of leaf images collected was 4735 from eight classes. Then, image augmentation was performed to enhance the size of the dataset by utilizing rotating, flipping, and scaling techniques. After the image augmentation, the mango leaf dataset developed in this study contains a total of 13004 images divided into training, testing, and validation data with percentages of 70, 20, and 10%, respectively. This dataset consists of eight classes, including seven classes of mango leaves with anthracnose, bacterial canker, cutting weevil, die back, gall midge, sooty mold, and powdery mildew, and one class of healthy mango leaves (normal). Specifically, Table 1 shows a detailed dataset employed in this study.

The next stage is image annotation where labeling is conducted for the instance object in the entire image. The purpose of image annotation is to allow the model to recognize and locate the instance object on the basis of the predicted class. In this study, the tool used in image annotation is LabelImg.

## 2.6 Evaluation metrics

Several evaluation metrics are used to measure the performance of the proposed model, such as precision, recall,  $mAP$ , and  $F1$ -score. Moreover, several indicators are considered to evaluate the energy efficiency of the model, which requires data on inference time, memory consumption, inference energy, storage consumption, and peak power consumption. *Precision* is measured to



Table 1  
Detailed dataset employed in this study.

Class Name	Training	Testing	Validation	Total
Anthracoese	1183	338	169	1691
Bacterial Canker	1274	364	182	1821
Cutting Weevil	1092	312	156	1560
Die Back	1001	286	143	1430
Gall Midge	910	260	130	1300
Sooty Mold	1274	364	182	1821
Powdery Mildew	1183	338	169	1691
Normal	1183	338	169	1691

determine the consistency of the positive prediction that has the correct value, and it is represented as

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}. \quad (1)$$

*Recall* is used to evaluate how the model can recognize the relevant target object within an image or a video. It is measured by dividing true positive by the summation of true positive and false negative.<sup>(17)</sup>

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}. \quad (2)$$

*mAP* is used for evaluating the performance of object identification models and *F1-score* is the harmonious value of precision and recall. *mAP* and *F1-score* are formulated as

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k, \quad (3)$$

$$F1 - score = \frac{2 \times (precision \times recall)}{(precision + recall)}. \quad (4)$$

## 2.7 Edge computing configuration

The edge computing system implemented in this study consists of three layers, namely, the physical layer, edge layer, and cloud layer. The physical layer consists of a vision-sensing device in the form of an HD camera that detects the condition of the plant. The image visualization of the mango plant's current condition is then transferred to edge computing on the edge layer,



which consists of the network system and the edge computing itself. The proposed intelligent model is embedded in edge computing to conduct the task of detecting the images that have been captured by the vision-sensing device. The data that has been processed on the edge layer is then transmitted to the cloud layer as part of the storage mechanism, which is ultimately accessed by the user as real-time data on the condition of the mango plant. Figure 5 shows the configuration of the edge computing system.

## 2.8 Hardware and software

In this study, there are several devices including hardware and software that support the implementation of the proposed model in detecting the conditions of the mango plant in real time. Table 2 shows the hardware and software used in this study.

## 3. Results and Discussion

The training model stage is performed on a computer with the following specifications: CPU is Intel® Core™ i5-13500 2.50 Gigahertz, GPU is NVIDIA GeForce RTX4070, RAM is 32.0 GB, and GPU engine is NVIDIA CUDA with 5888 cores. There are several experiments to evaluate the proposed model performance such as those involving model evaluation performance, comparative experiment, implementation in practical applications, and the analysis of energy efficiency.

### 3.1 Model evaluation performance

The evaluation metrics indicate the performance of the model based on class name. Table 3 shows the performance evaluation model using four evaluation metrics, namely, precision,

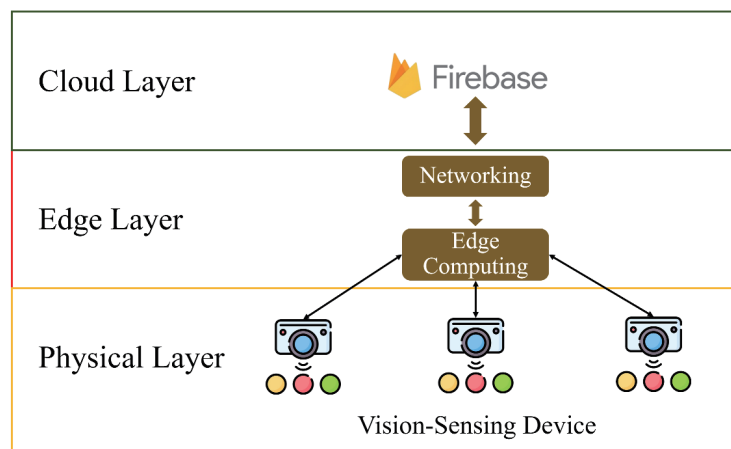


Fig. 5. (Color online) Configuration of edge computing system.

Table 2  
(Color online) Hardware and software in the system.

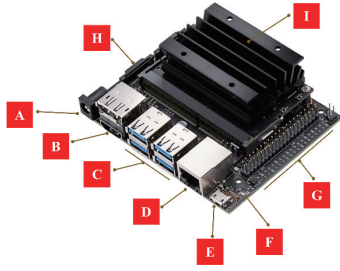
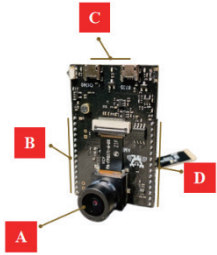

Name	Detailed Description	Visualization
NVIDIA Jetson Nano	The version of NVIDIA Jetson Nano is B01 with a 128-core Maxwell™ GPU and a Quad-core ARM A57 CPU @ 1.43 GHz. There are 9 pins as follows: pin A is the DC input, pin B is the monitor port, pin 3 is the USB connector, pin 4 is the Ethernet port, pin 5 is the USB port, pin 6 is the LED indicator, pin 7 is the 40-pin header, pin 8 is the camera connector, and pin 9 is the main storage.	
Ameba HD Camera	The Ameba HD camera employs an AI camera of the AMB82-MINI type and is integrated with Realtek RTL8735BDM SoC. There are 4 pins as follows: pin A is the HD wide range of the Ameba camera, pins B and D represent the device pinout, and pin C is the USB port.	
Firebase Cloud System	Firebase is provided by Google Cloud that has three features, namely, cloud firestore, cloud storage, and cloud function. Cloud firestore is the database NoSQL platform, which has flexibility in data modeling. Cloud storage is the secure and robust storing and serving platform, while cloud function is the serverless code execution platform.	

Table 3  
Proposed model performance based on class.

Class Name	Precision (%)	Recall (%)	AP (%)	F1-score (%)
Anthracnose	82.15	84.62	86.64	83.37
Bacterial Canker	93.52	94.74	95.13	94.13
Cutting Weevil	86.47	87.52	88.39	86.99
Die Back	83.85	80.17	82.24	81.97
Gall Midge	91.28	90.06	95.41	90.67
Sooty Mold	93.22	90.18	94.85	91.67
Powdery Mildew	92.14	93.65	93.36	92.89
Normal	84.75	82.48	85.72	83.60

recall, *AP*, and *F1*-score. In the proposed model, we can see how the model performs on the eight classes evaluated. Overall, the four evaluation metrics indicate a percentage greater than 80%. In the precision metric, four classes have a percentage greater than 90%, namely, bacterial canker, gall midge, sooty mold, and powdery mildew with percentages of 93.52, 91.28, 93.22, and 92.14%, respectively. In the *AP* evaluation metric, there are four classes with a percentage less than 90%, namely, anthracnose, cutting weevil, die back, and normal with percentages of 86.64,

88.39, 82.24, and 85.75%, respectively. The lowest value is found in die back with an  $AP$  percentage of 82.24% and the highest value is found in gall midge with an  $AP$  percentage of 95.41%. In the  $F1$ -score metric, the highest percentage is in bacterial canker with an  $AP$  percentage of 94.13%. Figure 6 shows the losses, including the training and validation losses. The training and validation losses are indicated by blue and red lines, respectively. Initially, both losses have different starting points, where the initial training loss is 18.26 and the validation loss is 19.14; in both models, there is a significant decrease in the first epoch of 2500. In epoch 2501 to 10000, there is a gradual decrease in loss, where at the end of the training epoch, a training loss of 0.048 and a validation loss of 0.052 are obtained.

The model performance was also evaluated using a comparative performance experiment between the original YOLOv7-Tiny and the proposed model. The results are shown in Table 4, where two models are evaluated. The first is the original model of YOLOv7-Tiny and the other is the proposed model, which is the integration of YOLOv7-Tiny and the segmentation technique. The selection of this comparative model was carried out to determine the significance of the detection performance of the proposed model, which is the integration of YOLOv7-Tiny and the segmentation technique, compared with the original model of YOLOv7-Tiny. Overall, both models show optimal results including the metrics of precision, recall,  $mAP$ , and  $F1$ -score.

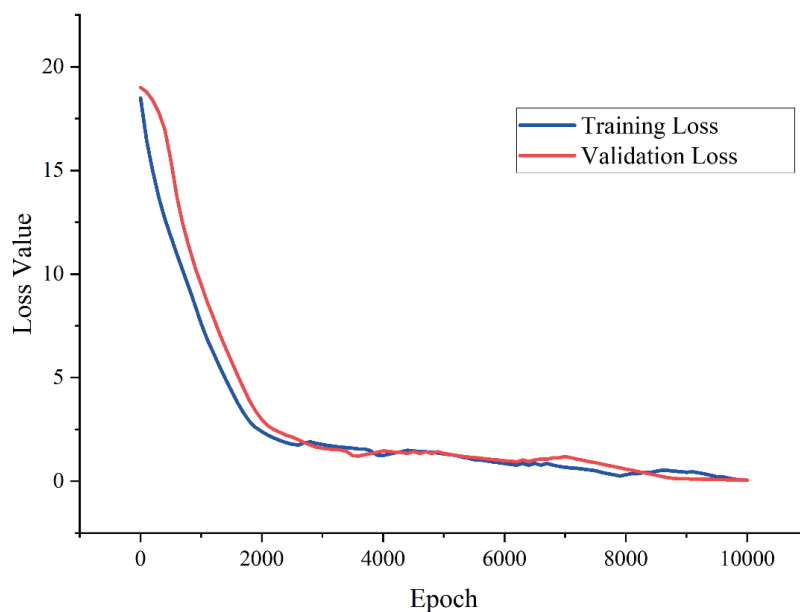


Fig. 6. (Color online) Training and validation losses of the proposed model.

Table 4

Performance comparison of the proposed model and original YOLOv7-Tiny.

Model Name	Precision (%)	Recall (%)	$mAP$ (%)	$F1$ -score (%)
YOLOv7-Tiny	82.25	81.54	84.43	81.89
Proposed Model	88.42	87.93	90.22	88.17

However, the proposed model indicates a higher performance than the original YOLOv7-Tiny. In terms of *mAP* percentage, the proposed model shows an increase of 7.50%, the original YOLOv7-Tiny achieves only 84.43%, and the proposed model shows 90.22%. Likewise with the metrics of precision and recall where in the proposed model there is an increase in percentage, gains of 88.42 and 87.93% are obtained. This increases the *F1*-score where the original YOLOv7-Tiny has a percentage of 81.89%, while the proposed model has 8.28% higher than the original YOLOv7-Tiny.

### 3.2 Comparative experiment

To evaluate the proposed model performance, a comparative experiment was conducted, which involves other tiny models such as YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, and YOLOv8n. The selection of the proposed model is based on its characteristics, namely, its use of fewer layers and inclusion in the category of tiny model versions. Therefore, the comparative experiment conducted provides the objective result. The configuration of the training epoch is 10000 with a learning rate of 0.01. Table 4 highlights the significance between the original model of YOLOv7-Tiny and the proposed model, while Table 5 shows the experimental results of the proposed model compared with those of the other models. In the precision metric, all four models, namely, YOLOv4-Tiny, YOLOv7-Tiny, YOLOv8n, and the proposed model, have percentages above 80%, whereas YOLOv5n has the lowest percentage of 78.65%. By contrast, the recall metric indicates that all the models have percentages above 80%, where the highest percentage is found in the proposed model. Overall, in the *mAP* metric, all the models have percentages above 80%. There are four models with *mAP* percentages between 80 and 90%, namely, YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, and YOLOv8n with *mAP* percentages of 84.06, 82.75, 84.43, and 88.35%, respectively. The highest *mAP* percentage is found in the proposed model with a percentage of 90.22%. Figure 7 shows the *mAP* results of the five models compared during 10000 training epochs. Overall, all five models experienced a significant increase in *mAP* during the first epoch of 2000. Then, in the next epochs, there were a slowdown in the increase and a tendency to fluctuate. This can be seen in YOLOv4-Tiny, which is indicated by the black line where there is a fluctuation during epoch 3000 to 4000. On the other hand, the proposed model, which is represented by the purple line, indicates a more stable epoch increase; in other words, the *mAP* produced during the training epoch is more stable.

Table 5  
Performance characteristics of the proposed model and state-of-the-art models.

Model Name	Precision (%)	Recall (%)	<i>mAP</i> (%)	F1-score (%)
YOLOv4-Tiny	83.12	82.64	84.06	82.88
YOLOv5n	78.65	80.33	82.75	79.48
YOLOv7-Tiny	82.25	81.54	84.43	81.89
YOLOv8n	85.63	83.27	88.35	84.43
Proposed Model	88.42	87.93	90.22	88.17

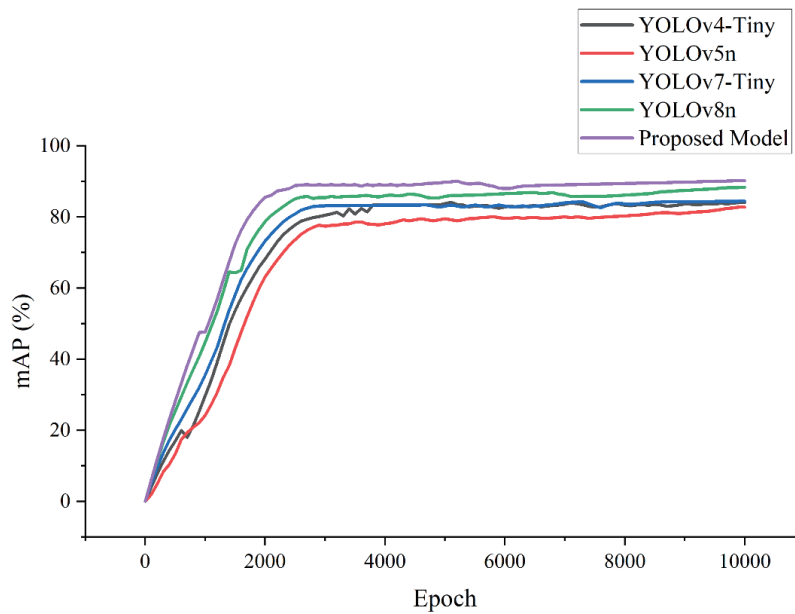
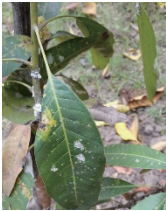




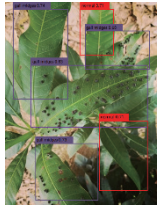


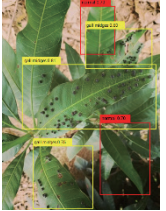


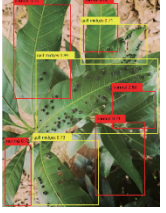
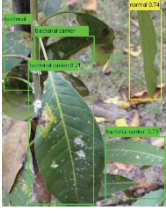

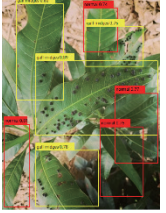
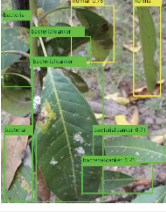

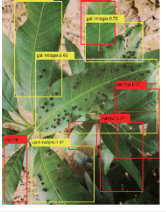


Fig. 7. (Color online) mAP results of five trained models.

### 3.3 Implementation in practical applications

From the previous experiment, the proposed model, which is the integration of the YOLOv7-Tiny and U-Net segmentation model, has outstanding performance compared with other tiny or nanosize YOLO series. The next experiment was conducted by implementing the proposed model in practical applications. As mentioned above that the proposed model will be implemented in an edge computing system, the NVIDIA Jetson Nano B01 version is utilized for testing in real-world data to determine its capacity and flexibility as the edge computing system. In the practical experiment, it can be seen how the performance model is based on different models including YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, YOLOv8n, and the proposed model. Table 6 shows the experimental results of the real-world data where there are three input images, namely, image A, image B, and image C. The three images have different characteristics; image A represents a mango leaf with bacterial cancer, image B a mango leaf with anthracnose, and image C a mango leaf with gall midge. All five models can perform detection tasks on mango leaf diseases, but some models exhibit misclassification detection or cannot detect all the instance objects, especially the tiny objects. The detection task performance of the proposed model indicates a significant result compared with that of the other models; that is, the proposed model can detect more leaf conditions. For example, in image C, the proposed model can correctly detect the leaves with gall midge and normal leaves. In addition, some tiny objects can be detected using the proposed model precisely and accurately. When YOLOv4-Tiny and YOLOv5n are compared, some object classes cannot be detected by both models. Furthermore, in image A, the proposed model can detect more image leaf disease conditions with bacterial canker classes than YOLOv8n; that is, the proposed model can detect six diseased leaves,

Table 6  
(color online) Real-world experimental results.

Model Name	Image A	Image B	Image C
Input Image			
YOLOv4-Tiny			
YOLOv5n			
YOLOv7-Tiny			
YOLOv8n			
Proposed Model			

whereas YOLOv8n can detect only four. In addition, there is only one normal leaf detected in YOLOv8n, while there are two normal leaves detected in the proposed model, which also represents the actual condition. YOLOv4-Tiny and YOLOv5n only detect two diseased leaves



and one normal leaf, whereas YOLOv7-Tiny has more leaves detected, namely, three diseased leaves and one normal leaf. Therefore, on the basis of the experimental results in practical applications, the proposed model has optimal performance in terms of detection task in the real-world data compared with the other models. This indicates that the proposed model is highly accurate and precise in detecting mango diseases.

### 3.4 Analysis of energy efficiency

Energy efficiency in the context of implementing the proposed model on an edge computing system is associated with how the generated parameters are represented by the model size. The larger the size of the model, the higher the memory consumption or computational cost on the edge computing system. Moreover, the smaller the size of the model, the lower the computational cost. In addition, the efficiency of the model can be seen from how fast the model performs inference, as represented by the inference time. Figure 8 shows the performance of the model in terms of how memory is consumed and how the detection performance is based on the five models compared, namely, YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, YOLOv8n, and the proposed model. Figure 8(a) shows the model size with the x-axis being the model name and the y-axis being the model size in megabytes (MB). The five models have sizes below 7 MB, and the model with the largest size is YOLOv8n (6.4 MB). There are three models with sizes below 6 MB, namely, YOLOv5n (5.8 MB), YOLOv7-Tiny (5.1 MB), and the proposed model (5.2 MB). Of the five models, YOLOv7-Tiny has the smallest size with only 5.1 MB. However, the size of the proposed model is not significantly different from that of YOLOv7-Tiny, which is 0.1 MB larger with a size of 5.2 MB. Figure 8(b) shows the inference time of each model, where the y-axis indicates the inference time in millisecond (ms). YOLOv8n has the longest inference time of 34 ms, but the other four models have inference times below 30 ms, namely, YOLOv4-Tiny, YOLOv5n, YOLOv7-Tiny, and the proposed model. Of the five models, the proposed model has

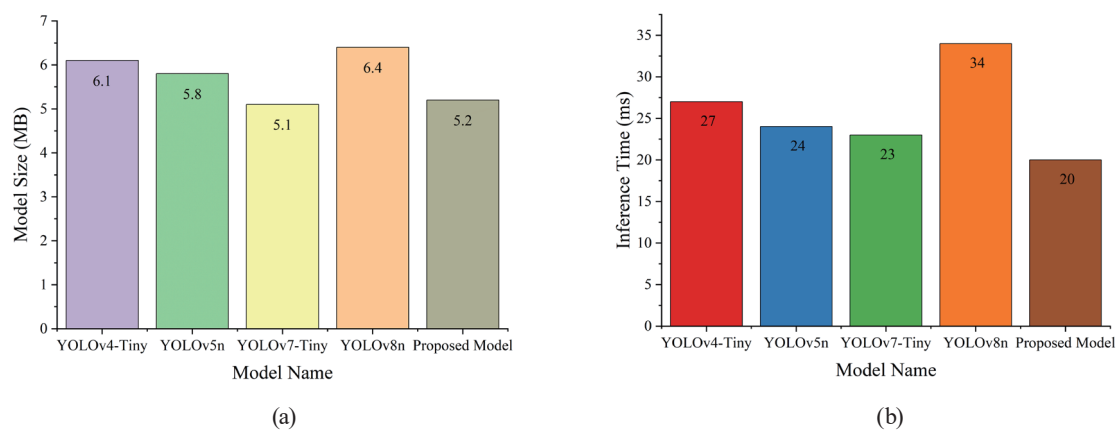


Fig. 8. (Color online) (a) Model size result and (b) inference time of each model.



the shortest inference time of 20 ms. This result indicates that the size and inference time of the proposed model are more efficient and optimal for use in the edge computing system. This can be seen in the size of the proposed model, which is included in the low category but has the most optimal detection performance in the context of inference time. Therefore, the proposed model is recommended to be implemented in an edge computing system that has good energy efficiency with fast and accurate detection performance.

#### 4. Conclusions

In this study, the proposed model has been developed to detect disease conditions in mango leaves by integrating the state-of-the-art YOLOv7-Tiny and U-Net segmentation. The proposed model has achieved the outstanding performance based on comprehensive experiments, which is indicated by the optimal accuracy detection performance and energy efficiency. In practical applications where the model is embedded into an edge computing device, the proposed model can detect objects in detail in the classification of mango leaf diseases. In terms of energy efficiency, which is represented by model size and inference time, the proposed model indicates the optimal performance. In the future, the structure of the model can be modified in order to achieve the optimal detection task, such as implementing the attention module mechanism or optimizing the block module in the structure of the model. In addition, the proposed model is suggested for detecting not only mango diseases, but also fruit ripeness or appropriate pest management. The comparative experiment can be conducted with the recent YOLO versions, such as YOLOv11 and YOLOv12.

#### Acknowledgments

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

#### References

- 1 A. Kocian and L. Incrocci: *Stats* **3** (2020) 3. <https://doi.org/10.3390/stats3030018>
- 2 C. Zhang: *Appl. Math. Nonlinear Sci.* **9** (2024) 1. <https://doi.org/10.2478/amns-2024-0845>
- 3 G. Maier, A. Albanese, M. Ciavotta, N. Ciulli, S. Giordano, E. Giusti, A. Salvatore, and G. Schembra: *Comput. Networks.* **243** (2024) 110248. <https://doi.org/10.1016/j.comnet.2024.110248>
- 4 C. Neupane, K. B. Walsh, R. Goulart, and A. Koirala: *Sensors* **24** (2024) 5593. <https://doi.org/10.3390/s24175593>
- 5 G. H. Wu, N. F. Huang, Y. H. Huang, P. C. Chan, and S. A. Bhat: *IEEE Trans. Instrum. Meas.* **73** (2024) 5021618. <https://doi.org/10.1109/TIM.2024.3403191>
- 6 M. L. Huang and Y. S. Wu: *Math. Biosci. Eng.* **20** (2023) 1. <https://doi.org/10.3934/mbe.2023011>
- 7 A. M. Roy and J. Bhaduri: *Comput. Electron. Agric.* **193** (2022) 106694. <https://doi.org/10.1016/j.compag.2022.106694>
- 8 H. Cui: *IEEE Access.* **10** (2022) 4896. <https://doi.org/10.1109/ACCESS.2021.3113723>
- 9 C. G. V. N. Prasad, A. Mallareddy, M. Pounambal, and V. Velayutham: *Int. J. Recent Innov. Trends Comput. Commun.* **10** (2022) 1. <https://doi.org/10.17762/ijritcc.v10i1s.5848>

- 10 P. Joshi, D. Das, V. Udutalapally, and S. C. Misra: Proc. 2021 IEEE Int. Symp. Smart Electronic Systems (iSES 2021) 255–260. <https://doi.org/10.1109/iSES52644.2021.00065>
- 11 J. J. Estrada-López, J. Vázquez-Castillo, A. Castillo-Atoche, E. Osorio-de-la-Rosa, J. Heredia-Lozano, and A. Castillo-Atoche: Energies. **16** (2023) 7. <https://doi.org/10.3390/en16072943>
- 12 J. Redmon, S. Divvala, R. Girshick, and A. Farhadi: in Proc. 2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR, 2016). <https://doi.org/10.1145/3243394.3243692>
- 13 C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao: Proc. 2023 Conf. Computer Vision and Pattern Recognition (CVPR, 2023) 7464–7475. <http://arxiv.org/abs/2207.02696>
- 14 L. Ma, L. Zhao, Z. Wang, J. Zhang, and G. Chen: Agronomy. **13** (2023) 5. <https://doi.org/10.3390/agronomy13051419>.
- 15 O. Ronneberger, P. Fischer, and T. Brox: in Proc. 201518th Int. Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI, 2015) 1–8. [https://doi.org/10.1007/978-3-319-42999-1\\_15](https://doi.org/10.1007/978-3-319-42999-1_15).
- 16 W. Baccouch, S. Oueslati, B. Solaiman, and S. Labidi: in Proc. 2022 Int. Conf. Health and Social Care Information System and Technologies (HCist, 2022) 1089–1096. <https://doi.org/10.1016/j.procs.2023.01.388>
- 17 G. M. Foody: PLoS One. **18** (2023) 1. <http://doi.org/10.1371/journal.pone.0291908>

## About the Authors



**Wen-Tsai Sung** (Member, IEEE) received his M.S. and Ph.D. degrees from the Department of Electrical Engineering, National Central University, Taoyuan City, Taiwan, in 2000 and 2007, respectively. He is a distinguished professor in the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung, Taiwan. Dr. Sung won the 2009 Journal of Medical and Biological Engineering (SCIE index) Best Annual Excellent Paper Award (Ph.D. Dissertation) and the Dragon Thesis Award (Master Thesis) sponsored by Acer Foundation. In October 2023, he was recognized as one of the world's top 2% scientists (Career Impact) (2001–2023) by Mendeley Data, a company owned by Elsevier. He has authored over 100 peer-reviewed journal papers and articles indexed by SCI/SCIE/EI. His research interests include artificial intelligence Internet of Things, intelligent computing and systems, and wireless sensor networks. ([songchen@ncut.edu.tw](mailto:songchen@ncut.edu.tw))



**Indra Griha Tofik Isa** received his B.S. degree in informatics engineering from Pelita Bangsa University in 2011 and his M.S. degree in computer science from STMIK LIKMI Indonesia in 2014. He is currently a lecturer at Sriwijaya State Polytechnic and is continuing his Ph.D. studies at National Chin-Yi University of Technology (NCUT) in the fields of electrical engineering and computer science. His research interests are in deep learning, neural network applications, data science, image processing, and object detection.



**Sung-Jung Hsiao** is an associate professor in the Department of Information Technology, Takming University of Science and Technology. He received his B.S. degree in electrical engineering from National Taipei University of Technology, Taiwan, in 1996, his M.S. degree in computer science and information engineering from National Central University, Taiwan in 2003, and his Ph.D. degree in electrical engineering from National Taipei University of Technology, Taiwan in 2014. He has work experience in research and design at Acer Universal Computer Co., Mitsubishi, and First International Computer. ([sungjung@gs.takming.edu.tw](mailto:sungjung@gs.takming.edu.tw))