Indoor Scene Recognition Using ARM-based MobileNets Architectures

Wei-Lung Mao, Sung-Hua Chen, Yu-Tang Huang, Yao-Teng Yang
Graduate School of Engineering Science and Technology and Department of Electrical Engineering
National Yunlin University of Science and Technology 123 University Road, Section 3, Yunlin, Taiwan
wlmao@yuntech.edu.tw, chenhua@yuntech.edu.tw, M10912055@yuntech.edu.tw, M11012017@yuntech.edu.tw

Po-Heng Chou
Research Center for Information Technology Innovation (CITI), Academia Sinica
d00942015@ntu.edu.tw

***Abstract*—** The rapid development of science and technology has improved the quality of people life. In recent years, the use of microcontrollers has increased due to the rise of edge computing. Based on low cost, low power consumption, and high stability, the controller can be widely used in various fields. In this research, an ARM-based platform is applied with a camera module to perform image recognition tasks. The MQTT protocol is realized to transmit the image recognition results. The MobileNets models are developed with X-CUBE-AI tool to perform transfer learning on indoor scene datasets. The verification results are obtained by training MobileNetV1 and MobileNetV2 structures. The proposed image system indeed achieves the average accuracies of 67.2% and 71.6% for MobileNetV1 and MobileNetV2, respectively.

***Keywords*— deep learning, transfer learning, MobileNets, ARM system, MQTT, image recognition.**

I. INTRODUCTION (*HEADING 1*)

Convolutional neural networks are already everywhere the field of computer vision since the popularity of AlexNet. In recent years, the neural network has been continuously evolving, but many models only focus on improving the accuracy, while ignoring the speed and size of the network to make the network more efficient. In real life, many identification tasks need to be Applied on devices or platforms with limited computation, such as autonomous driving, robots, etc. However, many models only focus on scale and ignore speed. This research attempts to use the image recognition neural network algorithm to classify the target category. It is hoped that the model size will meet the storage unit of the microcontroller, and at the same time, it will have certain accuracy and low latency conditions. The team of Lisong Ou et al. [4] used transfer learning combined with MobileNets to identify potato leaf diseases. The preprocessing such as data enhancement and graphic scaling is carried out, which is helpful for the recognition efficiency and robustness of the system. Regarding the dataset, the team of Venkatesh et al. [3] used MobileNets to train on multiple varieties of strawberries and cherries, and fine-tuned the model to achieve better accuracy. For the solution of deploying Artificial Intelligence on microcontrollers, the team of Butt Usman Ali [5] tested and validated using all available compilers and obtained their respective memory usage, inference time, and deviation. In this research, a microcontroller is used with a camera module to perform image recognition tasks, and a network module is used to realize the communication application of the Internet of Things. Among them, the neural network algorithms of the MobileNets series are used to perform transfer learning training on indoor scene datasets.

II. SYSTEM ARCHITECTURE

This research uses the STM32H747I-DISCO Discovery Board, which adopts the official STM32H747XIH6 microcontroller from STMicroelectronics. It is a high-performance 32-bit processor with a Cortex-M7 and Cortex-M4 dual core. The board comes with an STLINK-V3E programmer that can be debugged and compiled via a USB to Micro-B connection.

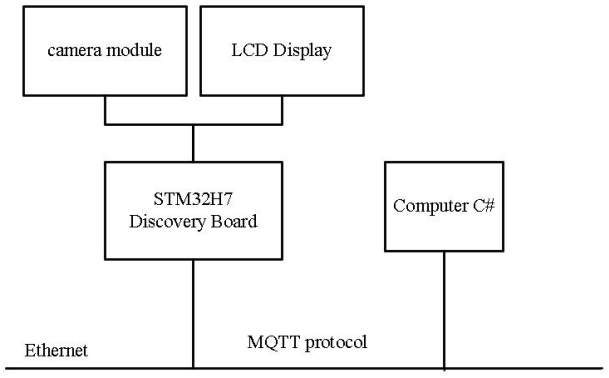


Fig. 1. Image recognition IoT system architecture.

As shown in Figure 1, the board is equipped with a B-CAMS-OMV camera module that comes with an OV5640 camera and can be used as an expansion board for various third-party cameras, or as a camera sub-board for STM32 microcontroller applications in computer vision. The

LAN8742A is a low-power 10BASE-T/100BASE-TX physical layer transceiver with variable I/O voltage, compliant with IEEE 802.3 and 802.3u standards. B-LCD40-DSI1 uses a 4-inch WVGA (800*480) TFT color LCD display, supports single-point touch and two-point touch; uses DSI (Display Serial Interface) interface to provide STM32 display solutions. The STM32H7 series supports the SD2.0 protocol and uses SDHC (High Capacity SD Memory Card) cards with a capacity between 2GB and 32GB.

A. MIT Indoor Scenes

The MIT Indoor Scenes dataset [7] provides 67 indoor categories with a total of 15,620 images. Indoor scene recognition is a challenging problem in high-level vision, and this study try to use 18 categories for image recognition. The detailed image categories are shown in Figure 2.

Fig. 2. Schematic diagram of MIT indoor scene dataset categories.

B. Model Deployment

The pre-trained models of MobileNetV1 and MobileNetV2 are used to perform transfer learning training on the target data, and the TensorFlow model is converted to the TensorFlow Lite model for process use. Eventually one of these formats will be converted to a TFLite model using the TFLite converter. The process of model conversion is shown in Figure 3.

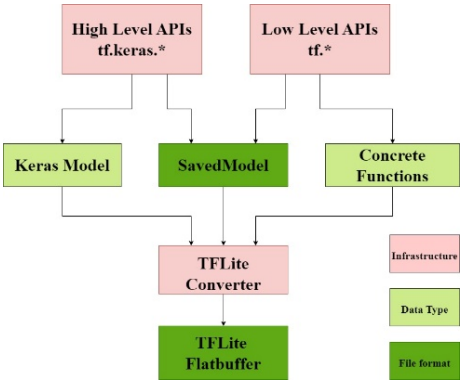


Fig. 3. TFLite converter diagram [8]

Since the STM32 microcontroller cannot directly use the weight model of the TFLite data format, the X-CUBE-AI extension package of STM32cubeMx will be used in the process of building the model to convert the generated TFLite weight model into C language. The operation process of the X-CUBE-AI tool is shown in Figure 4.

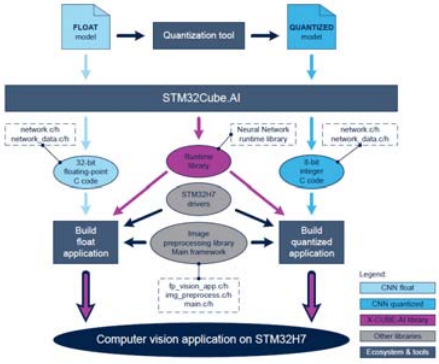


Fig. 4. Schematic diagram of the operation process of X-CUBE-AI tool[6].

III. DEEP LEARNING IMAGE DETECTION ALGORITHM

A. MobileNetV1

The MobileNetV1 network architecture[1] uses a large number of deep separable convolutions for feature extraction, which greatly reduces the overall multiplication and addition calculations. Depthwise separable convolutions all use batch normalization layers and linear rectification functions, which help slow down gradient disappearance and speed up convergence. The MobileNetV1 structure uses a large number of depthwise separable convolutions, as defined in Figure 5. All layers are followed by a batch normalization layer and Rectified Linear Unit, as shown in Figure 6.

MobileNetV1

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
5× Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

Fig. 5. MobileNetV1 neural network architecture [1].

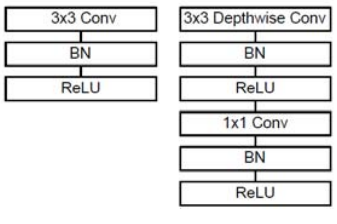


Fig. 6. Comparison of (a) standard convolution and (b) depthwise separable convolution.

In order to compare the total number of operations of depth convolution and standard convolution, the calculation formulas of each convolution are listed below, as shown in equations (1), (2), (3), and (4). Among them, D_K represents the length and width of the convolution kernel, D_F represents the length and width of the input image, M is the number of input channels, and N is the number of output channels, and it is assumed that the number of output channels N is 10.

Convolution calculation amount:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \quad (1)$$

Depthwise convolution calculation amount:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F \quad (2)$$

Depthwise separable convolution computation:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F \quad (3)$$

The ratio of depthwise separable convolution to convolution computation:

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K \cdot D_K} \quad (4)$$

1) Width multiplier: For some special cases, such as embedded devices, there are limitations in memory size and computing power, and the model must be made smaller and faster. To cater for these special cases, the authors introduce a parameter α called the width multiplier. The role of the width multiplier α is to uniformly thin the number of input channels M and the number of output channels N of each layer, reducing the amount of calculation and parameters by about the square of α, and the range of α is between (0,1].

The calculation amount of depth separable convolution adding width multiplier:

$$D_K \cdot D_K \cdot \alpha M \cdot D_F \cdot D_F + \alpha M \cdot \alpha N \cdot D_F \cdot D_F \quad (5)$$

2) Resolution Multiplier: The resolution multiplier ρ is the hyperparameter of the second thinning neural network, which is applied to the resolution of the input image.

Depth separable convolution adds the calculation amount of width multiplier and resolution multiplier:

$$D_K \cdot D_K \cdot \alpha M \cdot \rho D_F \cdot \rho D_F + \alpha M \cdot \alpha N \cdot \rho D_F \cdot \rho D_F \quad (6)$$

B. MobileNetV2

The architecture of MobileNetV2[2] is based on the Inverted Residuals structure, and an additional connection is added between the bottleneck layers, so that the output of the bottleneck layer is added to its input, and the feature information of the low-dimensional part is increased. The neural network architecture of MobileNetV2 is shown in Figure 7.

MobileNetV2

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Fig. 7. MobileNetV2 neural network architecture. [2]

1) Manifold of Interest: In order to obtain important feature information and reduce the computational load of the neural network, the intuitive way is to put the interest manifold into a low-dimensional space and reduce the number of channels used, which can also make the convolutional neural network also followed by a decrease. But in reality, the convolutional neural network will use a nonlinear function for activation, which will cause a large number of low-dimensional feature information to be lost. Therefore, for the low-dimensional feature extraction part, a linear method will be used instead of nonlinear.

2) Linear Bottlenecks Layer: In MobileNetV2, a linear bottleneck layer is used to extract low-dimensional features of the manifold of interest. First, in the low-dimensional part, the 1×1 convolution kernel and expansion coefficient are used to control the total number of channels, the low-dimensional space is increased to the high-dimensional space, and the nonlinear ReLU function is used to increase the complexity of the system. Then, the depth separable convolution is used for convolution operation, while reducing the total amount of calculation; finally, the projection layer of 1×1 convolution kernel is used to project the high-dimensional partial features to the low-dimensional space. Form a state where the input layer and the output layer are both low-dimensional and the middle layer is high-dimensional, which is called the bottleneck layer.

3) Inverted Residuals: MobileNetV2 connects a shortcut between the input and output of the bottleneck layer. This structure is called Residuals. This shortcut skips multiple linear transformations and nonlinear outputs, providing information about the original input features of the bottleneck layer. This advantage can improve the state that the gradient cannot be reflowed, and save the space used by the memory.

IV. SOFTWARE ARCHITECTURE

A. Image Augmentation Library

This study uses seven data enhancement methods for MIT indoor scene datasets (1) Gaussian blur, (2) dynamic blur, (3) affine transformation, (4) flip, (5) contrast, (6) saturation, (7) Brightness, where the probability of Gaussian blur and motion blur is adjusted to 30%, and the probability of flipping is adjusted to 50%. The rest of the methods do not adjust the

probability, and set various parameters to perform image transformation in the target range, so that Data is more diverse. The following data augmentation on a single random indoor scene dataset image is performed 25 times, and the results are shown in Figure 8.

Fig. 8 Tv studio performs 25 data enhancement effects.

B. Transfer Learning

Transfer learning is a method of training models, which can be used to process tasks of other models based on the training of the pre-trained model, and apply the learning of the old field to the part of the new field based on the similarity of the problem. Among them, in the pre-training models MobileNetV1 and MobileNetV2, different width multipliers are used. MobileNetV1 uses α of 0.25 times; MobileNetV2 uses α of 0.35 times, and uses the same input image resolution of 128×128. The freezing operation can be interpreted as setting the weights to be non-trainable, and the weights of the pre-trained model itself will not be updated during the training process. The time to use fine-tuning is that the model has reached a state of convergence for new data, and by unfreezing all or part of the basic pre-trained model, the entire model can be relearned at a very low learning rate. In this experiment, the fine-tuning learning rate is set to 1e-5. This approach may bring optimization to the model, or it may cause the model to overfit quickly.

C. STM32 program flow

After the initialization is completed, enter the LCD homepage and wait for the Wake up button to be pressed. If the button is not pressed after 2000ms, the image recognition process will be carried out. During the process, the category probabilities judged by the neural network algorithm are sorted by the bubble sorting method, and the category with the highest probability is output to the LCD display, and at the same time, the identified category is published through the MQTT protocol, so that subscribers can get the category by subscribing to the topic Message; If the Wake up key is pressed within the specified time, it will enter the LCD menu, and press the Wake up button after inserting the MicroSD card to start the board verification process, and output various performance indicators after the verification is completed.

D. STM32 MQTT Client

This study uses the STM32 microcontroller as the publisher of the MQTT client, and publishes the results of image recognition through the MQTT protocol. Whenever an image recognition is completed, the STM32 microcontroller sends a topic message, so that all subscribers connected to the local Mosquitto broker can receive the recognition result message by subscribing to this topic. In addition, in the connection part, the LwIP protocol stack has a keep-alive mechanism, and a timer needs to be set to continuously send keep-alive messages to ensure that the network can be used normally. In this study, a keep-alive message is sent every 10 milliseconds, and the connection part is separated from the image recognition to ensure that the recognition process is not affected during the process. But this operation will sacrifice part of the recognition speed.

E. C# program interface on computer

This study uses the M2MQTT library in the computer C# program to develop the MQTT communication protocol of the Internet of Things. The architecture based on the MQTT communication protocol can be divided into three parts: Subscriber, Publisher and Broker. In this study, the anonymous parameter is enabled, and the connection can be made without authentication, and the connection port is set to 1883. In the publish message section, publish the topic by selecting parameters such as the host IP address, topic, and message to be sent, and subscribe to the topic to obtain the message content. Subscribe to the message part, the subscriber can also subscribe through the specified host IP and topic name, and when the publisher publishes the relevant topic, the subscriber can receive the message content. Users can enter topics through the Subscribe Topic field, and use the Subscribe button to perform subscription operations. The Publisher Topic field can be used to publish the topic, fill in the corresponding topic, write the release message through the Publisher Text below, and click the publish button to publish the topic. The blank column below is the topic message column received by the subscriber, and the received topic message can be displayed, as shown in Figure 9.

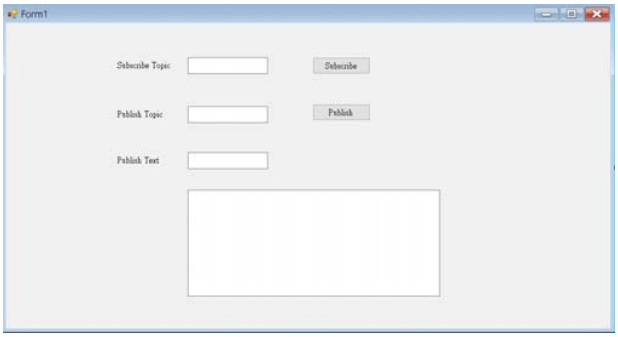


Fig. 9 MQTT communication C# graphic user interface.

V. EXPERIMENTAL RESULTS

This section is divided into two parts. (1) The models are trained in this research and describes the training results in detail; (2) Real-time image recognition, which combines the MQTT communication protocol in the process and publishes the identified categories to Mosquitto broker, which provides all subscribers connected to this server to obtain category information in real time. This study uses 1878 origin images and then using data enhancement methods to enlarge the divided training set to 13146 images, and perform subsequent training on the updated training set. The 70% of total images are used as the training set, 20% as the validation, and 10 % as the testing. The experimental configuration is shown in Figure 10 and the parameters and weights of the MobileNets model used are shown in Table I.

Fig. 10 The proposed image recognition system.

TABLE I. MOBILENETS MODEL PARAMETERS AND WEIGHT SIZES

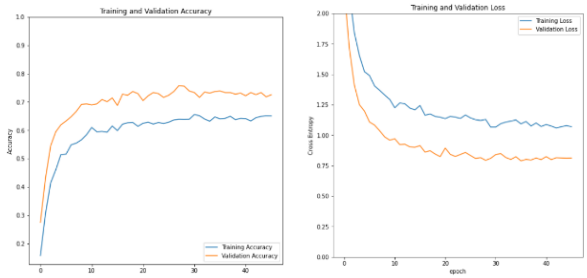
dataset	MobileNetV1		MobileNetV2		Flash used by the weight model	
	α	resolution	α	resolution	V1	V2
MIT Indoor Scene	0.25	128×128	0.35	128×128	292KiB	541KiB

A. MIT Indoor Scene Dataset Training Results

1) MobileNetV1 Training Results: In this section, using the MobileNetV1 neural network to train our indoor scene data set, using a width multiplier with an alpha coefficient of 0.25 for training. The number of iterations of the training process is set to 100, the process is added with an early termination function, and 10 times of Patience is set to monitor the loss function of the verification set. The training parameters are shown in Table II, and the obtained training results are shown in Figure 11.

TABLE II. TRAINING PARAMETER SETTINGS

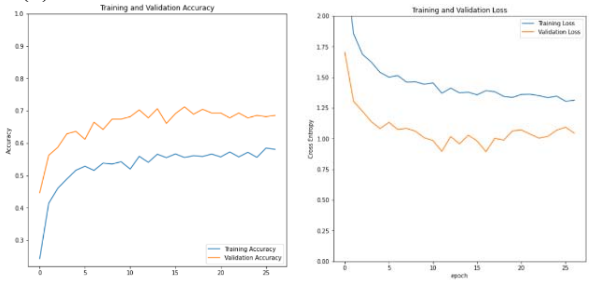
Patience	Iterations	Learning rate
10	100	0.001



(a) (b)

Fig. 11 MobileNetV1 (a)accuracy and (b)loss function.

2) MobileNetV2 Training Results: In this section, using the MobileNetV2 neural network to train our indoor scene dataset, using a width multiplier with an alpha coefficient of 0.35 for training. The number of iterations of the training process is set to 100, the process is added with an early termination function, and 10 times of Patience is set to monitor the loss function of the verification set. The loss functions of the training set and test set are 1.305 and 0.896, respectively. These results are shown in Figure 12(a) and 12(b).



(a) (b)

Fig. 12 MobileNetV2 (a)accuracy and (b)loss function.

B. Indoor Scene Recognition Results

1) MobileNetV1: From the recognition results, it can be seen that the highest accuracy rate is obtained in the corridor and indoor swimming pool categories, and the worst effect is obtained in the game room category.

2) MobileNetV2: From the identification results, it can be seen that the highest accuracy rate is obtained in the bowling category, followed by the corridor and the indoor swimming pool. The game room category also had the worst accuracy. The average precision of MobileNetV2 has increased by about 5.6%, as shown in Table IV.

TABLE III. IMAGE RECOGNITION PERFORMANCE OF MOBILENETS NETWORK

Dataset	MobileNetV1		MobileNetV2	
	Interface (ms)	Fps	Interface (ms)	Fps

Dataset	MobileNetV1		MobileNetV2	
MIT Indoor Scene	70	14.1	127	7.8

Fig. 13 Schematic diagram of indoor scene image recognition.

TABLE IV. INDOOR SCENE MODEL IMAGE RECOGNITION RESULTS

Classes	MobileNetV1		
	TP	FP	Precision
bathroom	8	2	80%
bedroom	4	6	40%
bowling	9	1	90%
casino	5	5	50%
Church inside	6	4	60%
cloister	10	0	100%
closet	6	4	60%
Dental office	7	3	70%
Game room	3	7	30%
greenhouse	7	3	70%
Grocery store	7	3	70%
gym	8	2	80%
Inside bus	8	2	80%
Inside subway	6	4	60%
kitchen	6	4	60%
Living room	4	6	40%
Pool inside	10	0	100%
Tv studio	7	3	70%
Average Precision			67.2%
Classes	MobileNetV2		
	TP	**FP**	**Precision**
bathroom	8	2	80%
bedroom	7	3	70%
bowling	10	0	100%
casino	6	4	60%
Church inside	8	2	80%
cloister	9	1	90%
closet	6	4	60%
Dental office	6	4	60%
Game room	3	7	30%
greenhouse	9	1	90%
Grocery store	8	2	80%
gym	7	3	70%
Inside bus	6	4	60%
Inside subway	8	2	80%
kitchen	8	2	80%
Living room	4	6	40%
Pool inside	9	1	90%
Tv studio	7	3	70%
Average Precision			72.8%

VI. CONCLUSION

In this paper, the MobileNets models are applied in Arm-based platform to achieve the indoor scene recognition. The STM32 microcontroller combined with MQTT protocol and image acquisition operations are developed. In order to perform image recognition tasks, two kinds of deep neural networks are tested to find a network system that considers the accuracy, low latency, and classification storage space. The selection of the neural network algorithm is based on MobileNetV1 and MobileNetV2, and a suitable weight model is obtained by adjusting the system hyperparameters. For data augmentation, the richness of the dataset is increased to reduce the impact of noise interference. By using an STM32 microcontroller with a camera and a network module, combined with deep learning neural network algorithms MobileNetV1 and MobileNetV2, a lightweight solution can be obtained. After image recognition verification testing, the average system precision obtained using the MobileNetV1 indoor scene model is 67.2% and 72.8%, respectively.

REFERENCES

[1] H. Andrew, Z. Menglong, C. Bo, K. Dmitry, W. Weijun, W. Tobias, A. Marco and A. Hartwig, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

[2] S. Mark, H. Andrew, Z. Menglon, Z. Andrey and C. Liang-Chieh, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510-4520, 2018.

[3] Venkatesh, Nagaraju Y., Siddhanth H. and Stalin S, "Fine-tuned MobileNet Classifier for Classification of Strawberry and Cherry Fruit Types," IEEE 2021 International Conference on Computer Communication and Informatics (ICCCI), Jan. 2021.

[4] L. Ou and K. Zhu, "Identification Algorithm of Diseased Leaves based on MobileNet Model," IEEE 2022 4th International Conference on Communications, Information System and Computer Engineering (CISCE), May 2022.

[5] B. U. Ali, "On the deployment of Artificial Neural Networks (ANN) in low cost embedded systems," POLITECNICO DI TORINO, Master thesis, July 2021.

[6] Artificial Intelligence (AI) and computer vision function pack for STM32H7 microcontrollers. STMicroelectronics UM2611, https://www.st.com/content/ccc/resource/technical/document/user_manual/group1/a0/53/9b/b7/f9/b5/4a/6f/DM00630755/files/DM00630755.pdf/jcr:content/translations/en.DM00630755.pdf (December 17, 2021)..

[7] Kaggle MIT Indoor Scenes. https://www.kaggle.com/datasets/itsahmad/indoor-scenes-cvpr-2019 (December 10, 2022).

[8] TensorFlow Lite Conversion Tool https://www.tensorflow.org/lite/convert (December 10, 2022).