Smart home Management System with Face Recognition Based on ArcFace Model in Deep Convolutional Neural Network

Thai-Viet Dang [1,*]
[1] Department of Mechatronics Engineering, Hanoi University of Science and Technology, Hanoi, Vietnam
Email: [1] viet.dangthai@hust.edu.vn
*Corresponding Author

Abstract—**In recent years, artificial intelligence has proved its potential in many fields, especially in computer vision. Facial recognition is one of the most essential tasks in the field of computer vision with various prospective applications from academic research to intelligence service. In this paper, we propose an efficient deep learning approach to facial recognition. Our approach utilizes the architecture of ArcFace model based on the backbone MobileNet V2, in deep convolutional neural network (DCNN). Assistive techniques to increase highly distinguishing features in facial recognition. With the supports of the facial authentication combines with hand gestures recognition, users will be able to monitor and control his home through his mobile phone/tablet/PC. Moreover, they communicate with data and connect to smart devices easily through IoT technology. The overall proposed model is 97% of accuracy and a processing speed of 25 FPS. The interface of the smart home demonstrates the successful functions of real-time operations.**

Keywords—Artificial intelligence; Computer vision; Facial recognition; Home security system; Internet of Thing (IoT).

I. INTRODUCTION

In 4th Industrial Revolution, informatic technology and data science have strongly supported to automatic production. The application of this remarkable breakthrough generated myriad of achievement, including smart monitoring systems [1], advanced transportation infrastructure [2], automated financial systems [3] or industrial assembly robot manipulators [4]. Data science includes digital transformation, data collection and data processing. Furthermore, the application of artificial intelligence (AI) and big data processing is gradually becoming popular. AI consists of main applications as following: Natural language processing, computer vision [61, 62], machine learning [63], data processing [64], and optimal decision.

Following this current trend, we exploit a powerful deep learning approach in facial and hand gestures recognition applying for IoT control management systems [27-39]. Local database of identified people supports into both security and privacy. Through surveillance cameras face, recognition requires high accuracy and good processing speed [5, 10, 18, 22-26]. A typical study can include the MTCNN model using three main network layers P-Net, R-Net, O-Net for face detection with very high accuracy (99.6%) [6]. Still, the processing speed is low due to the use of many network layers, and tested on Jetson Nano (quad-core ARM Cortex-A57 CPU, 128-core Maxwell™ GPU) captures low speed (FPS: 4, input image 720x1280 pixels). This makes it challenging to integrate into the system, particularly to embed MTCNN's low-resource devices. Therefore, we suggest ArcFace model based on backbone MobileNetV2 to optimize face recognition. The technique will reduce the load on the convolutional network layers and replacing the Conv2D layer with a depth-wise separable convolution layer. Next, we continue using the MobilenetV2's output as the input to the SSD network (Single-Shot multibox Detection) for face detection. The output shows that this is an optimal model for low-resourced mobile and embedded devices (score: 96%, FPS:22, Jetson Nano: quad-core ARM Cortex-A57 CPU, 128-core Maxwell™ GPU).

By comparing the pre-selected facial features from the image database, the system is able to consider the correction of the person's face with the dataset. A recent study, LBP (Local Binary Pattern), converted the input image to a binary image, then divided the face into blocks and calculated the histogram density per block to give the histogram feature [7]. However, the feature extraction from the histogram could be affected by external factors such as input image quality, lighting, etc. Besides, another advancement, Dlib algorithm correlated with HOG and SVM [8] was utilized. But there was ponderable drawback that the accuracy might reduce if we alter the faces' angle. In addition, a well-known study of the face recognition algorithm that was FaceNet [9] used the triplet loss function to calculate the distance between face vectors. Nevertheless, this algorithm had the disadvantages that the amount of math operations that the computer needs to perform increases exponentially if the volume of input data is increased and the features between the faces being overlapped. To minimize this influence, the researchers apply ArcFace model, with the method of calculating the distance between face vectors, in addition, creating a deviation angle θ and addictive angular margin m that allows to separate the features of face vectors. ArcFace has developed and improved FaceNet with the consequence that the decomposition helps to avoid misidentification when the original data image has some similarities to the photo taken with a direct angle. Therefore, we utilizes the architecture of ArcFace model based on backbone MobileNet V2, in deep neural convolutional network (DCNN). The model will optimize face recognition by getting the input image. Then, we implement the output as a 512-dimensional vector containing the distinct features of the face as input for the KNN classification algorithm to conduct the recognition [10].

Camera captures a person's image in front of the door, then ArcFace model will recognize person's in real-time. Besides, embedded computers Jetson Nano have been extensively applied into facial recognition because of the outstanding performance and hardware support [11]. For embedded computers in the segmentation [43-50], Jetson nano is the superior choice in terms of processing speed, support for many machine learning frameworks such as Tensorflow, Keras, Pytorch, MediaPie, etc.

The results of facial recognition for security are integrated with hand gestures for controlling smart home. Real-time monitoring data is displayed through smart home interface. System model includes IoT interface based on Tkinter GUI library [12]. According to practical examples, obtained face images reach to good detection performance with an accuracy of 92% to 97% and a fast inference speed of 25 FPS. For small datasets and less resources in training model, the model is more efficient and faster than the state-of-the-art models which require larger datasets for training and processing. The research contribution is as follows: Based on the proposed facial recognition based on MobileNet V2 [40-42], the performance of smart home management system is guaranteed. The security ability is enhanced. Moreover, obtained results can apply for intelligent IoT system and smart services [35-39].

The rest of the paper is organized as follows. In Section II, we describe related work in facial recognition of FaceNet and ArcFace model based on the backbone MobileNet V2. In Section III, we briefly introduce our proposed experimental system. In Section IV, practical experiments are conducted to compare with recent state-of-the-art methods. Finally, a conclusion is presented in Section V.

II. RELATED WORK

A. *Mobilenet V2*

In this section, we mainly introduce the core features of the MobileNetV2 to be used, the optimization of the loss function and utilizes the improved model architecture from FaceNet model. MobileNet V2 uses Depthwise Separable Convolutions, and additionally recommends: Linear bottlenecks and Inverted Residual Block (shortcut connections between bottlenecks) [13].

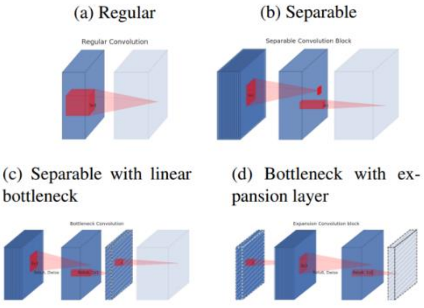
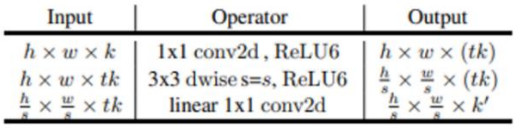

(a) Regular
(b) Separable
Regular Convolution
Separable Convolution Block
(c) Separable with linear bottleneck
(d) Bottleneck with expansion layer
Bottleneck Convolution
Expansion Convolution block

Fig. 1. Model of separable convolutional blocks

MobileNet v2's residual block is the opposite of traditional residual architectures, because the traditional residual architecture has a larger number of channels at the input and output of a block than the intermediate layers. Among layers, there is an inverted residual block, we also use depth-separated convolution transforms to minimize the number of model parameters. The solution helps the MobileNet model to be slightly reduced in size. The detailed structure of a depth-separated convolution of bottleneck with residuals is as follows in Fig. 2.



Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

Fig. 2. Bottleneck residual block converts from k to channel k' with stride and expansion factors.

In the era of mobile networks, the need for lightweight and real-time networking is growing. However, many identity networks cannot meet the real-time requirements due to too many parameters and computations. To solve this problem, the proposed method using the backbone MobileNetV2 achieves superior performance compared with other modern methods in the database of facial expressions and features.

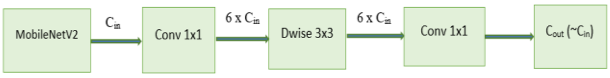


Fig. 3. Infrastructure of backbone MobileNetV2.

Fig. 3 describes MobileNetV2's workflow using 1 x 1 point convolution to extend the input channels. Then use depth convolution for input linear feature extraction and linear convolutional integration to combine output features while reducing the network size. After size reduction, it replaces Relu6 with a linear function to enable the output channel size to match the input. MobileNetV2 will be very useful when applied to SSD to reduce latency and improve processing speed. The SSD model [14] is one of the fast and efficient object detection features with better minimal processing time than YOLO [15-16] and Faster-RCNN [17]. Therefore, the SSD uses the MobilenetV2 network to extract the feature map and adds extra bits after MobilenetV2 to predict the object. Finally, this paper proposes to use MobileNetV2 and SSD for face detection and recognition in security systems.

B. *Facenet Embeddings*

FaceNet uses initiation modules in blocks to reduce the number of trainable parameters [9]. This model takes a 160×160 RGB image and generates a 128-d embedding vector for each image. FaceNet features extraction for face recognition. Use FaceNet to decompose facial features into vectors and then use the triplet loss function to calculate the distance between face vectors in Fig. 4.

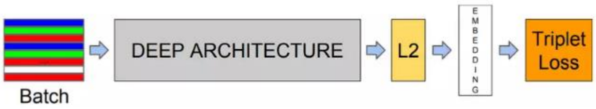


Fig. 4. Face vector extraction process.

Instead of using conventional loss functions that only compare the output value of the network with the actual ground truth of the data, triplet loss introduces a new formula that includes three input values, including the output anchor of the network, positive x_i^p: the photo is the same person with the anchor, positive x_i^p: the photo is not the same person with the anchor.

$$\left\|f(x_i^a) - f\left(x_i^p\right)\right\|_2^2 + \alpha \ \forall\left(f(x_i^a), f\left(x_i^p\right), f(x_i^n)\right) \in T < \left\|f(x_i^a) - f(x_i^n)\right\|_2^2 \qquad (1)$$

where α is the margin (is a distance) between the positive and negative pair, the minimum necessary deviation between the two values, f($x^a{}_i$) is the embedding of $x^a{}_i$. The above formula shows that the distance between two embeddings f($x^a{}_i$) and f($x^p{}_i$) will have to be at least α-value less than the pair f($x^a{}_i$) and f($x^n{}_i$). In other words, after training, the result obtained is that the difference between the two sides of the formula is as large as possible (meaning $x^a{}_i$ get close to $x^p{}_i$).

C. ArcFace model

Deep Convolutional Neural Networks (DCNN) models have proven to be outstanding, being a popular choice to extract facial features. In order to make the recognition more accurate from vectors with facial features, there are two main directions to create a classification model such as using the triplet loss function or using the SoftMax loss function.

In Fig. 5, FaceNet is an encoding process of the convolutional neural network supports to encode the image in 128 dimensions. Then these vectors will be input to the triplet loss function to evaluate the distance between the vectors. However, the triplet loss function has its own shortcomings, because triplet loss is inspired by comparing three samples at a time, so with the increase in the amount of data, the number of triplets will also increase exponentially leading to the number of loops is also significantly increased. Furthermore, the supposed optimal solution when training with triplet loss is semi-hard sample training, which is quite difficult to train effectively.

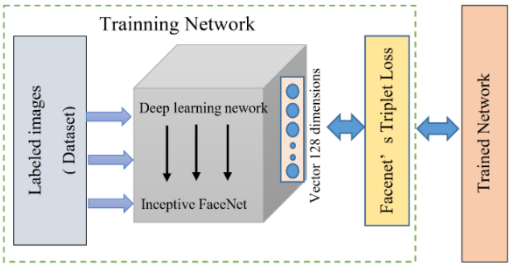
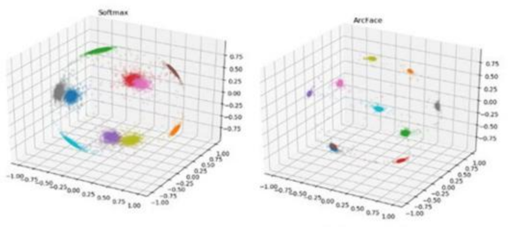


Fig. 5. The architecture of FaceNet.



a. Softmax loss function

b. Arcface loss function

Fig. 7. The classification of image's characteristic in colour image.

The SoftMax loss function is normally used for face recognition problems [18]. The SoftMax loss function is a combination of the cross-entropy loss function and the SoftMax activation function [19]. Using the SoftMax function causes the size of the linear transformation matrix to increase in proportion to the number of classes we want to classify, in equation (2).

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{yi}^T x_i + b_{yi}}}{\sum_{j=1}^{n} e^{W_{yi}^T x_i + b_{yi}}} \qquad (2)$$

where $x_i \in \mathbb{R}^d$ represents the depth feature of sample i, of class y_i. Embedded feature size d is set to 512. $W_j \in \mathbb{R}^d$ represents the jth column of the weight vector W ∈ $\mathbb{R}^{d\,x\,n}$ and $b_j \in \mathbb{R}^n$ is the bias. The batch and numeric class sizes are N and n respectively.

The method is performed by correcting the bias $b_j = 0$. Then, transform the logit to $W_j^T x_i = \|W_j\|\|x_i\|cos\theta_j$, where θ_j is the angle between the ground truth weight vector W_j and the feature vector x_i. Next, the algorithm fixes the individual ground truth weight $\|W_j\| = 1$ by normalizing l_2. Embedding $\|x_i\|$ is done by normalizing l_2 and re-scaling it to s. The feature and weight normalization step makes the predictions depend only on the angle between the feature and the weight. The features are distributed over a hypersphere with a radius of s.

$$L_2 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\,cos\theta_{yi}}}{e^{s\,cos\theta_{yi}} + \sum_{j=1,j\neq y_i}^{n} e^{s\,cos\theta_{yi}}} \qquad (3)$$

Since embedded features are distributed around each feature center on the hypersphere, an additive angular margin penalty m between x_i *and* W_{yi} is added while enhancing intra-class compactness and inter-class differentiation. Since the proposed additive angular margin penalty is equal to the geodetic distance margin penalty in the normalized hypersphere, the method is called ArcFace.

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\,(cos(\theta_{yi}+m))}}{e^{s\,(cos(\theta_{yi}+m))} + \sum_{j=1,j\neq y_i}^{n} e^{s\,(cos(\theta_{yi}+m))}} \qquad (3)$$

Fig. 6 illustrates the process of training a DCNN for face recognition by the ArcFace loss function.

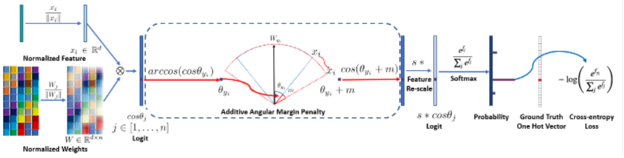


Fig. 6. Training steps of a DCNN for recognition by ArcFace loss function.

The dot product between the feature vector from the DCNN model and the final fully-connected layer is equal to the *cosine* distance of the normalized feature x_i and weight

W, by taking $cos\ \theta_j$ (logit) for each class as $W_j^T x_i$. We do the calculation of $arccos\ \theta_{yi}$ and get the angle between feature x_i and ground truth weight W_{yi}. Next, angular margin penalty m is added to angle θ_{yi}. Then we compute $cos\ (\theta_{yi} + m)$ and multiply all logits by s (scale feature). Logits are then fed into softmax to derive the probability distribution of the labels. Finally, we obtain the ground truth vector (the one-hot label) and the probability, which contributes to the cross-entropy loss.

To demonstrate the effectiveness of ArcFace loss compared with traditional softmax loss, Fig. 7 shows the role of margin m and deviation angle θ_{yi}. In the separation of features. Consequently, the simulation output of the softmax function is shown in Fig. 7a whilst the number of output layers through ArcFace loss trained with no overlap and the distance between layers are clearly shown in Fig. 7b.

The ArcFace model is performed through extensive experiments on many public datasets [20, 21]. The obtained results show better performance than existing approaches [22, 23]. Therefore, ArcFace model will be very useful and effective when applied to face recognition in security systems.

III. PROPOSED SYSTEM

A. Hardware

- Jetson Nano: Jetson Nano 4GB developer AI Artificial Intelligence Development Board including as follows: GPU: 128-core Maxwell; CPU: Quad-core ARM A57@1.43GHZ; Memory: 4 GB 64-bit LPDDR4 25.6 GB/s and Storage: 16GB EMMC.
- Camera: 8MP IMX219 Camera for NVIDIA Jetson Nano 77 Degree 1080P CSI Camera Module.

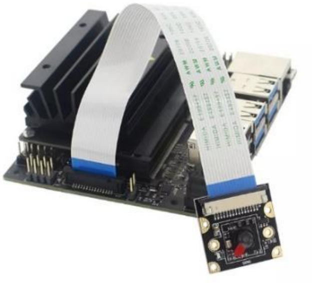


Fig. 8. Jetson nano with 8MP Sony IMX219 77-degree CSI camera

B. Software

- Use PYQT5, a utility tool for designing graphical user interfaces (GUIs). This is an interface written in python programming language for Qt, one of the most popular and influential cross-platform GUI libraries.
- Use some deep learning frameworks: Keras, Tensorflow, MediaPie, etc., which are very easy to use and powerful python libraries and learning models.

C. Facial recognition process

Face recognition process usually consists of four main steps: face detection, face normalization, Face stamp extraction, and matchings, as shown in the block diagram below in Fig. 9.

Face recognition system using Jetson Nano capable of handling multiple video streams [24]. Jetson Nano is connected to the camera module for storing the acquired camera images. An observation device that connects to the Jetson Nano such as a TV or LCD monitor. Jetson Nano also supports Internet access with smart devices [35, 37]: mobile phones, laptops, and desktop computers in Fig. 10.

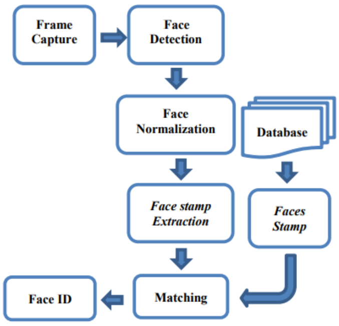


Fig. 9. Block diagram of facial recognition process

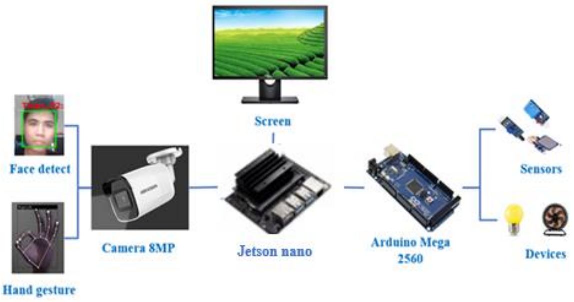


Fig. 10. The structure of smart home management system.

MobileNetV2 is used as the feature map to base the detection from the input images. Then, these feature maps with different scales are enhanced by the FPN, FPN module at the output of the network to improve the performance of the back-end detection network. Facial recognition was SSD detector based on backbone MobileNetV2 and FPN technique (Feature Pyramid Network) in Fig. 11.

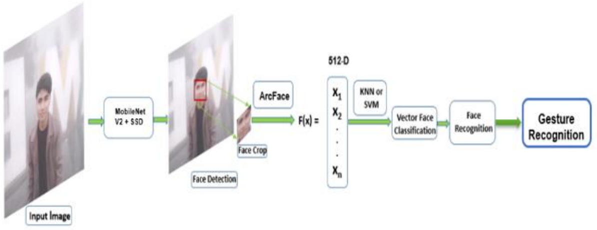


Fig. 11. Face recognition algorithm

Then, ArcFace takes an image of each person's face as input and outputs a vector of 512 numbers representing the most important features of the face. In machine learning, this vector is called embedding. The next step is a classifier which is used to get the distance of the facial features to distinguish between several identities.

D. Hand gestures recognition

Mediapipe library supports the solutions as follows: Face detection, face mesh, hands detection, human pose estimation. Hands detection provides a hand's skeleton from landmark 0 to landmark 20 in Fig. 12. Based on hand gestures, the computer converts the signals into control signals for managing smart home in Fig. 13.

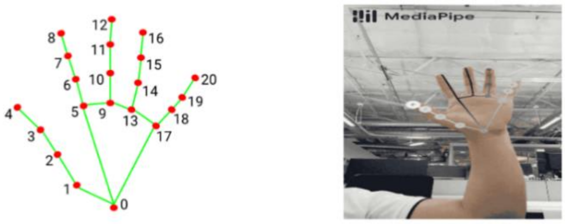


Fig. 12. Hand landmarks based on hand's skeleton

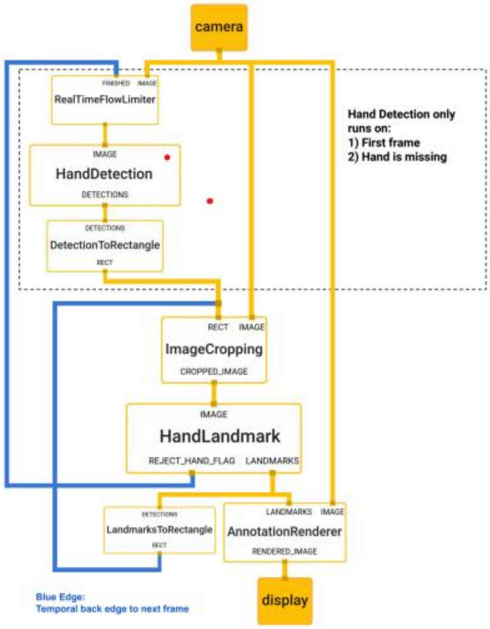


Fig. 13. Hand gestures recognition process

E. System interface

After completing face recognition, combining with hand gestures recognition, automatic smart home management will be launched successfully in Fig. 14.

The system takes input data about the face shown in Fig. 9. Initially, users proceed to enter personal information (Name, ID, …) into the system. The camera detects and scans the faces, recording 6 photos containing the frontal angles, the left and right angles of 10 degrees, the upturned face's chin angle. Then, the facial recognition system start the process, records the features and saves the parameters of the face into the database in the form of a hypersphere with each point being a recognized face. The above process takes place in 30 seconds to 01 minute. For each face added, the algorithm will train the model to recalculate the added features to the node in the hypersphere.

Face recognition algorithm is used to train ArcFace model. Furthermore, hand gestures recognition supports to automatically control smart home. Data base of user's face can be updated. When face ID is successfully detected, the control system will execute function commands.

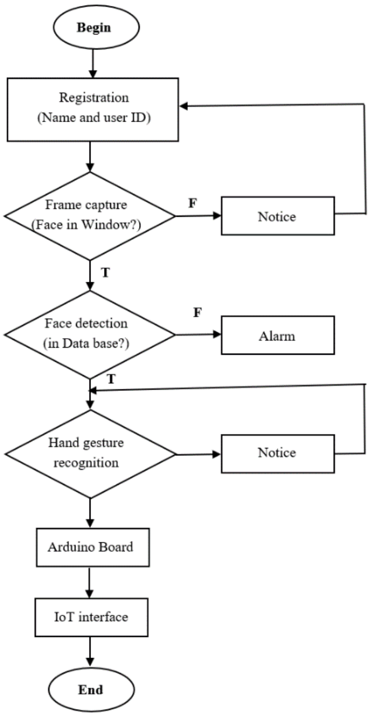
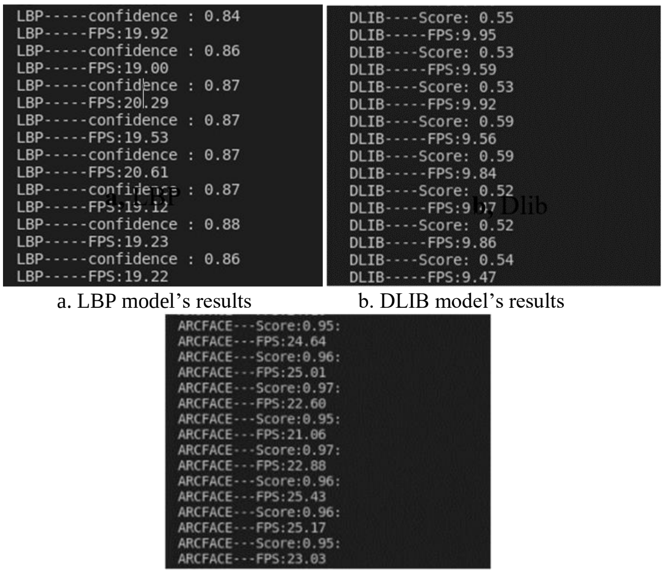


Fig. 14. Flowchart of smart home management

IV. EXPERIMENTAL RESULTS

A. Practical face recognition

The training dataset for the facial authentication model consists of 60 images each of 10 people. Each person has 6 images of face's different angle. Through testing and evaluating on the same data set with 4 frames for each class, we evaluate ArcFace model in comparison with LBP [7] and DLIB [8]. ArcFace shows high accuracy and fast FPS, specifically at 95-97 % and FPS 21-25 in Fig. 15. Table 1 and Fig. 16 present the correlation of the accuracy as well as the process speed.



```
LBP-----confidence : 0.84
LBP-----FPS:19.92
LBP-----confidence : 0.86
LBP-----FPS:19.00
LBP-----confidence : 0.87
LBP-----FPS:20.29
LBP-----confidence : 0.87
LBP-----FPS:19.53
LBP-----confidence : 0.87
LBP-----FPS:20.61
LBP-----confidence : 0.87
LBP-----FPS:19.12
LBP-----confidence : 0.88
LBP-----FPS:19.23
LBP-----confidence : 0.86
LBP-----FPS:19.22
```

a. LBP model's results

```
DLIB----Score: 0.55
DLIB-----FPS:9.95
DLIB----Score: 0.53
DLIB-----FPS:9.59
DLIB----Score: 0.53
DLIB-----FPS:9.92
DLIB----Score: 0.59
DLIB-----FPS:9.56
DLIB----Score: 0.59
DLIB-----FPS:9.84
DLIB----Score: 0.52
DLIB-----FPS:9.[illegible]7
DLIB----Score: 0.52
DLIB-----FPS:9.86
DLIB----Score: 0.54
DLIB-----FPS:9.47
```

b. DLIB model's results

```
ARCFACE---Score:0.95:
ARCFACE---FPS:24.64
ARCFACE---Score:0.96:
ARCFACE---FPS:25.01
ARCFACE---Score:0.97:
ARCFACE---FPS:22.60
ARCFACE---Score:0.95:
ARCFACE---FPS:21.06
ARCFACE---Score:0.97:
ARCFACE---FPS:22.88
ARCFACE---Score:0.96:
ARCFACE---FPS:25.43
ARCFACE---Score:0.96:
ARCFACE---FPS:25.17
ARCFACE---Score:0.95:
ARCFACE---FPS:23.03
```

c. ArcFace model's results

Fig. 15. The results of face recognition of LBP, DLIB and ArceFace models in terms of accuracy and FPS rate.

TABLE 1. FACIAL RECOGNITION TEST RESULTS OF SOME MODELS.

Model	Accuracy (%)	FPS
LBP	84-88	19-21
DLIB	52-59	9-10
Proposed ArcFace	**94-97**	**21-25**

A typical study on the DLIB algorithm in [8] has been implemented on Jetson Nano achieves face detection performance with 8.9 in Table 2. But this algorithm has a disadvantage as changing the viewing angle of the face causes the recognition model to fail the high efficiency.

TABLE 2. REAL TIME FPS RATE OF DLIP MODEL IN [8].

Embedded system	Face recognition/FPS
Jetson Nano	8.9

We evaluate ArcFace in face recognition algorithm with 09 faces with different face angle. The model achieves the accuracy of from 92% to 94% as shown in Fig. 16. The experimental results have been really better than DLIB model.

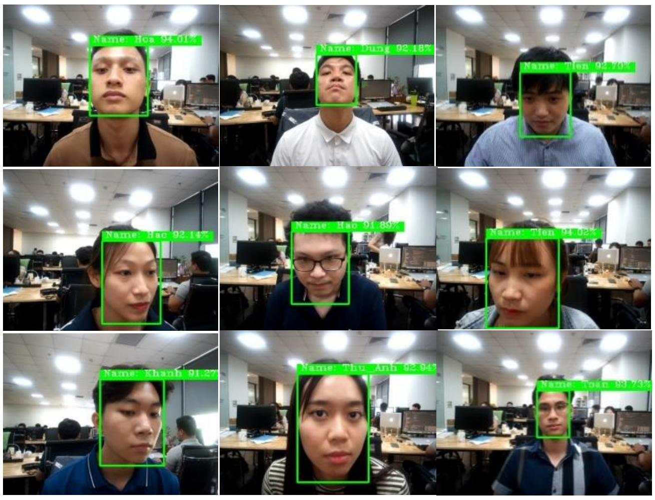


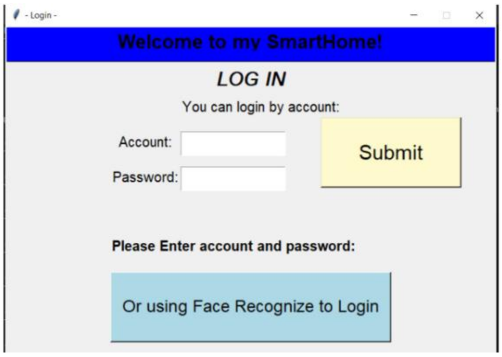
Fig. 16. Face recognition with different viewing angle of face based on ArcFace model

In experimental system, the use ArcFace model based on Mobilenet V2 backbone has the greatest performance when deployed on the Jetson Nano embedded computer. This model has an accuracy of about 92% to 97% and 25 FPS frame rate, much better than previous facial recognition models.

B. *Control management procedure*

The system operation is according to step by step-in real-time activities.

- Step 1: Login the first time by password and account (Fig. 17).
- Step 2: Updating the information of user's face in the main interface (Fig. 18). All steps of facial recognition in Fig. 9 will be performed at here.
- Step 3: Training the hand gesture recognition model in Fig. 19. We add more symbols to relate with smart home functions.
- Step 4: Using the hand gesture's data set to control automatically the smart home's (Fig. 20). All IoT smart devices in the room 1 and room 2 are monitored and controlled.


- Login -

Welcome to my SmartHome!

LOG IN

You can login by account:

Account:

Password:

Please Enter account and password:

Fig. 17. Login interface of smart home

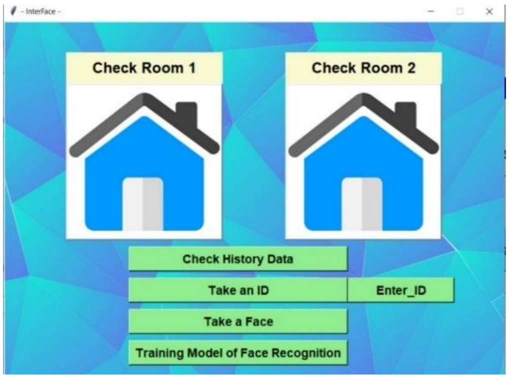


Fig. 18. Main interface of smart home

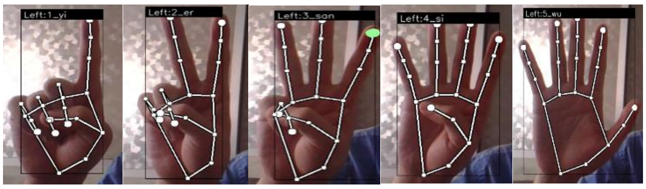
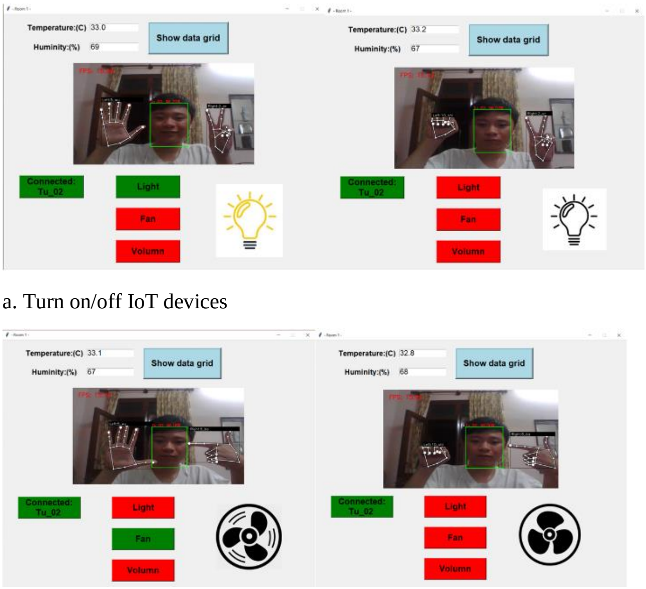


Fig. 19. Some hand gestures recognition.


Temperature:(C) 33.0
Huminity:(%) 69

Temperature:(C) 33.2
Huminity:(%) 67

a. Turn on/off IoT devices

Temperature:(C) 33.1
Huminity:(%) 67

Temperature:(C) 32.8
Huminity:(%) 68

b. Adjust automatically devices

Fig. 20. IoT monitor and control smart home interface.

V. Conclusions

In the paper, model for facial recognition has been proposed for user authentication. Moreover, hand gestures recognition has been supported home control management by using multiple automated and remotely functions. Because of increasing the facial recognition accuracy, ArcFace is improved from FaceNet model. The smart home security is guaranteed based on the high accuracy and fast processing time of facial recognition model. Author utilizes the architecture of ArcFace model based on the backbone Mobilenet V2 using an additive angular margin to achieve significantly facial recognition results. According to practical process, achieved face images reach to good detection performance with an accuracy of 92% to 97% and an inference speed of 25 FPS. For small datasets and less resources in training model, the ArcFace model is more efficient and faster than the previous state-of-the-art models which require larger datasets for training and processing. We will improve the backbone ArcFace to recognize human face with different face angles as well as real-time facial expressions. Moreover, it will strengthen anti-face spoofing methods to increase system security. Finally, obtained results can apply for intelligent IoT system and smart services.

Conflict of Interest

The author declares no conflict of interest

Acknowledgment

This research is funded by Hanoi University of Science and Technology (HUST) under project number T2022-PC-029. The authors express grateful thankfulness to Vietnam-Japan International Institute for Science of Technology (VJIIST) and School of Mechanical Engineering (SME), Hanoi University of Science and Technology (HUST), Vietnam.

References

[1] R. S. Peres, X. Jia, J. Lee, K. Sun, A. W. Colombo, and J. Barata, “Industrial Artificial Intelligence in Industry 4.0 - Systematic Review, Challenges and Outlook,” IEEE Access, vol. 8, pp. 220121-220139, 2020.

[2] V Pagire, S Mate, “Autonomous Vehicle using Computer Vision and LiDAR,” I-manager’s J. Embedded Syst., vol. 9, pp. 7-14, 2021.

[3] J. Huang, J. Chai, and S. Cho, “Deep learning in finance and banking: A literature review and classification,” Frontiers of Business Res. China, vol. 14, pp. 1-24, 2020.

[4] R. Gautam, A. Gedam, and A. A. Mahawadiwar, “Review on Development of Industrial Robotic Arm,” Int. Res. J. Eng. Technol., vol. 4, no. 3, March, 2017.

[5] M. Souded, “People detection, tracking and re-identification through a video camera network,” Université Nice Sophia Antipolis, 2013.

[6] N. Zhang, J. Luo, and W. Gao, "Research on Face Detection Technology Based on MTCNN," 2020 Int. Conf. Comput. Netw., Electron. and Automat. (ICCNEA), pp. 154-158, 2020.

[7] S. Pawar, V. Kithani, S. Ahuja, and S. Sahu, “Local Binary Patterns and Its Application to Facial Image Analysis,” 2011 Int. Conf. Recent Trends Inf. Technol. (ICRTIT), pp. 782-786, 2011.

[8] H. S. Dadi, G. Krishna, and M. Pillutla, “Improved Face Recognition Rate Using HOG Features and SVM Classifier,” IOSR J. Electron. Commun. Eng. (IOSR-JECE), vol 11, no. 4, pp. 33-44, 2016.

[9] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering,” 2015 IEEE Conf. Comput. Vision Pattern Recognit. (CVPR), pp. 815-823, 2015.

[10] A. Wirdiani, P. Hridayami, and A. Widiari, “Face Identification Based on K-Nearest Neighbor,” Scientist J. Inform., vol. 6, no. 2, pp. 150-159, 2019.

[11] A. Süzen, B. Duman, and B. Sen, “Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN,” 2020 Int. Congr. Human-Comput. Interact. Optim. Robot. Appl. (HORA), pp. 1-5, 2020.

[12] S. Kurzadkar, “Hotel management system using Python Tkinler GUI,” Int. J. Comput. Sci. Mobile Comput., vol. 11, no. 1, pp. 204-208, 2022.

[13] Y. Nan, J. Ju, Q. Hua, H. Zhang, and B. Wang, “A-MobileNet: An approach of facial expression recognition,” Alexandria Eng. J., vol. 61, no. 6, pp. 4435-4444, 2021.

[14] Y. Jimin and Z. Wei, “Face Mask Wearing Detection Algorithm Based on Improved YOLO-v4,” Comput. Sci., Sensor, vol. 3263, pp. 21, 2021.

[15] A. Bochkovskiy, C. Y. Wang, and H. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” arXiv 2020, arXiv:2004.10934.

[16] J. Jeong, H. Park, and N. Kwak, “Enhancement of SSD by concatenating feature maps for object detection,” arXiv 2017, arXiv:1705.09587.

[17] Z. Hong et. al., “Local Binary Pattern-Based Adaptive Differential Evolution for Multimodal Optimization Problems,” IEEE Trans. Cyber., vol. 50, no. 7, pp. 3343-3357, 2020.

[18] J. Wang, C. Zheng, X. Yang, and L. Yang, “Enhance Face: Adaptive Weighted SoftMax Loss for Deep Face Recognition,” IEEE Signal Process. Lett., vol. 29, pp. 65-69, 2022.

[19] X. Li, D. Chang, T. Tian, and J. Cao, “Large-Margin Regularized Softmax Cross-Entropy Loss,” IEEE Access, vol. 7, pp. 19572-19578, 2019.

[20] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S. Z. Agedb, “The first manually collected in-the-wild age database,” CVPR Workshop, 2017.

[21] W. Liu, Y. Wen, Z. Yu, and M. Yang, “Large-Margin Softmax Loss for Convolutional Neural Networks,” Proc. 33rd Inter. Conf. Mach. Learn., vol. 48, 2016.

[22] H. Wang et. al., “Cosface: Large margin cosine loss for deep face recognition,” 2018 IEEE/CVF Conf. Comput. Vision and Pattern Recognit., pp. 5265-5274, 2018.

[23] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj and L. Song, “SphereFace: Deep Hypersphere Embedding for Face Recognition,” 2017 IEEE Conf. Comput. Vision and Pattern Recognit. (CVPR), pp. 6738-6746, 2017.

[24] V. Sati, S.M. Sánchez, N. Shoeibi, A. Arora, and J.M. Corchado, “Face Detection and Recognition, Face Emotion Recognition Through NVIDIA Jetson Nano,” 11th Int. Symp. Ambient Intell., pp. 177-185, 2020.

[25] S. Qureshi et. al. “Face Recognition (Image Processing) based Door Lock using OpenCV, Python and Arduino,” Int. J. Res. Appl. Sci. & Eng. Technol., vol. 8, no. 6, pp. 1208-1214, 2020

[26] N. Saxena, and D. Varshney, “Smart Home Security Solutions using Facial Authentication and Speaker Recognition through Artificial Neural Networks,” Int. J. Cognitive Comput. Eng., no. 2, pp. 154-164, 2021.

[27] M. H. Khairuddin, S. Shahbudin, and M. Kasssim, “A smart building security system with intelligent face detection and recognition,” IOP Conf. Series: Mater. Sci. Eng., no. 1176, 2021.

[28] M. R. Dhobale, R. Y. Biradar, R. R. Pawar, and S. A. Awatade, “Smart Home Security System using Iot, Face Recognition and Raspberry Pi,” Int. J. Comput. Appl., vol. 176, no. 13, pp. 45-47, 2020.

[29] T. Bagchi et. al., “Intelligent security system based on face recognition and IoT,” Materials Today: Proc., vol.62, pp. 2133-2137, 2022.

[30] P. Kaur, H. Kumar, and S. Kaushal “Effective state and learning environment based analysis of students’ performance in online assessment,” Int. J. Cognitive Comput. Eng., vol. 2, pp. 12-20, 2021.

[31] S. Pawar, V. Kithani, S. Ahuja, and S. Sahu, “Smart Home Security Using IoT and Face Recognition,” 2018 Fourth Int. Conf. Comput. Commun. Control Automat. (ICCUBEA), 2018.

[32] N. A. Othman, and I. Aydin, “A face recognition method in the Internet of Things for security applications in smart homes and cities,” 2018 6th Int. Istanbul Smart Grids and Cities Congr. Fair (ICSG), 2018.

[33] A. Kumar, P. S. Kumar, and R. Agarwal "A Face Recognition Method in the IoT for Security Appliances in Smart Homes, offices and Cities," 3rd Int. Conf. Comput. Methodologies and Commun. (ICCMC), 2019.

[34] T. Kamyab, A. Delrish, H. Daealhaq, A. M Ghahfarokhi, F. Beheshtinejad, "Comparison and Review of Face Recognition Methods Based on Gabor and Boosting Algorithms," Int. J. Robot. Control Syst., vol. 2, no. 4, pp 610-617, 2021.

[35] M. H. Widianto, A. Sinaga, and M. A. Ginting, "A Systematic Review of LPWAN and Short-Range Network using AI to Enhance Internet of Things," J. Robot. Control (JRC), vol. 3, no. 2, pp. 505-518, 2022.

[36] A. A. Sahrab and H. M. Marhoon, "Design and Fabrication of a Low-Cost System for Smart Home Applications," J. Robot. Control (JRC), vol. 3, no. 4, pp 409-414, 2022.

[37] G. P. N. Hakim, D. Septiyana, and Iswanto Suwarno, "Survey Paper Artificial and Computational Intelligence in the Internet of Things and Wireless Sensor Network," J. Robot. Control (JRC), vol. 3, no. 4, pp. 439-454, 2022.

[38] M. H. Widianto, A. Sinaga, and M. A. Ginting, "A Systematic Review of LPWAN and Short-Range Network using AI to Enhance Internet of Things," J. Robot. Control, vol. 3, no. 4, pp. 505-518, 2022.

[39] N. S. Irjanto and N. Surantha, "Home Security System with Face Recognition based on Convolutional Neural Network," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 11, pp. 408-412, 2020.

[40] M. Sandler, A. Howard, M. Zhu, Andrey Zhmoginov, and L. C. Chen. "MobileNetV2: Inverted Residuals and Linear Bottlenecks," The IEEE Conf. Comput. Vision Pattern Recognit. (CVPR), pp. 4510-4520, 2018.

[41] V. Patel and D. Patel, "Face Mask Recognition Using MobileNetV2," Int. J. Sci. Res. Comp. Sci. Eng. Inf. Technol, vol. 7, no. 5, pp. 35-42, 2021.

[42] C. L. Amar et al., "Enhanced Real-time Detection of Face Mask with Alarm System using MobileNetV2," Inter. J. Sci. Eng., vol.7, no. 7, pp. 1-12, 2022.

[43] G. Zhili, "Embedded Computer System in Hospital and Atomized Intelligent Monitoring of Children's Respiratory Diseases," Microprocessors and Microsystems, vol. 82, p. 103930, 2021.

[44] J. Ji and F. Lieng, "Influence of Embedded Microprocessor Wireless Communication and Computer Vision in Wushu Competition Referees' Decision Support," Wireless Communications and Mobile Computing, vol. 1, pp. 1-13, 2022.

[45] R. Chang, B. Li, and B, Zheng, "Research on visualization of container layout based on embedded computer and microservice environment," Journal of Physics Conference Series, vol. 1744, no. 3, p. 032210, 2021.

[46] W. Han, X. Bai, and J. Xie, "Assessment Model of the Architecture of the Aerospace Embedded Computer," Procedia Eng., vol. 99, pp. 991-998, 2015.

[47] A. Subbotin, N. Zhukova, and M. Tianxing, "Architecture of the intelligent video surveillance systems for fog environments based on embedded computers," 2021 10th Mediterranean Conf. Embedded Comput. (MECO), 2021.

[48] K. Belattar, A. Mehadjbia, A. Bala, and A. Kechida "An embedded system-based hand-gesture recognition for human-drone interaction," Int. J. Embedded Syst., vol. 15, no. 4, pp. 333-343, 2022.

[49] R. Li, "Computer embedded automatic test system based on VxWorks," Int. J. Embedded Syst., vol. 15, no.3, pp.183-192, 2022.

[50] O. G. Bedoya and J, V. Ferreira, "An embedded software architecture for the development of a cooperative autonomous vehicle," Int. J. Embedded Syst., vol.15, no. 2, pp. 83-92, 2022.

[51] H. Gao et. al., "Tool Wear Monitoring Algorithm Based on SWT-DCNN and SST-DCNN," Sci. Program., vol. 1, pp.1-11, 2022.

[52] A. Tariq, T. Zia, and M. Ghafoor, "Towards counterfactual and contrastive explainability and transparency of DCNN image classifiers," Knowl. Syst., vol. 257, no.10, 109901, 2022.

[53] A. B. W. Putra, A. F. O. Gaffar, M. T. Sumadi, and L. Setiawati, "Intra-frame Based Video Compression Using Deep Convolutional Neural Network (DCNN)," Int. J. Inform. Vis., vol. 6, no. 3, pp. 650-659, 2022.

[54] I. Kalita, G. P. Singh, and M. Roy, "Crop classification using aerial images by analyzing an ensemble of DCNNs under multi-filter & multi-scale framework," Multimedia Tools Appl., 2022.

[55] A. Saffari, M. Khishe, M. Mohammadi, A. Hussein Mohammed, and S. Rashidi, "DCNN-FuzzyWOA: Artificial Intelligence Solution for Automatic Detection of COVID-19 Using X-Ray Images," Comput. Intell. Neuroscience, vol. 11, pp. 1-11, 2022.

[56] S. Subbiah and R. Dheeraj, "Twitter Sentimentality Examination Using Convolutional Neural Setups and Compare with DCNN Based on Accuracy," ECS Trans., vol. 107, no. 1, pp. 14037-14050, 2022.

[57] H. Liu, J. Du, Y. Zhang, and H Zhang, "Performance analysis of different DCNN models in remote sensing image object detection," EURASIP J. Imag. and Video Process., vol. 1, pp. 1-18, 2022.

[58] R. Wang, T. Liu, J. Lu, and Y. Zhou, "Interpretable Optimization Training Strategy-Based DCNN and Its Application on CT Image Recognition," Math. Problems in Eng., vol. 3, pp. 1-13, 2022.

[59] Y. Le, Y. A. Nanehkaran, D. S. Mwakapesa, R. Zhang, J. Yi, and Y. Mao, "FP-DCNN: a parallel optimization algorithm for deep convolutional neural network," J. Supercomput., vol. 78, no. 3, 2022

[60] D. Hussain, M. Ismail, I. Hussain, R. Alroobaea, S. Hussain, and S. S. Ullah, "Face Mask Detection Using Deep Convolutional Neural Network (DCNN) and MobileNetV2 Based Transfer Learning," Wireless Commun. Mobile Comput., vol. 1, 2022.

[61] T. H. Nguyen and M. C. Nguyen, "Oscillation Measurement of the Magnetic Compass Needle Employing Deep Learning Technique," The AUN/SEED-Net Joint Regional Conf. Trans. Energy Mech. Manuf. Eng., pp. 220-229, June 2022.

[62] T. H. Nguyen, N. P. Doan, and T. T. Nguyen, "Battery Sorting Algorithm Employing a Deep Learning Technique for Recycling," International Conference on Advanced Mechanical Engineering, Automation and Sustainable Development, pp. 846-853, 2022.

[63] H. H. Hoang, T. H. Nguyen, V. H. Tran, T. H. Bui, and D. M. Nguyen: "Machine Vision System for Automated Nuts Classification and Inspection," International Conference on Material, Machines and Methods for Sustainable Development, pp. 755-759, 2020.

[64] T. V. Dang, "Design the encoder inspection system based on the voltage 7 wave form using finite impulse response (FIR) filter," Int. J. Modern Phys. B, pp. 1-6, 2020.