IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received 2 April 2023, accepted 30 April 2023, date of publication 8 May 2023, date of current version 12 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3273713

RESEARCH ARTICLE

A Lightweight Network Deployed on ARM Devices for Hand Gesture Recognition

MINGYUE ZHANG[1,2], ZHIHENG ZHOU[1,2], (Member, IEEE), TIANLEI WANG[3], AND WENLVE ZHOU[1,2]

[1] School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China
[2] Key Laboratory of Big Data and Intelligent Robot, South China University of Technology, Guangzhou 510641, China
[3] Department of Intelligent Manufacturing, Wuyi University, Jiangmen 529020, China

Corresponding author: Zhiheng Zhou (zhouzh@scut.edu.cn)

This work was supported in part by the Open Fund of the National Key Research and Development Program of China under Grant 2022YFF0607000, in part by the National Natural Science Foundation of China under Grant 61871188, and in part by the Guangdong Provincial Key Laboratory of Human Digital Twin under Grant 2022B1212010004.

ABSTRACT Hand gestures are a well-known and straightforward method of human-computer interaction. The majority of the study focused on hand gesture recognition. However, little work has been done to develop a complete set of gesture recognition applications. With the improvement of model feature extraction ability and the increase in the number of model parameters, it is becoming more challenging to achieve a small memory footprint on mobile devices based on an ARM architecture or CPU devices based on x86 architecture. However, these existing methods are heavy, requiring more memory and inference time. The execution of memory-efficient CNNs without compromising accuracy has been a challenge, especially when the inference has to be performed on an edge computing device in real time. Therefore, we propose a lightweight network for hand gesture recognition (LHGR-Net) and deploy it on a Raspberry Pi. LHGR-Net consists of three main parts: the base network structure, the multiscale structure (MSS), and the lightweight attention structure (LAS). We present pre-trained weights that are learned from other data to initialize the network structure. In addition, the LHGR-Net model was made to be deployed on a Raspberry Pi, and a deployed model can be used to control home appliances. Extensive experiments show that our method achieves almost as good as state-of-the-art performance in hand gesture recognition and running time.

INDEX TERMS Lightweight, deployment, multiscale structure, lightweight attention structure, pre-trained weights, Raspberry Pi.

I. INTRODUCTION

In computer science and language technology, hand gesture recognition aims to interpret human gestures through mathematical algorithms. In social interactions, gestures play an important role in conveying information and expressing our thoughts and feelings more effectively. It is possible to interact with machines by using hand gestures. Hand gesture recognition (HGR) is an active area of research in visual pattern analysis in applications such as human-computer interaction [1], sign language communication [2], virtual reality [3], and smart homes [4]. Currently, the most common hand gesture recognition application is sign language recognition [5], intelligent robotic [6], and intelligence controlling of household-appliances [7]. While a large number of studies have been devoted to such a field, existing approaches generally employ complex models to extract gesture features, which can lead to problems of high computational consumption and also high model parameters. In addition, existing hand gesture recognition does not fully consider the whole process, from model design to deployment and application.

Due to the recent advance in hand-crafted features and deep-learned features, a large number of attempts have been made to the hand gesture recognition area. In particular, these studies [5], [8], [9], [10], [11] can be categorized into two categories: traditional method-based approaches [8], [9],

The associate editor coordinating the review of this manuscript and approving it for publication was Utku Kose.

and deep learning-based approaches [5], [10], [11]. The systems proposed by [8] and [9] are typical examples of traditional method-based approaches, which used hand-crafted features extractor and SVM classifier for the gesture recognition task. Although traditional method-based approaches are fast, they not only have complex operations but also low efficiency and poor generalization ability. Deep neural networks have achieved significant success in the field of image classification in recent years. Thus, deep learning-based approaches have been widely investigated. For example, to address the problem of overfitting, Fang et al. [10] proposed a deep convolutional generative adversarial network (DCGAN). By using fewer samples for training, the authors also achieved better performance. Sharma and Singh [5] proposed a G-CNN model packed with compact representation, the remarkable recognition results are obtained. Cheng et al. [11] proposed a joint network of CNN and RBM for gesture recognition. It has been demonstrated that the jointly proposed network can identify simple background gesture samples more accurately; however, the ability to recognize gesture samples against complex backgrounds still needs to be improved. Moreover, the model is complex and contains a large number of parameters.

While the aforementioned approaches already achieved good recognition performance on the gesture dataset, they are still not able to provide reliable predictions on the different scales of gestures and a very small percentage of gestures in images. In addition, with the improvement of model feature extraction ability and the increase in the number of model parameters, it is becoming more challenging to achieve a small memory footprint on mobile devices based on ARM architecture or CPU devices based on x86 architecture. Generally, classical network structures(ResNet [12], DenseNet [13], HRNet [14]) are used for hand gesture recognition tasks; however, these structures are heavy, requiring more memory and inference time. These models are difficult to deploy on edge computing devices. The execution of memory-efficient CNNs without compromising accuracy has been a challenge, especially when the inference has to be performed on an edge computing device in real time. Meanwhile, gesture recognition technology is used mostly to perform gesture prediction tasks on computer and server platforms, making it difficult to develop a rapid product. The reason is that gesture recognition models are large and difficult to deploy. The Raspberry Pi is a popular device for edge computing; it is very small, lightweight, and has a very low power supply. The current gesture recognition models have difficulty deploying gesture recognition models on Raspberry Pi and achieving effective recognition. In addition, some other reasons are the lack of suitable hardware platforms and the fact that researchers in this field pay less attention to the model deployment landing and focus more on model optimization.

As a consequence, this paper proposes a lightweight network for hand gesture recognition (LHGR-Net) and deploys it on a Raspberry Pi. LHGR-Net consists of three main parts: the base network structure, the multiscale structure (MSS), and the lightweight attention structure (LAS). The motivation behind our LHGR-Net method is based on considering that both the MSS and the LAS can strengthen the representation ability of a neural network. The multiscale architecture can not only capture the global structure of input images but also retain their local details [15], while attention modules can well handle long-range dependencies, which enables the neural network to focus more on useful information within a context [16], [17]. Regarding the image hand gesture recognition task, it is better to combine them rather than separately employing them so that their merits are taken and their demerits are overcome. To the best of our knowledge, this is the first attempt to implement the whole process, design algorithms, model deployment, and applications for hand gesture recognition. To evaluate the performance of the proposed LHGR-Net approach on the HGR1 and OUHANDS datasets. This proposed model is found to be advantageous over the present state-of-art approaches since it has high accuracy and inference speed.

The main contributions of this article are as follows:

- We propose a lightweight CNN model called LHGR-Net that can classify the type of multi-size gestures with extremely low computational complexity while maintaining high accuracy. Additionally, LHGR-Net provides a better balance between inference speed and accuracy.
- We implement the whole process, design algorithms, model deployment, and applications. Most current algorithms are not considered as models are applied. The LHGR-Net model is deployed to a Raspberry Pi, and a deployed model is used to control home appliances.
- LHGR-Net is state-of-the-art in terms of inference speed and accuracy trade-off on the HGR1 and OUHANDS datasets, and the experimental results show that LHGR-Net performs better than other networks in other evaluation metrics.

II. RELATED WORKS

To overcome the challenges of real-time deployment and memory efficiency of the deep learning models. In the recent literature [18], [19], [20], [21], [22], there has been rising interest in building small and efficient neural networks.

MobileNetV2 [18] adopts a resource-efficient block with inverted residuals and linear bottlenecks. ShuffleNet [19] employs two operations, group convolution and channel shuffle, to design a convolutional neural network model to reduce the number of parameters. The author of GhostNet [20] proposed a novel ghost module for constructing neural network structures. With the ghost module, the original convolutional layer is divided into two parts: first, by creating the original feature maps with fewer convolutional kernels and then, by producing more ghost feature maps via cheap transformation operations. In these studies, some lightweight modules were designed. LightAMC [21] introduces a scaling factor for each neuron in a convolutional neural network and enforces scaling factor sparsity via compressive sensing. It can assist in screening out redundant neurons, and these neurons are pruned. PCBNet [22] compresses and expands the dimensions of input images via convolution layers and then extracts feature maps. After that, the feature maps are input into the tail of the backbone, which consists of global average pooling (GAP), a 1 × 1 convolution, and a flattened layer. The literature [18], [19], [20], [21], [22] provided a lightweight network structure using the efficient structure of network design ideas. Reference [18] retained the depth-separable convolution, and a linear bottleneck and inverted residual were added. Literature [19] proposed a method primarily composed of two operations: pointwise group convolution and channel shuffle. Literature [20] used a series of linear variations to find the desired information from the original features. Literature [21] first introduced a scaling factor for each neuron in a convolutional neural network and enforced scaling factor sparsity via compressive sensing, which can assist in screening out redundant neurons, and then these neurons are pruned. Reference [22] provided two versions of the modified FusedMBConv block: standard and downsampling, which contain a 3 × 3 convolution block and a 1 × 1 convolution block. References [18] and [22] employed depth-separable convolution, which compresses the size and expands the dimension of input images through convolution layers and then outputs extracted feature maps. The literature [19], [20], [21] focused on the model, feature maps, redundancy of neurons, and solution methods.

Breland et al. [23] developed a model based on the bottleneck module, which was inspired by deep residual networks and MobilenetV2. They utilized the 'Sign Language Digits Dataset' to train the proposed model, and the model was deployed on a Raspberry Pi 4 Model B edge computing system to serve as an edge device for user verification. Dayal et al. [24] presented a model design consisting of several bottleneck layers, which were inspired by deep residual networks, and the model was deployed on a Raspberry Pi 4 Model B edge computing system to classify hand gestures captured from thermal images. Breland et al. [25] proposed a robust hand gesture recognition system based on high-resolution thermal imaging that is light-independent. The proposed models based on the dilated convolution layer were also tested on Raspberry Pi 4 edge computing devices. Sikkandar [26] proposed a memory-efficient deep learning convolutional neural network model to identify and classify the hand movements of sign language digits and extract the function by combining the two BEMD and SIFT algorithm techniques. The model was deployed in the Raspberry Pi 4 Model B edge computing system to act as an edge device for user verification. The above literature mostly focused on analysing sign language digits, whereas the literature [24], [25] was devoted to analysing sign language digits from thermal images. The network structures used were based on residual networks and basic operations such as the dilated convolution layer.

The optimization of these algorithms is somewhat detached from the industry's most commonly used ARM architecture CPU device environment, and the acceleration capabilities often fall short of expectations. In addition, lightweight models are rarely applied for industrial deployment on gesture recognition tasks. In this paper, the LHGR-Net model is designed for the gesture recognition deployment application task, using DepthSepConv as the base block. To avoid reducing inference speed, we discard the commonly used Shortcut Connections and use the H-Swish activation function with the exponential operation. In order to cope with gesture size diversity in images, the MSS structure is adopted, and an improved ASPP module is introduced in the MSS structure, inspired by the target detection task. To obtain more meaningful feature information from gesture images, the LAS structure is used. The overall structure of the model is composed of these structures and strategies. The model is also deployed and applied to achieve the full process.

III. PROPOSED METHOD

We propose a lightweight gesture recognition network (LHGR-Net) that uses pre-trained weights from other data training to build initial weights of the network structure, which is retrained using a new dataset. It is possible to deploy the lightweight model on a Raspberry Pi to achieve full process success.

A. LIGHTWEIGHT NETWORK ARCHITECTURE FOR HAND GESTURE RECOGNITION

Due to the deeply optimized depthwise separable convolution (DepthSepConv) block by Intel's CPU acceleration library, the inference speed can exceed other lightweight blocks, such as inverted blocks and ShuffleNet blocks. Therefore, we adopt DepthSepConv of MobileNetV2 [18]. We also use the hard-swish activation function of EfficientNet [27] since the original activation function was improved to avoid a large number of exponentiation operations. In addition, the adaptive pooling operation is also used to reduce network parameters [28]. Every basic block of the network structure consists of both DepthSepConv and the hard-swish activation function to combine to form F1, F2, the multiscale structure (MSS), and the lightweight attention structure (LAS). A set of F1 and F2 is composed of two convolutional blocks of 3 × 3 and four convolutional series, respectively, where each convolutional series is made up of several basic blocks. F3 is composed of 1 × 1 convolutional blocks followed by fully connected layers. The feature extraction process is performed by F1 and F2, while gesture classification is performed by the softmax classifier. Additionally, the MSS and the LAS are incorporated. As shown in Figure 1, the above series of operations define LHGR-Net.

1) MULTISCALE ENSEMBLE STRUCTURE FOR LHGR-NET

The different scales of gestures in an image make it difficult to obtain gesture feature maps at different scales simply through convolution, and it is difficult to report satisfactory results for tasks related to gestures. To address this challenge, this paper exploits the atrous spatial pyramid pooling (ASPP) module in DeepLabv3 [29] to extract multiscale features.

With the help of the ASPP module of DeepLabV3, multiple scales of contextual image information can be effectively extracted with different sampling rates of atrous convolution. It also achieves good results for semantic segmentation tasks. Based on this, we propose a multiscale structure (MSS) for gesture recognition.

The multiscale structure is shown in Figure 1. It consists of a modified ASPP module and 1 × 1 convolution blocks. The modified ASPP module uses a five-branch atrous convolution, which uses expansion coefficients of 1, 3, 6, 12, and 18, corresponding to the number of output channels of 256, 128, 64, 32, and 16. By using 1 × 1 convolution, batch normalization, and hard-swish activation function operations, the 1 × 1 convolution block can fuse different scales of feature maps obtained from each branch of a modified ASPP module while reducing the number of channels. The MSS can be used to increase the receptive field without increasing the parameters, which can effectively extract feature information at different gesture scales. The MSS can be formulated as:

$$\begin{aligned} Y = {} & \delta_{256}\left(BN\left(f_{1,1}(F)\right)\right) + \delta_{128}\left(BN\left(f_{3,3}(F)\right)\right) \\ & + \delta_{64}\left(BN\left(f_{3,6}(F)\right)\right) + \delta_{32}\left(BN\left(f_{3,12}(F)\right)\right) \\ & + \delta_{16}\left(BN\left(f_{3,18}(F)\right)\right), \qquad (1) \end{aligned}$$

$$\hat{Y} = \delta_{16}\left(BN\left(f_{1}(Y)\right)\right), \qquad (2)$$

where $f_{n,m}(.)$ denotes a mapping function learned by the n × n convolutional layer, m denotes the dilation rate, F denotes the input feature map. BN(.) denotes batch normalization to alleviate internal covariate shift, $\delta_c(.)$ is a hard-swish activation function of EfficientNet [27], c denotes the number of channels. Y denote the intermediate features resulted from the MSS.

2) LIGHTWEIGHT ATTENTION ENSEMBLE STRUCTURE FOR LHGR-NET

It is difficult to optimize the model performance using conventional operations since gestures are a very small percentage of a large number of images. The above deficiencies can be effectively addressed by convolutional block attention module (CBAM)-based modules. CBAM is a lightweight and general module that can be seamlessly integrated into any CNN architecture with negligible overhead [30]. Since the CBAM module was proposed, it has been used by a large number of networks. It does a good job of weighting the network channels and space for better features. Based on this, we propose a lightweight attention structure (LAS) for gesture recognition based on this.

The lightweight attention structure is shown in Figure 1. It consists of DepthSepConv groups and CBAM blocks. According to PP-LCNet, the attention mechanism is located at the end of the network and can play a better role [31]. The PP-LCNet ensures that a large convolution kernel is used in the case of low latency and high accuracy [31]. The lightweight attention structure is derived from the PP-LCNet method in addition to the CBAM module at the tail of the network structure. To obtain more feature information, the convolutional kernel size is increased and integrated into the CBAM module, and feature fusion is carried out using 1 × 1 convolutional blocks. The result is a reduction in parameters and a decrease in computational resources. To complete the full operation of the lightweight attention structure, the above procedure is repeated once. The LAS is computed as:

$$F' = \delta_{256}\left(BN\left(f_5(F)\right)\right), \qquad (3)$$

$$\begin{aligned} F'' &= M_c(F') \otimes F', \\ F''' &= M_s\left(F''\right) \otimes F'', \qquad (4) \end{aligned}$$

$$\tilde{F} = \delta_{512}\left(BN\left(f_1\left(F'''\right)\right)\right), \qquad (5)$$

where $f_n(.)$ denotes a mapping function learned by the n × n convolutional layer and F and F' denotes an intermediate feature map. $BN(.)$ denote batch normalization to alleviate the internal covariate shift, $\delta_c(.)$ is a hard-swish activation function of EfficientNet [27], and c represents the number of channels. ⊗ represents element-wise multiplication, CBAM sequentially infers a 1D channel attention map M_c and a 2D spatial attention map M_s. F'' and F''' represent the results of channel attention and spatial attention, respectively.

B. DEPLOYMENT BASED ON THE LHGR-NET MODEL AND APPLICATION

In Paddle Lite, a variety of strategies are provided to optimize the original training model, including quantization, subgraph fusion, hybrid scheduling, and kernel preference. With the opt tool, we automate the optimization steps and produce a lightweight, optimized executable model. There is also a lightweight serialization/deserialization implementation among the output model types.

The figure shows a block diagram of the LHGR-Net model deployment structure and application. The LHGR-Net model is deployed to the Raspberry Pi controller based on Paddle Lite, and the Raspberry Pi is capable of gesture recognition. Raspberry Pi controllers receive gesture information collected by the CSI camera, and a deployed model generates the corresponding control commands that can be used to control home appliances. On the right, the circuit diagram for home appliance control is shown. First, the general-purpose input/output (GPIO) level is adjusted by deploying the prediction result, and the amplifier circuit is used to amplify the current. Then, the relay will be opened and closed, the infrared emitters transmit control commands, and finally, the home appliance is controlled.

IV. EXPERIMENTS

We evaluated our method on two benchmark datasets:

HGR1 [32] - This dataset contains 899 RGB images, skin masks and feature points of 12 individuals performing 25 different hand gestures. This dataset is split into training, validation and testing sets with 631, 179 and 89 images, respectively. The HGR1 collected the gestures from Polish Sign Language and American Sign Language under uncontrolled backgrounds without any occlusion. We only use RGB images for evaluating hand gesture recognition accuracy.

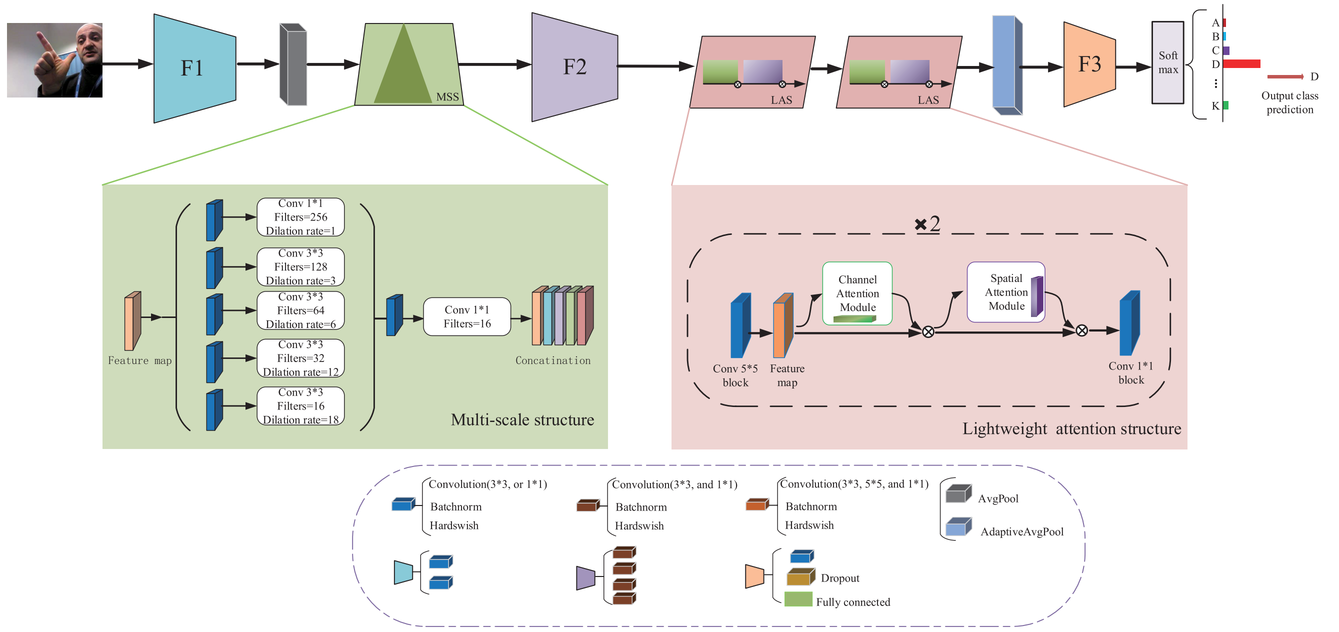


FIGURE 1. Illustration of the architecture of the proposed LHGR-Net. It is composed of three main parts: the BaseNet structure, the multiscale structure, and the lightweight attention structure.

OUHANDS [33] - This dataset contains 3,000 RGB images, bounding boxes, depths and segmentations of 23 individuals performing 10 different hand gestures. This dataset is split into training, validation and testing sets with 2,100, 600 and 300 images, respectively. There are images in each set that are highly challenging, such as varying lighting, complex backgrounds, and face-hand occlusions with a range of hand shapes and sizes. We use this dataset to evaluate hand gesture recognition accuracy.

A. EXPERIMENTAL SETUP

1) IMPLEMENTATION DETAILS

To implement the proposed method, we use a deep learning library with PaddlePaddle. Using the cross-entropy loss as the cost function, we used the momentum optimization algorithm to minimize the loss. We set the learning rate to 0.025 and the mini-batch size to 32 for training the hand gesture recognition network. We also employ pre-trained strategies for the gesture recognition task.

When the training data for hand gesture recognition in this study are insufficient, there is a lack of useful information to be learned, which results in poor recognition results. Using pre-trained weights learned from other data, the network structure is initialized, and the training data are fed into the LHGR-Net for retraining and outputting prediction results. Our proposed LHGR-Net is trained on the hand-pose_gesture_v1 dataset [34], and the weight parameters are used as initial weights to learn on the HGR1 [32] and OUHANDS datasets [33].

We train the proposed method by using an Nvidia RTX 2080Ti GPU. The maximum number of epochs for training is set at 200, and after evaluating the samples for each category, the mean accuracy is calculated.

2) EVALUATION METRIC

Statistical measures are used to evaluate hand gesture recognition performance. In addition to the mean accuracy (mAcc), Recall, and F1-score, which are calculated to evaluate the system's efficiency, inference time and model parameters are two more essential assessment metrics that must be considered for lightweight models.

Accuracy is one of the aspects that must be considered when constructing classification models. Informally, accuracy refers to the proportion of correct predictions made by a model. The mAcc is obtained by calculating the weighted average of the average accuracy (Acc) of all category detections. It is also a crucial parameter for evaluating hand gesture recognition.

$$\begin{cases} Acc_i = \dfrac{TP_i + TN_i}{TP_i + FP_i + FN_i + TN_i} \\ mAcc = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} Acc_i \end{cases} \tag{6}$$

In reality, when we attempt to enhance the precision of our model, the recall suffers, and vice versa. The F1-score captures the following tendencies in a single number:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

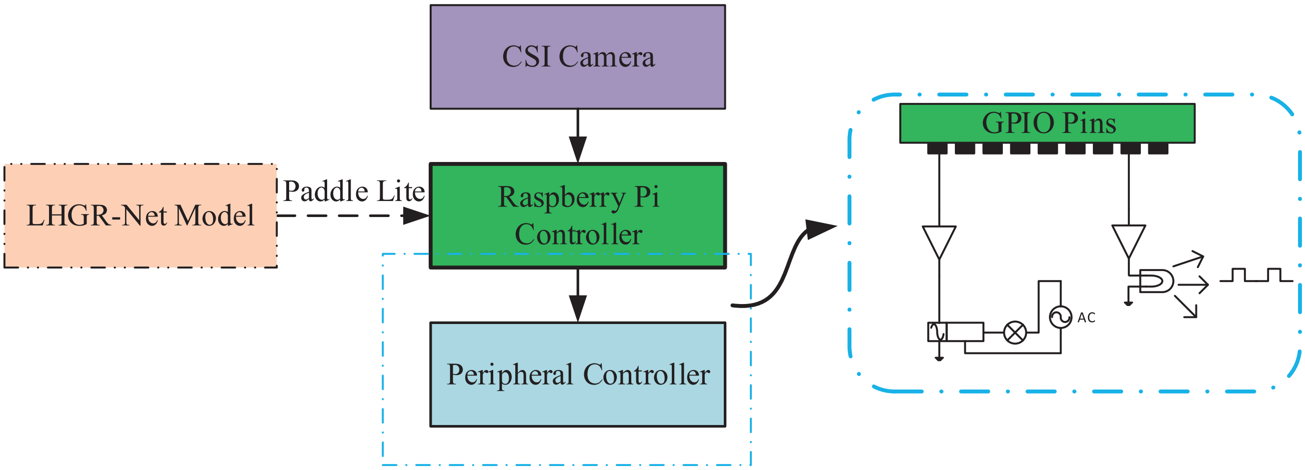


FIGURE 2. The figure shows a block diagram of the LHGR-Net model deployment structure and application.

$$F1-\text{score}=2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \tag{9}$$

where TP is true positive, TN denotes true negative, FP denotes false positive and FN denotes false negative.

B. DATA AUGMENTATION

To increase image diversity, data enhancement strategies are implemented. We use data enhancement strategies, including mixup, RandomCrop, RandomHorizontalFlip, and normalize. By mixing images from different classes, a mixup algorithm can expand the training dataset in computer vision. In image classification, mixup is an important image enhancement technique [35]. In this study, the input network images are cropped due to the inconsistent image sizes in the dataset. Using RandomCrop, we adjust the image sizes to 224. Because of the angle deviation in the same gesture, the diversity of the image is increased by horizontal reversal of the angle so that it can adapt to gestures at various angles. Then, the normalization enhancement strategy is carried out. Normalization is performed for the convenience of data processing and to accelerate the convergence speed.

C. EXPERIMENTS ON HAND GESTURE RECOGNITION

For the hand gesture recognition task, we evaluated LHGR-Net. Tables 1 and 2 show the classification scores on the HGR1 and OUHANDS test sets, respectively. Tables 1 and 2 compare the performances of seven popular network architectures (ResNet-50 [12], MobileNet [36], EfficientNet [27], DeepLabV3 [29], PP-LCNet [31], HRNet [14]) and two new gesture recognition network structures (Pinto et al. [37], ExtriDeNet [38], EDenseNet [39]) on the HGR1 and OUHANDS datasets, respectively. In Rows 1-6 of Tables 1 and 2, we present the scores of six popular deep networks. We only replaced the softmax layer, which was originally trained to recognize 1,000 classes, with a softmax layer that recognizes 25 and 10 classes for training these networks on the HGR1 and OUHANDS datasets, respectively. In Rows 7-9 of Tables 1 and 2, we present the scores of three new gesture recognition network structures. The most recent algorithms among them are ExtriDeNet and EDenseNet, with EDenseNet, in particular, having been published in journals with high impact factors. In addition, ExtriDeNet provides state-of-the-art hand gesture recognition approaches.

We report the results of our method and other state-of-the-art methods in Table 1. DeepLabV3 (Row 4) outperforms these models (Rows 1-3, 5, and 6), achieving 90.62% on mAcc, and there are many parameters in this model. MobileNet is often used in mobile or embedded devices as a lightweight network structure. EDenseNet (Row 9) outperforms the models (Row 7,8), achieving 92.02% on mAcc, which requires considerable inference time. Our LHGR-Net outperforms these models, obtaining a 3.74% increment over the next best on mAcc, and the parameters of the model are suitable for deployment at the edge. In addition, the inference time for our LHGR-Net is 27 ms, nearly two times as fast as the best reasoning algorithm. The performance of LHGR-Net (pre-trained) as a whole then jumps to an mAcc of 2.61% when we apply retraining the pretrained weights on the handpose_gesture_v1 dataset as the initial weights. This result emphasizes the effectiveness of our network architecture.

Table 2 reports the hand gesture recognition performances of our approach and the existing state-of-the-art approaches. As a result, PP-LCNet (Row 5) outperforms these models (Rows 1-4,6), which achieved 97.53%. In comparison to the models (Row 7), ExtriDeNet (Row 8), and EDenseNet (Row 9) has a better mAcc of 96.73% but requires a considerable amount of time to infer predictions. As a lightweight network structure, MobileNet is often used in mobile or embedded devices. Our LHGR-Net outperforms the next best with 1.04% improvements in mAcc, 15 ms reductions in inference time, and parameters suitable for edge deployment. As a result, our network architecture is proven to be effective. It has the same mAcc as LHGR-Net (pre-trained) when

TABLE 1. For a fair comparison, we used the same training strategy and hyperparameters for all models in the table to achieve the recognition mean accuracy(mAcc), Recall, F1-score, AUC, inference time, and parameters of the model with respect to the HGR1 test set.

Model	mAcc↑	Recall↑	F1-score↑	AUC↑	Time(ms))↓	Params(MB))↓
ResNet-50 [12]	0.8126	0.8203	0.8094	0.9805	50	90.11
MobileNet [36]	0.8426	0.8376	0.8403	0.9906	59	8.78
EfficientNet [27]	0.8437	0.8413	0.8492	0.9915	92	45.23
DeepLabV3 [29]	0.9062	0.9092	0.9107	0.9938	90	129.95
PP-LCNet [31]	0.8603	0.8697	0.8538	0.9925	130	7.37
HRNet [14]	0.8769	0.8874	0.8706	0.9936	328	74.01
Pinto et al. [37]	0.5701	0.5825	0.5726	-	1570	29.75
ExtriDeNet [38]	0.9183	0.9275	0.9207	-	1300	4.95
EDenseNet [39]	0.9202	0.9186	0.9224	0.9927	173	27.59
LHGR-Net	0.9315	0.9346	0.9286	0.9975	27	6.45
LHGR-Net(pre-trained)	0.9576	0.9524	0.9613	0.9986	27	6.45

TABLE 2. Comparison of recognition mean accuracy(mAcc), Recall, F1-score, AUC, inference time, and parameters of the model on the OUHANDS test set. Note that for fair comparison we use the proposed data augmentation strategy for training all models considered in the table.

Model	mAcc↑	Recall↑	F1-score↑	AUC↑	Time(ms))↓	Params(MB))↓
ResNet-50 [12]	0.9316	0.9296	0.9304	0.9916	48	89.98
MobileNet [36]	0.9492	0.9517	0.9503	0.9936	40	8.70
EfficientNet [27]	0.9534	0.9517	0.9571	0.9939	81	45.12
DeepLabV3 [29]	0.9632	0.9601	0.9686	0.9953	110	129.82
PP-LCNet [31]	0.9753	0.9757	0.9764	0.9986	120	7.29
HRNet [14]	0.9619	0.9583	0.9591	0.9952	234	73.87
Pinto et al. [37]	0.4036	0.4091	0.4106	-	1430	29.68
ExtriDeNet [38]	0.6426	0.6473	0.6412	-	1250	4.87
EDenseNet [39]	0.9673	0.9628	0.9693	0.9934	157	27.32
LHGR-Net	0.9857	0.9873	0.9882	0.9987	25	6.37
LHGR-Net(pre-trained)	0.9857	0.9873	0.9882	0.9987	25	6.37

we apply to retrain the pre-trained weights on the hand-pose_gesture_v1 dataset as the initial weights. The mAcc is not improved when the pre-training weights are initialized because it reaches a certain value.

To assess the performance of the qualitative analysis model, this study employed two testing sets - the HGR1 testing set and the OUHANDS testing set, displaying the confusion matrix and prediction performance figures for gesture recognition, respectively. Figures 3 and 4 in this paper present the prediction results for the HGR1 testing set. Figure 3 presents the confusion matrix. It reveals that due to the limited number of images in the HGR1 testing data, the number of diagonal elements in the confusion matrix is also relatively low. By observing the confusion matrix, it can be found that the letters "S" and "T" are the most easily mistaken in all prediction results, while other prediction results are relatively accurate. In order to further improve the model's performance, more data can be added for letters that are more likely to be mistaken. Additionally, Figure 4 shows prediction performance figures for four different categories of predictions. It can be seen that the model still has acceptable prediction performance under low light and shadow conditions, indicating that the model has significant adaptability to different lighting conditions. Overall, this model has certain advantages in gesture recognition, but there is still room for improvement.

In this article, Figure 5 shows the results of prediction based on the OUHANDS gesture test dataset. A confusion matrix is used to evaluate the performance of the gesture recognition model, which displays the cross-tabulation between actual gestures and predicted gestures. In this confusion matrix, each row represents the true gesture, and each column represents the predicted gesture. For example, the sixth row represents the actual gesture as gesture F, where 62 gestures F were correctly predicted, and 2 gestures F were incorrectly predicted as gesture B. This is also one of the letters that is easiest to misclassify in the OUHANDS gesture dataset: "F" and "B". It should be noted that the numbers on the diagonal of this confusion matrix indicate the number of correct predictions, not the number of incorrect predictions on the diagonal. In addition, Figure 6 shows the recognition results of OUHANDS gesture test data. The model performs well in recognizing gestures in different complex backgrounds, indicating good adaptability of the model in different backgrounds.

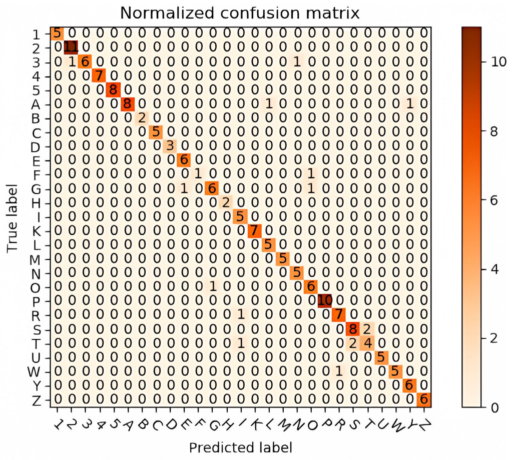


FIGURE 3. Confusion matrix of the predictions made by the model trained with the HGR1 gesture dataset.

D. ABLATION STUDY

To demonstrate the importance of each block within the LHGR-Net, we conducted an ablation experiment. In the experiment, we gradually remove components of the proposed framework and re-train and test the models using the HGR1 dataset. In order to verify that the LAS improves the model accuracy, we add the LAS into the backbone of the structure. We incorporate the MSS into the structure's backbone in order to demonstrate that it can increase the model's feature extraction capability, which will increase the recognition accuracy. In Table 3, we report the evaluation results for four ablation models with the mean accuracy(mAcc) and the number of model parameters(Params) metric.

The analysis of the ablation experiment results shows that the recognition accuracy of the model is increased by 0.88% after the addition of LAS which does a good job of weighting the network channels and space for better features in a very small percentage gesture of image. The MSS is introduced to enhance the extraction of multiscale features, and the recognition accuracy of the model is increased by 1.46%. We observe each additional block improves the precision of model identification. Additionally, the MSS has a higher impact than LAS on the mAcc metric because the data contains a large number of gestures of various sizes. We can achieve an improvement

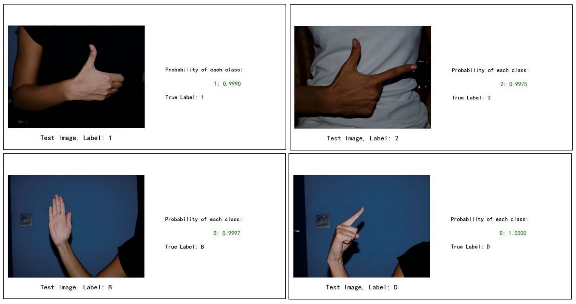
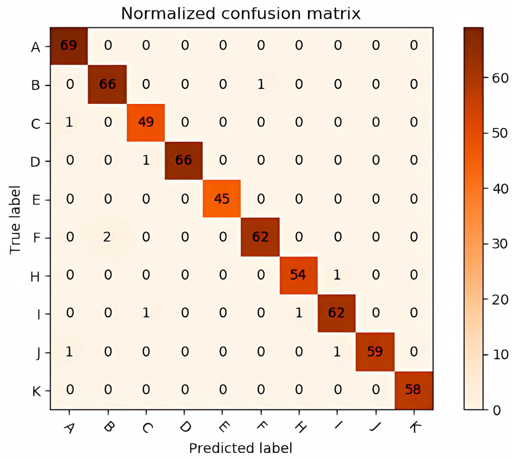


FIGURE 4. Prediction results based on the HGR1 gesture test dataset.



	A	B	C	D	E	F	H	I	J	K
A	69	0	0	0	0	0	0	0	0	0
B	0	66	0	0	0	1	0	0	0	0
C	1	0	49	0	0	0	0	0	0	0
D	0	0	1	66	0	0	0	0	0	0
E	0	0	0	0	45	0	0	0	0	0
F	0	2	0	0	0	62	0	0	0	0
H	0	0	0	0	0	0	54	1	0	0
I	0	0	1	0	0	0	1	62	0	0
J	1	0	0	0	0	0	0	1	59	0
K	0	0	0	0	0	0	0	0	0	58

FIGURE 5. Confusion matrix generated from the OUHANDS gesture test dataset for model performance evaluation.

of 2.70% by integrating the MSS and LAS blocks into our LHGR-Net. A significant improvement is brought about by the combination of the MSS and LAS blocks, according to the findings reported in LHGR-Net.

E. LHGR-NET DEPLOYED ON A RASPBERRY PI FOR HAND GESTURE RECOGNITION

To implement the deployment model, a Paddle Lite tool is used. We used a Raspberry Pi 4B as the hardware deployment platform. Gesture images were acquired with a Pi camera, which can capture still images up to 3280 × 2464 pixels and video at a resolution of up to 1080p at 30 frames per second. The final step in implementing a product is the deployment model. As a result, the Raspberry Pi controller gains a soul. Based on the Raspberry Pi deployment model, hand gesture recognition can be easily implemented as intelligent control.

In Figure 7, we can see the predicted results of gesture recognition on the Raspberry Pi. A prediction of Raspberry Pi video capture results is shown in the top row, and the results of Raspberry Pi image prediction using the OUHANDS dataset are shown in the middle row, while the bottom row corresponds to the prediction of the panorama of the Raspberry Pi. The bottom row of images was taken by a cell phone and appears blurry, the middle image is a crop of the predicted results of the OUHANDS test data, and the top row of images is the predicted video images captured by the Raspberry Pi in real time and cropped by the Raspberry Pi function. According to Figure 7, the average time is between 30 and 40 ms based on the Raspberry Pi CPU prediction, which satisfies the real-time requirements of the product landing. Furthermore, the prediction of accuracy using the Raspberry Pi CPU model achieves high performance. It enables intelligent control of home devices through Raspberry Pi deployment. Figure 8 demonstrates that the model was deployed on a Raspberry Pi and successfully applied to a smart home, enabling the control of the desk lamp to be executed successfully.

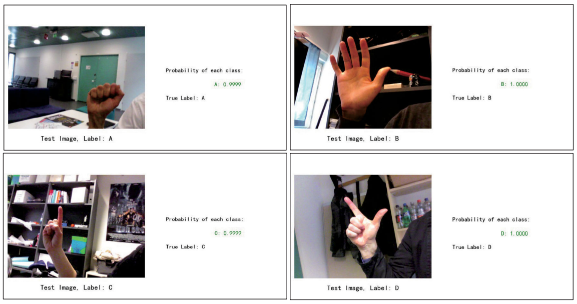


FIGURE 6. Visualization of prediction results from the OUHANDS gesture test dataset.

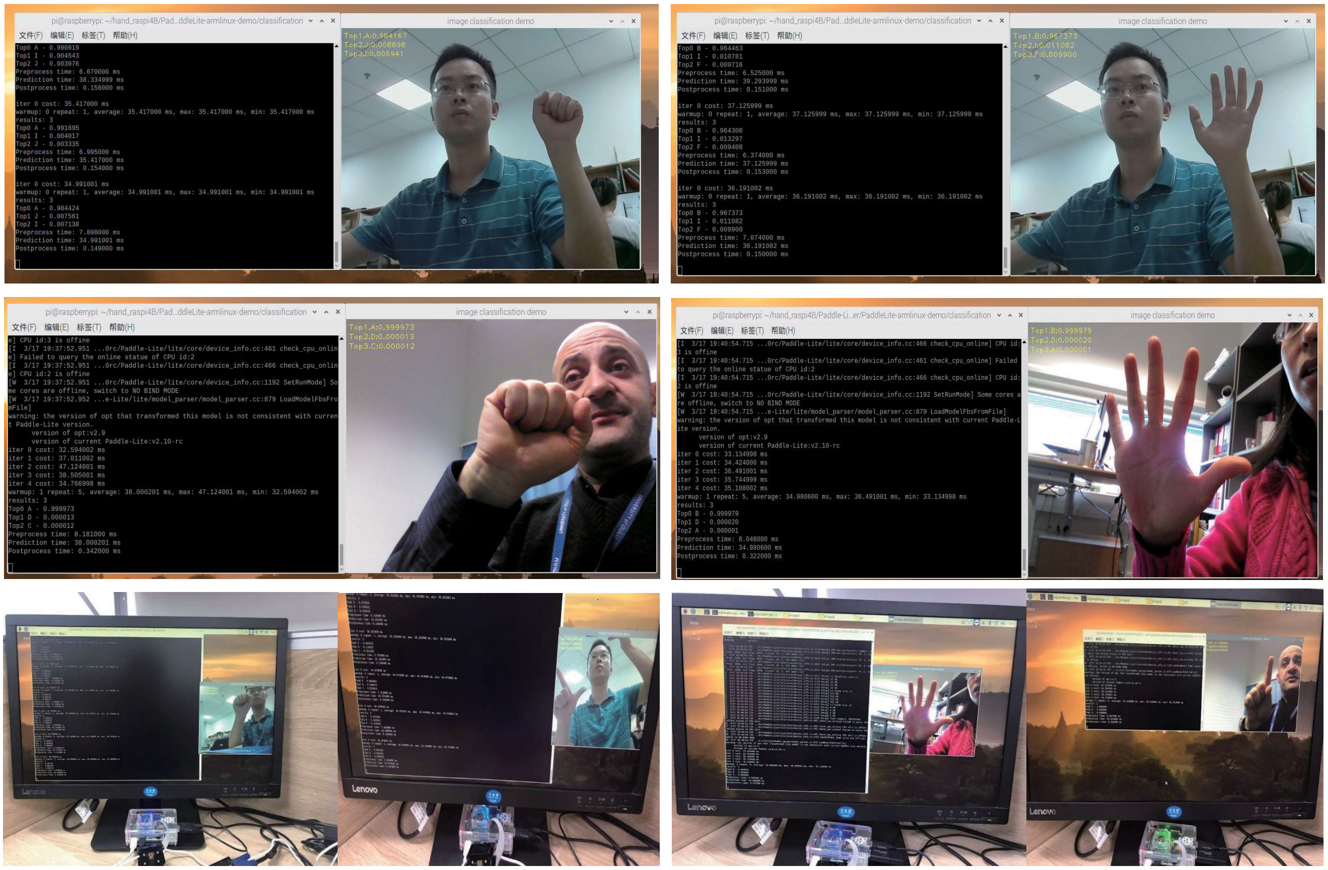
TABLE 3. Numerical results for the ablation study of hand gesture recognition evaluated the different types of ablation on the HGR1 testing data.

Method	LAS(include:CBAM)	MSS(include:ASPP)	mAcc↑	Params(MB))↓
BaseNet	-	-	0.9306	5.95
	√	-	0.9394	6.12
	-	√	0.9452	6.25
LHGR-Net	√	√	0.9576	6.45

F. DISCUSSION

In this paper, we show that this is the first attempt to implement the whole process, including design algorithms, model deployment, and applications for hand gesture recognition. Additionally, the algorithm provides a better balance between inference speed and accuracy in this paper.

Further comparisons with previous studies yield various interesting findings. First, the use of vision-based gesture recognition is commonly used for computer application control, and hardware-based hand gesture recognition is commonly used for embedded terminal applications with myoelectricity and PAJ7620U2 sensors. To the best of our knowledge, this is the first attempt to implement the whole process, design algorithms, model deployment, and applications for hand gesture recognition. Second, based on an in-depth analysis of the evaluation results from the HGR1 and OUHANDS datasets, it is evident that the comparison approach results in more misclassifications, especially for similar gestures with low numbers and different sizes. The number patterns of misclassified samples are quite similar, for example, HGR1: S-T, F-O, F-T, and W-R. The paper uses MSS, pre-trained, and effective data augmentation strategies


pi@raspberrypi: ~/hand_raspi4B/Pad...ddleLite-armlinux-demo/classification
文件(F) 编辑(E) 标签(T) 帮助(H)
image classification demo
Top1.A:0.984167
Top2.J:0.008898
Top3.I:0.005941

```
Top0 A - 0.990919
Top1 I - 0.004543
Top2 J - 0.003976
Preprocess time: 6.670000 ms
Prediction time: 38.334999 ms
Postprocess time: 0.156000 ms

iter 0 cost: 35.417000 ms
warmup: 0 repeat: 1, average: 35.417000 ms, max: 35.417000 ms, min: 35.417000 ms
results: 3
Top0 A - 0.991695
Top1 I - 0.004017
Top2 J - 0.003335
Preprocess time: 6.995000 ms
Prediction time: 35.417000 ms
Postprocess time: 0.154000 ms

iter 0 cost: 34.991001 ms
warmup: 0 repeat: 1, average: 34.991001 ms, max: 34.991001 ms, min: 34.991001 ms
results: 3
Top0 A - 0.984424
Top1 J - 0.007561
Top2 I - 0.007138
Preprocess time: 7.898000 ms
Prediction time: 34.991001 ms
Postprocess time: 0.149000 ms
```

pi@raspberrypi: ~/hand_raspi4B/Pad...ddleLite-armlinux-demo/classification
文件(F) 编辑(E) 标签(T) 帮助(H)
image classification demo
Top1.B:0.967373
Top2.I:0.011082
Top3.F:0.009900

```
Top0 B - 0.964463
Top1 I - 0.010781
Top2 F - 0.009716
Preprocess time: 6.525000 ms
Prediction time: 39.293999 ms
Postprocess time: 0.151000 ms

iter 0 cost: 37.125999 ms
warmup: 0 repeat: 1, average: 37.125999 ms, max: 37.125999 ms, min: 37.125999 ms
results: 3
Top0 B - 0.964306
Top1 I - 0.013297
Top2 F - 0.009408
Preprocess time: 6.374000 ms
Prediction time: 37.125999 ms
Postprocess time: 0.153000 ms

iter 0 cost: 36.191002 ms
warmup: 0 repeat: 1, average: 36.191002 ms, max: 36.191002 ms, min: 36.191002 ms
results: 3
Top0 B - 0.967373
Top1 I - 0.011082
Top2 F - 0.009900
Preprocess time: 7.074000 ms
Prediction time: 36.191002 ms
Postprocess time: 0.150000 ms
```

pi@raspberrypi: ~/hand_raspi4B/Pad...ddleLite-armlinux-demo/classification
文件(F) 编辑(E) 标签(T) 帮助(H)
image classification demo
Top1.A:0.999973
Top2.D:0.000013
Top3.C:0.000012

```
e] CPU id:3 is offine
[I  3/17 19:37:52.951 ...0rc/Paddle-Lite/lite/core/device_info.cc:461 check_cpu_onlin
e] Failed to query the online statue of CPU id:2
[I  3/17 19:37:52.951 ...0rc/Paddle-Lite/lite/core/device_info.cc:466 check_cpu_onlin
e] CPU id:2 is offine
[W  3/17 19:37:52.951 ...0rc/Paddle-Lite/lite/core/device_info.cc:1192 SetRunMode] So
me cores are offline, switch to NO BIND MODE
[W  3/17 19:37:52.952 ...e-Lite/lite/model_parser/model_parser.cc:879 LoadModelFbsFro
mFile]
warning: the version of opt that transformed this model is not consistent with curren
t Paddle-Lite version.
      version of opt:v2.9
      version of current Paddle-Lite:v2.10-rc
iter 0 cost: 32.594002 ms
iter 1 cost: 37.011002 ms
iter 2 cost: 47.124001 ms
iter 3 cost: 38.505001 ms
iter 4 cost: 34.766998 ms
warmup: 1 repeat: 5, average: 38.000201 ms, max: 47.124001 ms, min: 32.594002 ms
results: 3
Top0 A - 0.999973
Top1 D - 0.000013
Top2 C - 0.000012
Preprocess time: 8.181000 ms
Prediction time: 38.000201 ms
Postprocess time: 0.342000 ms
```

pi@raspberrypi: ~/hand_raspi4B/Paddle-Li...er/PaddleLite-armlinux-demo/classification
文件(F) 编辑(E) 标签(T) 帮助(H)
image classification demo
Top1.B:0.999979
Top2.D:0.000020
Top3.A:0.000001

```
[I  3/17 19:40:54.715 ...0rc/Paddle-Lite/lite/core/device_info.cc:466 check_cpu_online] CPU id:
3 is offine
[I  3/17 19:40:54.715 ...0rc/Paddle-Lite/lite/core/device_info.cc:461 check_cpu_online] Failed
to query the online statue of CPU id:2
[I  3/17 19:40:54.715 ...0rc/Paddle-Lite/lite/core/device_info.cc:466 check_cpu_online] CPU id:
2 is offine
[W  3/17 19:40:54.715 ...0rc/Paddle-Lite/lite/core/device_info.cc:1192 SetRunMode] Some cores a
re offline, switch to NO BIND MODE
[W  3/17 19:40:54.715 ...e-Lite/lite/model_parser/model_parser.cc:879 LoadModelFbsFromFile]
warning: the version of opt that transformed this model is not consistent with current Paddle-L
ite version.
      version of opt:v2.9
      version of current Paddle-Lite:v2.10-rc
iter 0 cost: 33.134998 ms
iter 1 cost: 34.424000 ms
iter 2 cost: 36.491001 ms
iter 3 cost: 35.744999 ms
iter 4 cost: 35.108002 ms
warmup: 1 repeat: 5, average: 34.980600 ms, max: 36.491001 ms, min: 33.134998 ms
results: 3
Top0 B - 0.999979
Top1 D - 0.000020
Top2 A - 0.000001
Preprocess time: 8.048000 ms
Prediction time: 34.980600 ms
Postprocess time: 0.322000 ms
```

FIGURE 7. Selected results of predicted hand gesture recognition of samples from the OUHANDS test set and the video images captured by the Raspberry Pi in real-time.

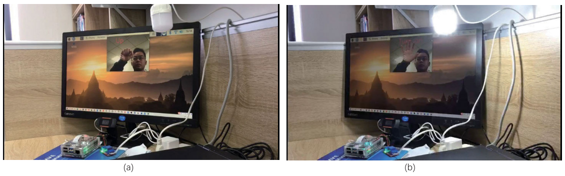


FIGURE 8. Deployment and application of a lightweight model based on Raspberry Pi. (a) Gesture-controlled desk lamp off operation. (b) Gesture-controlled desk lamp on operation.

to address the deficiencies. Third, there is a possible misunderstanding: The inference of small parameters is fast. According to Tables 1 and 2, the model parameters for ExtriDeNet are the smallest, but the inference speed is slow. For realistic applications of algorithms, accuracy, inference time, and parameter size are particularly important evaluation metrics, and it is worth considering how to balance them. There was no consideration of these issues in the comparative literature.

Compared to desktop computers, the Raspberry Pi has limited processing power and memory resources, which pose significant challenges when deploying convolutional neural networks. In this paper, we explore several approaches that can be taken to address these challenges. 1. We designed a lightweight network called LHGR-Net for hand gesture recognition. The model was initialized with pre-trained weights learned from other data, which helped reduce the training time and improve accuracy. We also optimized the code and used techniques such as batching to reduce the computational load on the Raspberry Pi. 2. We found that hardware optimization can significantly improve the Raspberry Pi's performance. This can be achieved by using a high-performance SD card, increasing memory, or using a more powerful Raspberry Pi model. 3. Depending on the application, real-time optimization can be used to dynamically adjust the model's complexity and processing requirements based on the available resources. Our extensive experiments demonstrate the effectiveness of these approaches in addressing the challenge of deploying a CNN on a Raspberry Pi.

LHGR-Net is a network architecture composed of three main components: the base network structure, the MSS, and the LAS. However, overfitting of the LHGR-Net model can indirectly affect the MSS and LAS components. To address this issue, this paper proposes several strategies, including early stopping during the training process, pre-training, dropout, and cross-validation. Additionally, we employed data augmentation techniques to increase the diversity of the training data, which can help prevent overfitting. Overall, these strategies aim to improve the generalization ability of the LHGR-Net model and prevent overfitting.

In this paper, by adding the LAS, the model can be made more accurate, but it will slow down the inference speed when too much LAS is used. The algorithm provides a better balance between inference speed and accuracy. Admittedly, there are two main limitations: 1. The large number of gesture categories easily causes confusion when controlling multiple appliances. 2. A lack of versatility in the deployment of multiple embedded terminals.

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a lightweight network for hand gesture recognition (LHGR-Net) demonstrating a new state-of-the-art in Raspberry Pi classification. To improve the recognition performance, we proposed a multiscale structure (MSS) and a lightweight attention structure (LAS) in this lightweight network. Our experimental results show that our model has great performance against challenging situations. Moreover, we employed an effective pre-trained weights technique, which plays an important role in obtaining higher recognition accuracy. The best model achieves state-of-the-art performance on the HGR1 dataset and the OUHANDS dataset. In addition, the LHGR-Net model is deployed to the Raspberry Pi, and a deployed model can be used to control home appliances. In future work, we will further refine the proposed algorithm. Firstly, we will conduct a deep analysis of the error-prone categories in the text and implement measures such as targeted data enhancement and model optimization. Secondly, we will adopt a multimodal approach to improve model performance, such as Ray-vision fusion. Finally, we aim to achieve more efficient control of smart home appliances.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

[1] T. Vuletic, A. Duffy, L. Hay, C. McTeague, G. Campbell, and M. Grealy, "Systematic literature review of hand gestures used in human computer interaction interfaces," *Int. J. Hum.-Comput. Stud.*, vol. 129, pp. 74–94, Sep. 2019.

[2] Q. Xiao, M. Qin, and Y. Yin, "Skeleton-based Chinese sign language recognition and generation for bidirectional communication between deaf and hearing people," *Neural Netw.*, vol. 125, pp. 41–55, May 2020.

[3] M. Numfu, A. Riel, and F. Noel, "Virtual reality based digital chain for creating a knowledge base of hand gestures in maintenance tasks," *Proc. CIRP*, vol. 90, pp. 648–653, Jan. 2020.

[4] P. Huu and Q. Minh, "An ANN-based gesture recognition algorithm for smart-home applications," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 5, pp. 1967–1983, 2020.

[5] S. Sharma and S. Singh, "Vision-based hand gesture recognition using deep learning for the interpretation of sign language," *Exp. Syst. Appl.*, vol. 182, pp. 115657–115669, 2021.

[6] Q. Lei, H. Zhang, Y. Yang, Y. He, Y. Bai, and S. Liu, "An investigation of applications of hand gestures recognition in industrial robots," *Int. J. Mech. Eng. Robot. Res.*, vol. 8, no. 5, pp. 729–741, 2019.

[7] D. K. Vishwakarma and R. Kapoor, "An efficient interpretation of hand gestures to control smart interactive television," *Int. J. Comput. Vis. Robot.*, vol. 7, no. 4, pp. 454–471, 2017.

[8] M. P. Tarvekar, "Hand gesture recognition system for touch-less car interface using multiclass support vector machine," in *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. (ICICCS)*, Jun. 2018, pp. 1929–1932.

[9] J. Li, S. Ray, V. Rajanna, and T. Hammond, "Evaluating the performance of machine learning algorithms in gaze gesture recognition systems," *IEEE Access*, vol. 10, pp. 1020–1035, 2022.

[10] W. Fang, Y. Ding, F. Zhang, and J. Sheng, "Gesture recognition based on CNN and DCGAN for calculation and text output," *IEEE Access*, vol. 7, pp. 28230–28237, 2019.

[11] W. Cheng, Y. Sun, G. Li, G. Jiang, and H. Liu, "Jointly network: A network based on CNN and RBM for gesture recognition," *Neural Comput. Appl.*, vol. 31, pp. 309–323, Jan. 2019.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[14] J. Wang and K. Sun, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Apr. 2021.

[15] Y. Gou, B. Li, Z. Liu, S. Yang, and X. Peng, "CLEARER: Multi-scale neural architecture search for image restoration," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 17129–17140.

[16] B. Ma, J. Zhang, Y. Xia, and D. Tao, "Auto learning attention," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1488–1500.

[17] L. Cai, Y. Fu, W. Huo, Y. Xiang, T. Zhu, Y. Zhang, H. Zeng, and D. Zeng, "Multiscale attentive image de-raining networks via neural architecture search," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 2, pp. 618–633, Feb. 2023.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[19] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[20] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1577–1586.

[21] Y. Wang, J. Yang, M. Liu, and G. Gui, "LightAMC: Lightweight automatic modulation classification via deep learning and compressive sensing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3491–3495, Mar. 2020.

[22] H. Wu, R. Lei, and Y. Peng, "PCBNet: A lightweight convolutional neural network for defect inspection in surface mount technology," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–14, 2022.

[23] D. Breland, S. Skriubakken, and A. Dayal, "Design and implementation of deep learning based contactless authentication system using hand gestures," *Electronics*, vol. 10, no. 2, pp. 182–197, 2021.

[24] D. S. Breland, S. B. Skriubakken, A. Dayal, A. Jha, P. K. Yalavarthy, and L. R. Cenkeramaddi, "Deep learning-based sign language digits recognition from thermal images with edge computing system," *IEEE Sensors J.*, vol. 21, no. 9, pp. 10445–10453, Feb. 2021.

[25] D. S. Breland, A. Dayal, A. Jha, P. K. Yalavarthy, O. J. Pandey, and L. R. Cenkeramaddi, "Robust hand gestures recognition using a deep CNN and thermal images," *IEEE Sensors J.*, vol. 21, no. 23, pp. 26602–26614, Dec. 2021.

[26] M. Y. Sikkandar, "Design a contactless authentication system using hand gestures technique in COVID-19 panic situation," *Ann. Romanian Soc. Cell Biol.*, vol. 25, pp. 2149–2159, Jan. 2021.

[27] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learning.*, 2019, pp. 6105–6114.

[28] W. Li, Z. Shang, S. Qian, B. Zhang, J. Zhang, and M. Gao, "A novel intelligent fault diagnosis method of rotating machinery based on signal-to-image mapping and deep Gabor convolutional adaptive pooling network," *Exp. Syst. Appl.*, vol. 205, pp. 117716–117727, Jan. 2022.

[29] L. Cruz, D. Junior, and J. Diniz, "Kidney tumor segmentation from computed tomography images using DeepLabv3+ 2.5 D model," *Exp. Syst. Appl.*, vol. 192, pp. 116270–116284, Apr. 2022.

[30] S. Woo, J. Park, and J. Lee, "CBAM: Convolutional block attention module," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3–19.

[31] C. Cui, T. Gao, and S. Wei, "PP-LCNet: A lightweight CPU convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1–8.

[32] *HGR1*. Accessed: Dec. 21, 2021. [Online]. Available: https://csun.aei.polsl.pl/mkawulok/gesture

[33] M. Matilainen, P. Sangi, J. Holappa, and O. Silven, "OUHANDS database for hand detection and pose recognition," in *Proc. Int. Conf. Image Process. Theory, Tools Appl.*, 2016, pp. 1–5.

[34] *Eric.Lee*. Accessed: Mar. 8, 2021. [Online]. Available: https://codechina.csdn.net/EricLee/handpose_x

[35] D. Liang, F. Yang, T. Zhang, and P. Yang, "Understanding mixup training methods," *IEEE Access*, vol. 6, pp. 58774–58783, 2018.

[36] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[37] R. F. Pinto, C. D. B. Borges, A. M. A. Almeida, and I. C. Paula, "Static hand gesture recognition based on convolutional neural networks," *J. Electr. Comput. Eng.*, vol. 2019, pp. 1–12, Oct. 2019.

[38] G. Bhaumik, M. Verma, M. C. Govil, and S. K. Vipparthi, "ExtriDeNet: An intensive feature extrication deep network for hand gesture recognition," *Vis. Comput.*, vol. 38, no. 11, pp. 3853–3866, Nov. 2022.

[39] Y. Tan, K. Lim, and C. Lee, "Hand gesture recognition via enhanced densely connected convolutional neural network," *Exp. Syst. Appl.*, vol. 175, pp. 114797–114809, Aug. 2021.

MINGYUE ZHANG received the M.S. degree in electronics and communication engineering from Hunan Normal University, in 2019. He is currently pursuing the Ph.D. degree with the School of Electronic and Information Engineering, South China University of Technology. His research interests include machine learning, human–computer interaction, and model compression.

ZHIHENG ZHOU (Member, IEEE) received the B.S. and M.S. degrees in applied mathematics and the Ph.D. degree in electronic and information engineering from the South China University of Technology, Guangzhou, China, in 2000, 2002, and 2005, respectively. He is currently a Professor with the South China University of Technology. His research interests include image processing and image and video transmission.

TIANLEI WANG received the B.S.E. degree in electrical engineering from the Beijing University of Posts and Telecommunications and the M.S. degree in signal processing from the South China University of Technology. His research interests include intelligent control and pattern recognition.

WENLVE ZHOU received the M.S. degree in information and communication engineering from Wuyi University, Jiangmen, China. He is currently pursuing the Ph.D. degree in information and communication engineering with the South China University of Technology, Guangzhou, China. His research interests include computer vision, especially transfer learning and representation learning.