Robust Hand Detection Based on Convolutional Neural Network and Attention Module

Duy-Linh Nguyen, Muhamad Dwisnanto Putro, Xuan-Thuy Vo, Tien-Dat Tran and Kang-Hyun Jo
School of Electrical Engineering, University of Ulsan,
Ulsan, Korea
Email: ndlinh301@mail.ulsan.ac.kr; dwisnantoputro@mail.ulsan.ac.kr;
xthuy@islab.ulsan.ac.kr; tdat@islab.ulsan.ac.kr; acejo@ulsan.ac.kr

***Abstract*—The hands are essential parts, helping people to contact and communicate with the surrounding environment. Hand gesture and position detection is an interesting topic in computer vision, it can be applied in several areas such as action recognition, Human-Computer Interaction, Human-Robot Interaction, control systems, etc. With the strong emergence of artificial neural networks and computer hardware devices, it becomes easier to apply hand detection in practice. Based on the benefits of convolutional neural network (CNN) and bottleneck attention module, this paper proposes a robust CNN for hand detection. As a result, the network achieved 95.52% of average precision (AP) on the Egohands test sub-dataset and 59.07 frames per second (FPS) on the Intel Core I7-4770 @ 3.40 GHz CPU in real-time testing.**

***Index Terms*—Bottleneck attention module, convolutional neural network, hand detection.**

I. Introduction

In the process of evolution, humans have been known to use their hands to hunt and gather to maintain life. On the other hand, they also use their hands to interact with each other and with the environment [1]. Thus, the hands become an extremely important sensory organ. Most actions and postures of the hand can be considered by its position and motion [2]. Decades ago, researchers focused on detecting hand position and shape based on traditional machine learning techniques. Recently, the advantage of convolutional neural networks in modern machine learning has attracted much attention from scientists to develop many artificial neural networks and novel techniques to improve accuracy. However, the hand has flexible joints and can take on many different shapes. This requires the detector to have a deep enough structure to accurately detect the hand while still ensuring the speed of the system when deployed in practice. From that observation, this paper proposes a robust CNN network for hand detection. This CNN is trained on a combination of the Hand and EgoHands datasets and then tested on the EgoHands test sub-dataset. In addition, this work also implements real-time testing with a camera connected to a CPU-based personal computer.

The main contributions of this paper are shown as:
1 - Proposed a robust and compact CNN architecture for hand detection task which consists of feature extraction and detection head modules.
2 - Develop simple real-time hand detection system that can run on the CPU with high accuracy.

The rest of the paper is organized: Section II reviews the related works to hand detection and its advantage and advantage. Section III presents the detail of the proposed method. Section IV analyzes the experiment. Finally, Section V concludes the issue and future works.

II. Related Work

A. The traditional machine learning techniques

Traditional machine learning techniques are mainly based on skin color analysis [3] and hand shape characteristics [4] to detect hand position relative to the background. The Haar-like feature detection is applied in [5] to locate the hand. [6] proposes a multi-proposal detector based on a combination of the hand shape, skin color, and context. In another approach, [7] uses an attention map to extract the hand position in the complex background. In general, these methods are easy to implement with low computational costs. However, they are limited by environmental conditions such as illumination, contrast, and color similarity between foreground and background. In addition, to achieve the desired accuracy, traditional methods also require careful preprocessing steps for feature extraction.

B. The CNN-based techniques

Nowadays, convolutional neural networks are widely applied in object detection topics including hand detection. [8] proposes a one-stage CNN network that detects hand position and hand orientation. The Faster-RCNN network is used in [9] to detect the position of the driver's hand on the steering wheel and combines with the hand skin segmentation technique in [10] to locate the hand position in unconstrained scenarios. The ability to detect the hand of these methods is quite weak due to the lack of benchmark datasets and it is not covering the contexts of the hand in the natural environment. [2] proposes a hand mask and hand orientation detection based on the Mask-CNN network and new attention techniques. This method achieves high performance but requires complex computational techniques and is unsuitable for low computing devices.

III. Proposed Methodology

The proposed hand detection network is shown in Fig. 1. The network is divided into two modules: feature extraction and detection head.

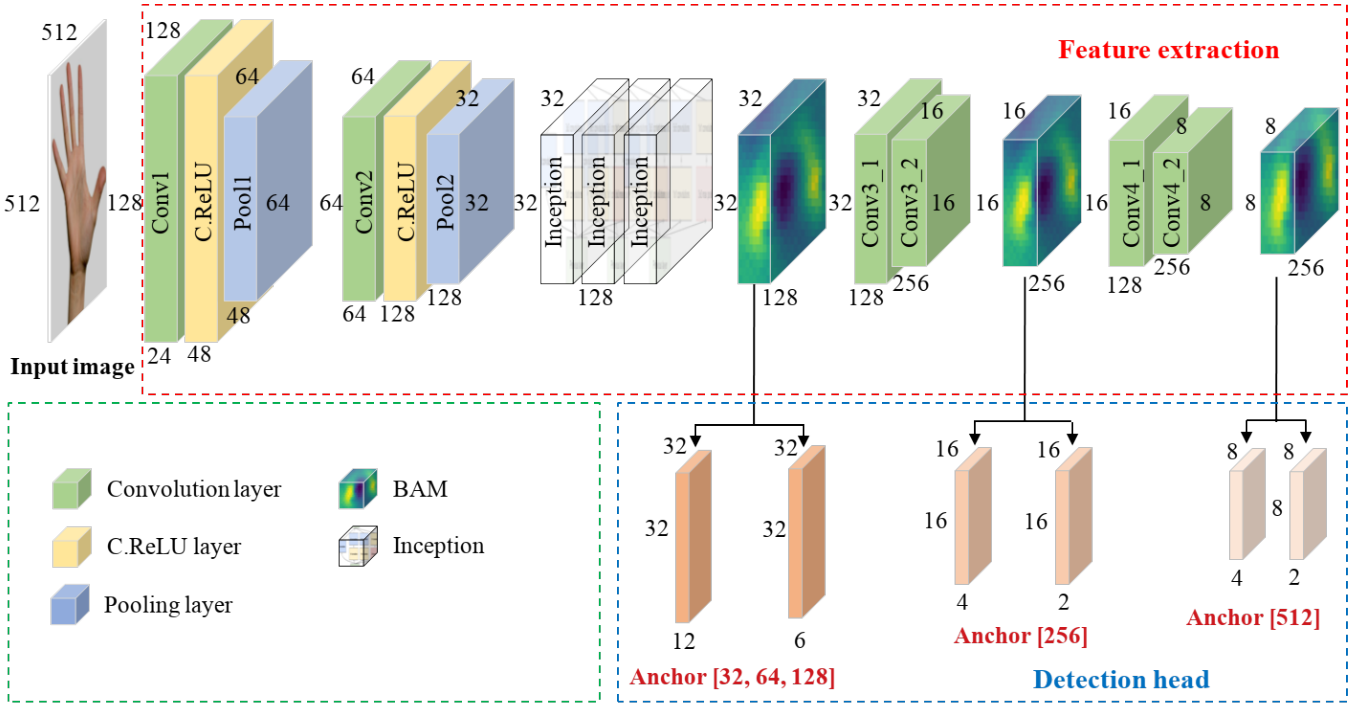

Fig. 1. The proposed hand detection network. The network consists of feature extraction and detection head modules.

A. Feature extraction

The feature extraction is a combination of convolutional layer groups, inception module [11], and bottleneck attention Module (BAM) [12]. First, two groups of convolution layers are used. Each group consists of a convolution layer with kernel size 3×3, a concatenated rectified linear unit (C.ReLU) [13], and a pooling layer. The input image, when going through these two convolutional groups, will extract basic context information about the object and reduce the size to 32×32 with 128 channels. The use of C.ReLU is to ensure full object information extraction during the training. In the next phase, the inception block with three inception modules is used immediately after the first two groups of convolutional layers. Each inception consists of convolution layer branches according to the width of the network with kernels of different sizes. This structure makes it possible for the network to capture object information with different receptive fields. These streams of information are then concatenated together to enrich the object information. The architecture of the inception module is described in Fig. 2.

At the end of the feature extraction are the BAMs combined with the remaining convolution groups. The attention mechanism in each BAM helps the network to focus on the strong features of the object to distinguish them from the background. Fig. 3 shown the architecture of bottle attention module. This module consists of two sub-modules: channel attention and spatial attention. Given the input feature map $F \in \mathbb{R}^{H \times W \times C}$ is the input of BAM. The output feature map $F^{'} \in \mathbb{R}^{H \times W \times C}$ is described as follows:

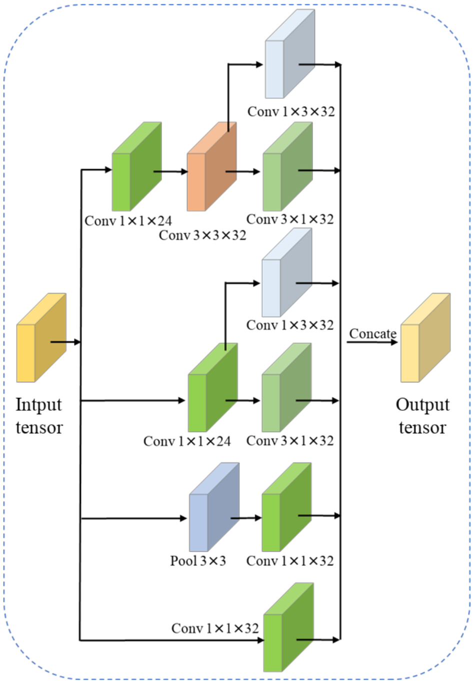

Fig. 2. The architecture of the inception module.

$$F^{'} = F + F \otimes M(F), \tag{1}$$

where $\otimes$ denotes element-wise multiplication. $F(M)$ is the

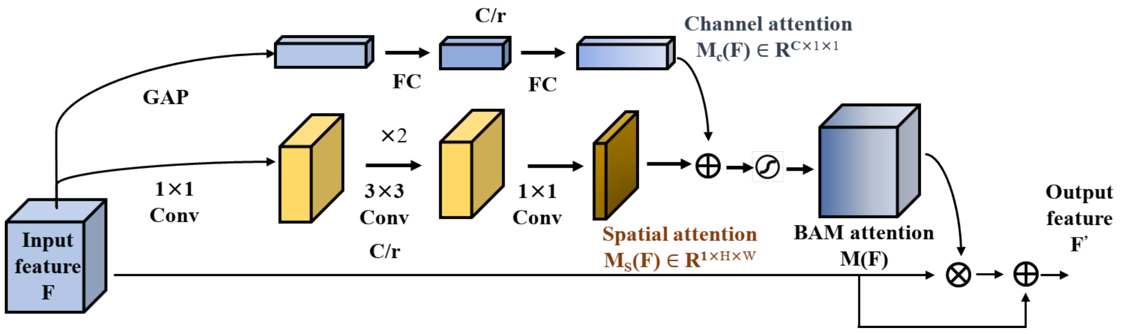

Fig. 3. The architecture of bottleneck attention module. It consists of the channel attention module and spatial attention module.

attention map from BAM and it is computed as:

$$M(F) = \sigma(M_c(F) + M_s(F)), \tag{2}$$

with σ is sigmoid function, M_c is channel attention module, and M_s is spatial attention module. M_c and M_s are defined as:

$$\begin{aligned} M_c(F) &= BN(MLP(GAP(F)) \\ &= BN(W_1(W_0 GAP(F) + b_0) + b_1), \\ M_s(F) &= BN(f_3^{1\times1}(f_2^{3\times3}(f_1^{3\times3}(f_0^{1\times1}(F))))), \end{aligned} \tag{3}$$

in which, $W_0 \in \mathbb{R}^{C/r \times C}, b_0 \in \mathbb{R}^{C/r}, W_1 \in \mathbb{R}^{C \times C/r}, b_1 \in \mathbb{R}^{C}$, $f_i^{1\times1}$ is 1×1 convolution layer, $f_i^{3\times3}$ is 3×3 dilated convolution layer with dilation rate of 4, BN is batch normalization, MLP is Multi-Layer Perceptron, GAP is global average pooling.

The last convolutional layers continue the feature extraction process to generate the feature maps to a suitable size as input for the detectors with 32×32, 16×16, and 8×8 feature map sizes, respectively.

B. Detection head

This work uses three detection layers for different scales. Each detection applies two 3×3 sibling convolutions for classification and bounding box regression. These layers captures the feature maps from 32×32, 16×16, and 8×8. Those are output feature maps from the feature extraction. The detection head uses square anchors with sizes 32, 64, 128 for small size hands, 256 for medium size hand and 512 for large size hand. Each detector generates a location offset with four-dimensional vector (x, y, w, h) and a label classification with two-dimensional vector ($hand$ or $not - hand$).

C. Loss function

The loss function is the sum of the 2-classes softmax loss for classification task and the smooth L1 loss for regression task. The loss function is defined as follows:

$$\begin{aligned} L\left(\{pro_i\}, \{co_i\}\right) = \frac{1}{N_{cls}} \sum_i L_{cls}\left(pro_i, pro_i^*\right) + \\ \lambda \frac{1}{N_{reg}} \sum_i pro_i^* L_{reg}\left(co_i, co_i^*\right), \end{aligned} \tag{4}$$

Where i is an anchor index. pro_i and pro_i^* are the predicted probability of $i - th$ anchor being an object and ground truth label (0 or 1), respectively. co_i and co_i^* are the center point coordinates (x, y) and dimension (w, h) of the predicted and ground truth bounding box, respectively. The classification loss $L_{cls}\left(pro_i, pro_i^*\right)$ using the softmax-loss defined in Eq. (5). The classification loss $L_{reg}\left(co_i, co_i^*\right) = \sum_{i \in \{x,y,w,h\}} Smooth(co_i - co_i^*)$ with $Smooth$ is the Smooth $L1$ loss defined in Eq. (6). N_{cls} (mini-batch size) and N_{reg} (number of anchor location) are two normalized terms, and λ is a balancing parameter.

$$L_{cls}\left(pro_i, pro_i^*\right) = -\sum_{i \in Pos} x_i^{pro} log\left(pro_i\right) - \sum_{i \in Neg} log\left(pro_i^0\right), \tag{5}$$

Where $x_i^{pro} = \{0, 1\}$ is indicator for matching the $i - th$ anchor bounding box to ground-truth bounding box of object i, pro_i and pro_i^0 are the probabilities for object and non-object classification, respectively.

$$Smooth(x) = \begin{cases} 0.5x^2 & if\, |x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \tag{6}$$

IV. EXPERIMENT

A. Dataset

A combination of Hand [14] and Egohands [15] datasets is used for training and the Egohands test sub-dataset is used in the testing phase. The Hand dataset contains 4,069 images collected from various public image sources on the internet. The Egohands dataset has 4,782 images taken with Google glass through indoor and outdoor environments and various activities such as solving puzzles, playing Jenga, cards, and chess. All image annotations in the training set are converted to PASCAL VOC format by Python code. In total, the entire train set consists of 8,373 images, and the test sub-dataset consists of 478 images.

B. Experimental setup

The proposed network are trained on Tesla-V100 GPU and tested on Intel Core I7-4770 CPU @ 3.40 GHz, 8GB of RAM. The training phase applies several basic configurations

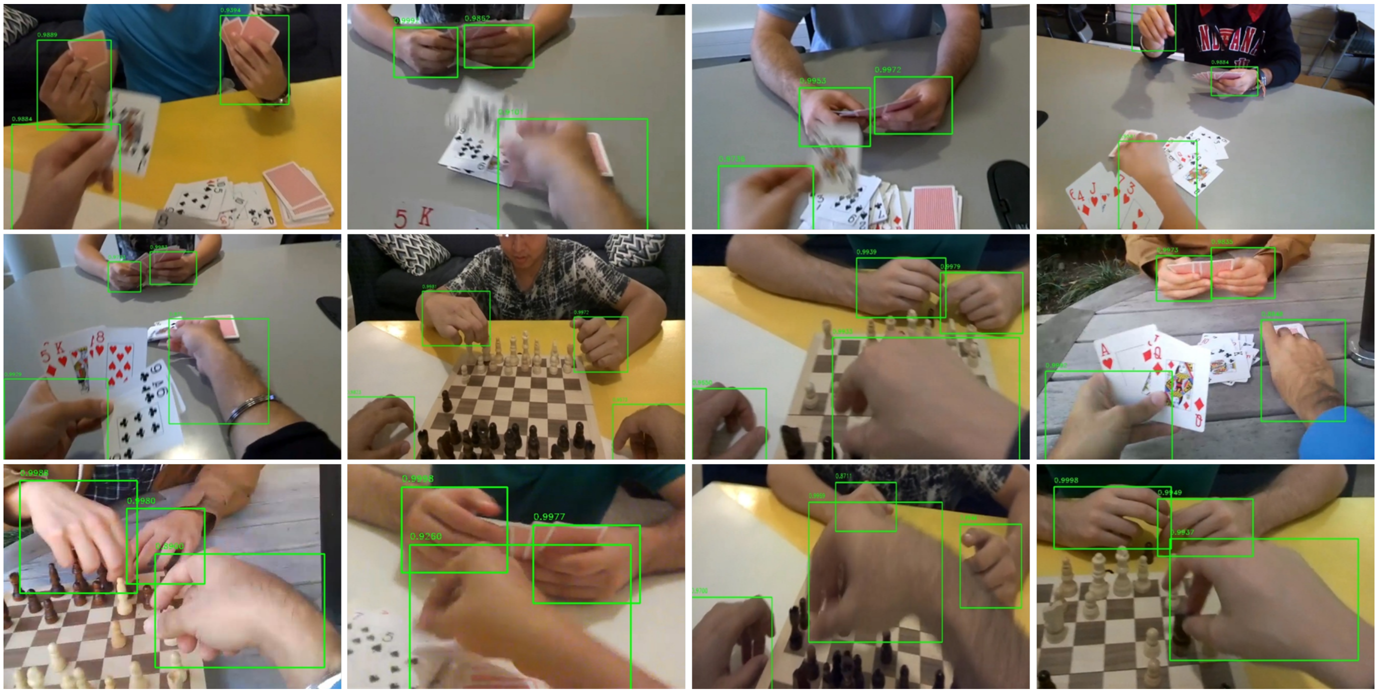

Fig. 4. The qualitative results on EgoHands test sub-dataset. The number on each bounding box presents the confidence core.

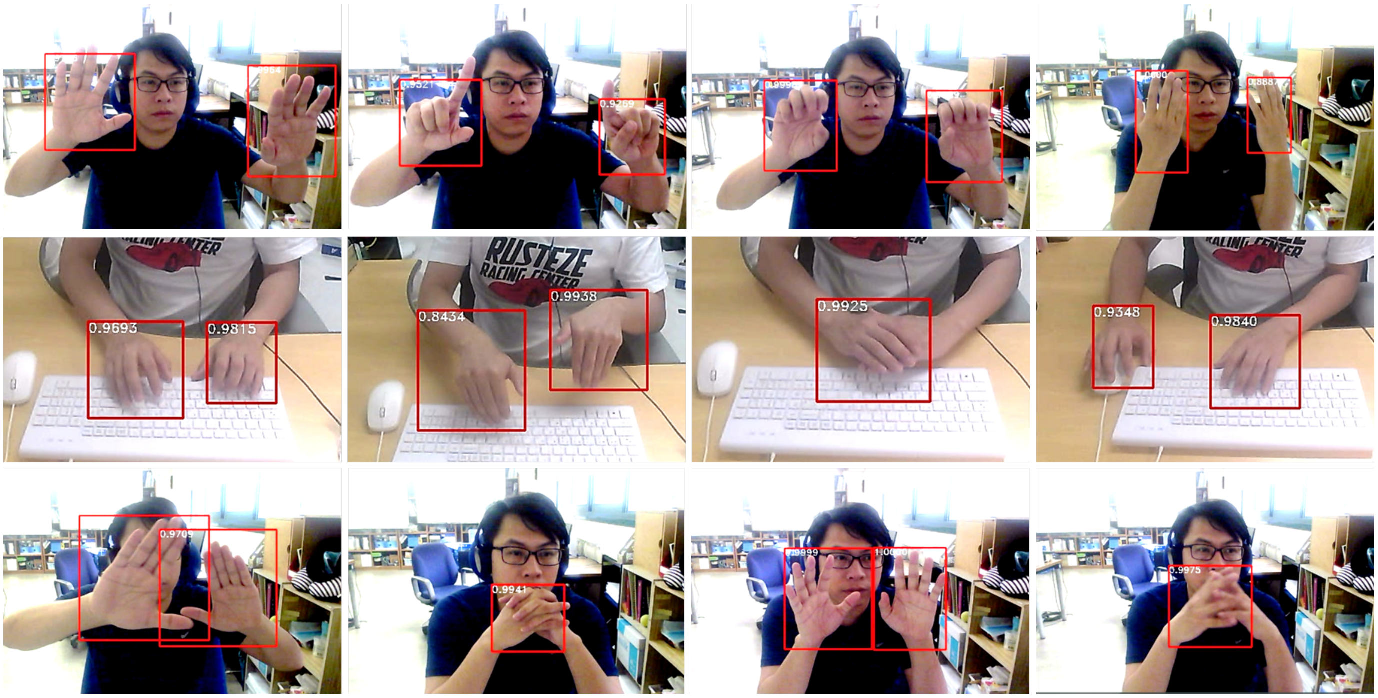

Fig. 5. The result testing in real-time on a camera connect to a CPU-Based personal computer. The number in each bounding box presents the confidence core.

in training object detection networks. Specifically, the network is trained with 300 epochs, batch size 32, weight decay of 5×10^{-4}, momentum of 0.9, a learning rate of 10^{-3}, and set λ=10. In order to optimize the weight updating process, the stochastic gradient descent optimization method is used. The non-maximum suppression (NMS) with intersection over union (IoU) threshold of 0.5 is also used to generate the best-fit bounding box.

C. Experimental results

The proposed network is trained and tested on the image dataset as mentioned above. It was also tested in real-time with a conventional camera connected to a CPU-based personal computer. As a result, the network achieved 95.52 % of AP on the test sub-dataset of the EgoHands dataset. This result can be compared with other mobile networks in the object detection topic such as SSD, FaceBoxes, and the YOLO network family.

TABLE I

COMPARISON RESULT OF HAND DETECTION NETWORK ON THE EGOHANDS TEST SUB-DATASET. THE RED COLOR INDICATES THE BEST RESULT.

Network	Parameters	Average Precision (%)
Proposed	**1,107,198**	**95.52**
Network in [16]	944,150	93.32
SSD512 [16]	24,358,668	90.30
SSD300 [16]	23,745,908	90.23
FaceBoxes [16]	1,007,330	78.39
YOLOv3-Tiny [17]	8,669,876	98.50
YOLOv4-Tiny [17]	3,061,636	98.30
YOLOv5s [17]	7,063,542	**98.60**

The results in Table 1 show that the proposed network outperforms the networks implemented in [16] and is only 113,048 parameters larger than the best detector. Besides, this experiment also compared the results obtained with tiny networks in the YOLO family including YOLOv3-Tiny, YOLOv4-Tiny, and YOLOv5s. The comparison proves that the proposed network is more compact than YOLOv3-Tiny, YOLOv4-Tiny, and YOLOv5s, but only less than 2.98, 2.78, and 3.08 of AP, respectively. The network can detect hands in different poses, sizes, and shapes, even in overlap and blur conditions. The qualitative results are shown in Fig. 4.

The real-time system testing under the same conditions as the experiments in [16] also shows that the proposed network achieved a speed of 59.07 frames per second (FPS). This result is dominant when compared to the best performing network (FaceBoxes network) with a difference of 24.62 FPS. The results presented in Fig. 5 demonstrate that the proposed network detects hands well in different contexts, postures on the front and back of the hand, and interlaced hands. However, with bad conditions such as low light or overexposure, fast hand movement speed, and camera angle greatly affect the detection accuracy and processing speed of the system.

D. Ablation study

This work conducted several experiments with different attention mechanisms. To do that, the BAM module in the proposed network is replaced by the squeeze-and-excitation (SE) attention module [18] and convolutional block attention module (CBAM) [19], respectively. The evaluation results on the Egohands test sub-dataset shown in Table 2 present that when the network uses SE and CBAM, the AP only reaches 92.16% and 94.13%, respectively. In contrast, when using BAM, the network increases by about 1K parameters but reaches the highest AP at 95.52%. Therefore, BAM is chosen to design the hand detection network in this paper.

TABLE II

ABLATION STUDY WITH DIFFERENT ATTENTION MODULE ON EGOHANDS TEST SUB-DATASET. THE RED COLOR INDICATES THE BEST RESULT.

Attention module	**Proposed network**		
SE	✓		
CBAM		✓	
BAM			✓
Parameters (Million)	1,088	1,107	1,108
AP (%)	92.16	94.13	**95.52**

V. CONCLUSION AND FUTURE WORK

This paper proposed a compact and robust convolutional neural network for hand detection including two sub-modules, feature extraction and detection head. The network is designed based on the advantages of convolutional layers, C.ReLU module, inception module, and BAM attention mechanism. The proposed network is trained and evaluated on image datasets with high accuracy and with negligible latency when testing in a real-time system. In the future, this hand detection network will be further developed to be deployed in systems such as hand tracking, hand control, and game control.

ACKNOWLEDGMENT

This work was supported by "Region Innovation Strategy (RIS)" through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE)(2021RIS-003).

REFERENCES

[1] M. J. Kim, M. Choi, Y. Kim, F. Liu, H. Moon, J. Koo, and H. Choi, "Exploration of unknown object by active touch of robot hand," *International Journal of Control, Automation and Systems*, vol. 12, pp. 406–414, 04 2014.

[2] S. Narasimhaswamy, Z. Wei, Y. Wang, J. Zhang, and M. Hoai, "Contextual attention for hand detection in the wild," *CoRR*, vol. abs/1904.04882, 2019.

[3] H. Cooper and R. Bowden, "Large lexicon detection of sign language," in *Human–Computer Interaction* (M. Lew, N. Sebe, T. S. Huang, and E. M. Bakker, eds.), (Berlin, Heidelberg), pp. 88–97, Springer Berlin Heidelberg, 2007.

[4] M. Kolsch and M. Turk, "Robust hand detection," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pp. 614–619, 2004.

[5] E. u. Haq, S. J. H. Pirzada, M. W. Baig, and H. Shin, "New hand gesture recognition method for mouse operations," in *2011 IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1–4, 2011.

[6] A. Mittal, A. Zisserman, and P. Torr, "Hand detection using multiple proposals," pp. 75.1–75.11, 01 2011.

[7] P. Pisharady, P. Vadakkepat, and A. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, 02 2013.

[8] X. Deng, Y. Zhang, S. Yang, P. Tan, L. Chang, Y. Yuan, and H. Wang, "Joint hand detection and rotation estimation using cnn," *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 1888–1900, 2018.

[9] T. H. N. Le, Y. Zheng, C. Zhu, K. Luu, and M. Savvides, "Multiple scale faster-rcnn approach to driver's cell-phone usage and hands on steering wheel detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 46–53, 2016.

[10] K. Roy, A. Mohanty, and R. R. Sahay, "Deep learning based hand detection in cluttered environment using skin segmentation," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 640–649, 2017.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.

[12] J. Park, S. Woo, J. Lee, and I. S. Kweon, "BAM: bottleneck attention module," *CoRR*, vol. abs/1807.06514, 2018.

[13] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," *CoRR*, vol. abs/1603.05201, 2016.

[14] A. Mittal, A. Zisserman, and P. Torr, "Hand detection using multiple proposals," pp. 75.1–75.11, 01 2011.

[15] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1949–1957, 2015.

[16] D.-L. Nguyen, M. D. Putro, and K.-H. Jo, "Hand detector based on efficient and lighweight convolutional neural network," in *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, pp. 432–436, 2020.

[17] G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, tkianai, A. Hogan, lorenzomammana, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Y. , changyu98, M. Wang, N. Gupta, O. Akhtar, PetrDvoracek, and P. Rai, "ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements," Oct. 2020.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, vol. abs/1709.01507, 2017.

[19] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," 2018.