ShuffleNetv2-YOLOv3: A real-time recognition method of static sign language based on a lightweight network

Shiniu Sun
Taiyuan University of Technology
Lisheng Han
Taiyuan University of Technology
Jie Wei
Taiyuan University of Technology
Huimin Hao (haohuimin@tyut.edu.cn)
Taiyuan University of Technology
Jiahai Huang
Taiyuan University of Technology
Wenbin Xin
Taiyuan University of Technology
Xu Zhou
China Coal Science & Industry Group Taiyuan Research Institute Co. LTD
Peng Kang
China Coal Science & Industry Group Taiyuan Research Institute Co. LTD

Research Article

Keywords: YOLOv3, convolutional neural network, gesture recognition, object detection, network lightweight

Posted Date: April 26th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1582533/v1

Additional Declarations: No competing interests reported.

Abstract

In order to better meet the communication needs of the hearing impaired and the general public, it is of great significance to recognize sign language more quickly and accurately in the embedded platforms and mobile terminals. A sign language recognition method based on the ShuffleNetv2-YOLOv3 lightweight model was proposed. By Using ShuffleNetv2 as the backbone network of the YOLOv3 model, the ShuffleNetv2-YOLOv3 method improved the recognition speed by the lightweight network with the CIoU loss function and more less detection layers; kept the recognition accuracy of similar gestures that were difficult to accurately recognize. Statistical analysis of the self-made sign language images was carried out to evaluate the recognition effectiveness of the model by F1 score and mAP value, and to compare it with YOLOv3-tiny, SSD, Faster-RCNN, MobileNetv2-YOLOv3, and YOLOv4-tiny models, respectively. The experimental results show that the proposed ShuffleNetv2-YOLOv3 model achieved a good balance between the accuracy and speed of gesture detection under the premise of model lightweight. The F1 score and mAP value of the ShuffleNetv2-YOLOv3 model were 99.1% and 98.4%, respectively, and the target detection speed on the GPU can reach 54 frames per second, which is better than other models. The ShuffleNetv2-YOLOv3 sign language recognition method is conducive to quick, real-time, and similar static sign language gesture recognition, which lays a good foundation for real-time dynamic gesture recognition.

1 Introduction

For people with language and hearing impairments, sign language is the main communication tool. Sign language recognition uses algorithms to recognize gesture sequences and then realizes semantic expression in text or voice. Such as Support Vector Machine (SVM) [1], FSM (Finite State Machine) [2], Hidden Markov Model (TMHMM) based on Hybrid Meta-bundling [3], Hough Transform and Neural Network [4], convolutional neural network and GPU acceleration [5], CNN classifier [6], etc., have been used for gesture recognition and achieved good recognition results.

With the development of target detection technology, scholars converted the classification problem of target recognition into target detection problems, RCNN (Region-Convolutional Neural network) [7], Fast-RCNN [8], Faster-RCNN [9], and other algorithms were proposed to obtain higher recognition accuracy. However, the methods mentioned before have disadvantages, such as complex networks, large models, and slow running speeds, which make it difficult to achieve high-speed target recognition on the mobile terminal.

For mobile target detection, a number of miniaturized deep neural networks, such as the YOLO algorithm [10] for one-stage detection were proposed. And then, the research team proposed the YOLOv2 algorithm [11] which further improved the recognition accuracy; the YOLOv3 [12] algorithm was proposed in 2018 to improve the recognition rate of small targets. Liu proposed the SSD (Single Shot MultiBox Detector) [13], however, the detection speed of this network is slow when applied to the mobile terminal, because the GPU computing speed of the mobile terminal is much lower than that of the PC terminal. In order to meet the needs of mobile devices, some lightweight CNN networks such as MobileNet [14] and ShuffleNet [15] have been proposed, which have a good balance between speed and accuracy.

For mobile terminal gesture recognition, it is necessary to resolve the contradiction between model lightweight, accuracy, and recognition speed. In response to this problem, a real-time static sign language recognition method based on the lightweight network ShuffleNetv2-YOLOv3 is proposed. This method has little requirements on equipment computing capability and stable detection effect. Under the premise of ensuring accuracy and detection speed, the model size is significantly reduced, making it easier to deploy on mobile terminals or embedded platforms.

2 Related Work

Xu Liukai[16] et al. proposed a new architecture that uses convolutional neural network (CNN) to classify the energy kernel phase diagram of sEMG signal through modeling and analysis of sEMG signal energy kernel characteristics, so as to recognize human gestures in real time.Huang[17] propose a finger-emphasized multi-scale descriptor for hand gesture representation, this method is robust to noises, hand articulations and rigid transformations. Tan[18] et al. proposed a convolutional neural network (CNN) integrated with spatial pyramid pooling (SPP), called CNN-SPP, for vision-based gesture recognition. SPP extends the input features to fully connected layers, enabling the network to better distinguish between different gestures. Fan Jingjing et al.[19] proposed a lightweight gesture recognition algorithm based on the YOLOv4-tiny network structure, aiming at the problems that the lightweight target detection network has insufficient ability to extract static gesture features, high false detection rate and missed detection rate. This can achieve accurate classification and real-time detection. and has better recognition effect for small-scale gestures. We use the lighter feature extraction network shufflenetv2[20] as the backbone network. ShufflenetV2 uses channel shuffling to improve the exchange of information flow between channels, and further considers the actual speed of hardware.

3 Proposed Method

3.1 ShuffleNetv2-YOLOv3 Network

As shown in Fig. 1, the proposed ShuffleNetv2-YOLOv3 is an end-to-end object detection framework based on regression. Different from the traditional YOLOv3 with Darknet-53 as the backbone network, the ShuffleNetv2-YOLOv3 used ShuffleNetv2 as the backbone network and its detection layer still follows the detection mechanism of YOLOv3. In terms of feature extraction: the stage in backbone uses depth-separable convolution to extract features, effectively reducing the number of parameters and computing costs. In the detection layer: the network depth is increased by multi-layer feature fusion and point convolution, which improves the detection accuracy of the model. Reducing the number of detection layers while maintaining model accuracy again reduces the model size. The accuracy of the model for gesture recognition is further improved by replacing the loss function.

3.2 Loss function for ShuffleNetv2-YOLOv3

The traditional YOLOv3 bounding box loss is calculated using MSE, but this method ignores the intersection ratio of detection frames – IoU [21]. The IoU calculation expression is shown in Eq. (1) and the bounding box loss is shown in Eq. (2). In target detection, the IoU value of the bounding box regression is often used as an evaluation metric for object detection, which can reflect the overlap effect of the predicted detection frame and the real detection frame, but suffers from the following shortcomings:

(1) If IoU is used as a loss function, when the bounding box and the real box do not overlap, the IoU value is 0, which does not reflect the distance between the two, and there is no gradient back propagation, and no learning training can be performed.

(2) IOU cannot distinguish the different alignment between two objects. More specifically, the IOU of two overlapping objects with the same intersection level in different directions will be exactly equal.

In CVPR2019, paper [22] proposes GIoU, which solves the problem when the bounding boxes do not overlap based on IoU. The GIoU calculation expression is shown in Eq. (4). The bounding box loss is shown as:

$$IoU = \frac{|(A \cap B)|}{|(A \cup B)|} (1)$$

$$L_{IoU} = 1 - IoU \, (2)$$

$$GIoU = IoU - \frac{|C\,(A \cap \beta)|}{|C|} (3)$$

$$L_{GIoU} = 1 - GIoU \, (4)$$

Where, A and B are any two rectangular parallelepipeds, and C is the smallest bounding rectangle surrounding A and B.

However, GIoU failed to completely solve another problem, so based on IOU, Zheng Z[23] proposed DIoU taking into account the distance between the centroids of the two frames. the DIoU bounding box loss is shown in Eq. (5). CIoU was also proposed to solve the problem of inconsistent width and height of the two frames when the predicted frame overlaps with the real frame and the centroids also overlap. the CIoU bounding box loss is shown in Eq. (6).

$$L_{DIoU} = 1 - IoU + R(B, B^{gt}) \, (5) \quad R\left(B, B^{gt}\right) = \frac{\rho^2\left(b, b^{gt}\right)}{c^2}$$

Where $R(B, B^{gt})$ is defined as the penalty term of the prediction frame B and the target frame B^{gt}, $b$and b^{gt} represent the center points of B and B^{gt} respectively, ρ represents the Euclidean distance, and c represents the diagonal distance of the smallest outer rectangle.

$$L_{CIoU} = 1 - IoU + R\left(B, B^{gt}\right) + \alpha v\,(6)$$

Where αv is the influence factor and α is the parameter used to balance the ratio. The calculation formula is shown as:

$$a = \frac{v}{(1 - IoU) + v}(7)$$

Where v is used as a parameter to measure the consistency of the aspect ratio. The calculation formula is shown as:

$$\frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{n^{9t}} - arctan\frac{w}{h}\right)^2 (8)$$

4 Model Training

4.1 Experimental configuration and datasets

All the experiments were performed on a Ubuntu 16.04 LTS desktop running on Intel Core i9-9900x, 32G DDR4, and GeForce RTX2080Ti.The gesture data used in this experiment include public and homemade databases of sign language images. The homemade sign language images were captured by the kinectv2 camera and had been converted into 640×480 pixel images. There are 2500 images in the dataset, and 80% of the data are randomly selected as the training set through the python algorithm. The rest is divided into test set and validation set according to 8:2. The datasets were manually annotated using the LabelImg tool and saved in PASCAL VOC format. The gestures contained in the data have eight categories: C, D, Y, G, Q, H, L, and O, as shown in Fig. 2. The data set consists of 12 different left-handed and right-handed sign language gestures of different people. All eight categories of gestures were selected from the American Sign Language (ASL).

The public datasets is Microsoft Kinect and Leap Motion datasets and Creative Senz3D datasets [24, 25]. The Microsoft Kinect and Leap Motion datasets were

taken at the Department of Information Engineering,

University of Padova and contains images of 14 different people using Leap-Motion and Kinect devices to acquire gestures performed by each person, each person performing 10 different gestures and each person repeating 10 times. The Microsoft Kinect and Leap Motion data are shown in **Fig. 3**.

The Creative Senz3D datasets [26, 27] contains several different static gestures that were acquired with the Creative Senz3D camera from four different people performing 11 different gestures each, repeated 30 times each, for a total of 1320 samples. The Creative Senz3D datasets is shown in **Fig. 4**

4.2 Evaluation criterion

In this experiment, mAP (mean average precision), F1 score, and model size were used as criteria to evaluate the model:

$$P=\frac{TP}{TP+FP}(9)$$

$$R=\frac{TP}{TP+FN}(10)$$

$$F_1=\frac{2PR}{P+R}(11)$$

$$AP=\int_0^1 P\left(R\right)dR\,(12)$$

$$\text{mAP}=\frac{\sum_{i=1}^{n}AP}{\text{n}}(13)$$

Where, P is the precision rate; R is the recall rate; TP (true positive) is the number of true positive samples; FP (false positive) is the number of false positive samples; FN (false negative) is the number of false negative samples; n is the number of samples in all categories. AP is the average precision.

4.3 Training parameters

In the training phase, the size of the anchor frame was recalculated for the gesture datasets using the K-means clustering algorithm, the model parameters were modified by migration learning on the COCO datasets using a back-propagation algorithm to gradually reduce the loss function; the optimization was performed using the SGD (Stochastic Gradient Descent) optimizer. The hyperparameters in training are set as follows: the momentum factor is 0.937, the initial learning rate is 0.01, the decay coefficient is 0.0005, and the images were transformed into 416×416 pixel images during training. The Batch size is 32, a total of 300 epochs are trained. This research adopts the method of comparative experiment.

5 Results And Discussion

5.1 Detection results of different backbone network models

In order to reflect the advantages of ShuffleNetv2 as the backbone network and to select the best channel count for the ShuffleNetv2 model. Therefore, the comparison is between the ShuffleNetv2-YOLOv3 model with different number of channels and the YOLOv3 model with IoU loss function. F1 score, mAP and detection speed are tested several times to find the average value. The comparison results and training time of different models are shown in Table 1.

Darknet53 has a large amount of calculation parameters and high storage space requirements. ShuffleNetV2 considers the actual speed at the target hardware. It uses channel shuffling to improve the exchange of information flow between channels. Using Depthwise convolution and 1 * 1 convolution to reduce the amount of parameters. So YOLOv3 had the longest training time and exceeded the ShuffleNetv2-YOLOv3-2X channel version by 0.585 hours, while the four channel count versions of ShuffleNetv2-YOLOv3 had a small difference in training time, with the time increasing slightly with the number of channels 0.88, 0.89, 0.895 and 0.911 hours respectively.The training time was significantly shorter compared to YOLOv3.

The 1.5X channel version ShuffleNetv2-YOLOv3 was tested and although the model size and Flops were slightly larger than the 0.5X and 1X channel versions, the model size was only 9.2MB and it outperformed all other types in terms of mAP and F1 score. Therefore, the 1.5X channel count ShuffleNetv2 is selected as the backbone network for the mode.

Table 1
Detection results of different backbone network models

Network models	Output channels	F1 score/ %	mAP/ %	Weight size/MB	Flops/G	FPS	Hours
Darknet53-YOLOv3		98.4	98.8	123.5	155.2	42	1.496
ShuffleNetv2-YOLOv3	0.5X	97.4	98.8	4.8	4.1	50	0.88
ShuffleNetv2-YOLOv3	1X	98.4	98.8	6.7	5.9	51	0.89
ShuffleNetv2-YOLOv3	1.5X	98.4	99.1	9.2	8.4	51	0.895
ShuffleNetv2-YOLOv3	2X	98.1	99.1	15.5	13.3	48	0.911

Table 2
Model performance under different detection scales

Network models	Output channels	Detection scale	mAP/ %	Weight size/MB	F1 score/ %	FPS
ShuffleNetv2-YOLOv3	1.5X	3	99.1	9.2	98.4	51
ShuffleNetv2-YOLOv3	1.5X	2	99.1	8.9	98.3	54

Table 3
Test results using G/D/CIoU loss function

Network models	loss function	mAP/ %	Weight size/MB	F1 score/ %	FPS
ShuffleNetv2-YOLOv3	IoU	99.0	8.9	98.0	54
ShuffleNetv2-YOLOv3	GIoU	98.9	8.9	98.5	53
ShuffleNetv2-YOLOv3	CIoU	99.1	8.9	98.4	54
ShuffleNetv2-YOLOv3	DIoU	99.0	8.9	98.3	50

5.2 Model performance under different detection scales

The detection layer of YOLOv3 adopts a structure similar to FPN (Feature Pyramid Network), which can merge feature maps of different levels.

In order to further reduce the model size, ShuffleNetv2-YOLOv3 is trained with two detection scales and compared with the original three detection scales. The comparison results are shown in Table 2.

The size of the model with two detection scales is 8.9MB, which is 0.3MB smaller than that of the model with three detection scales, and the detection speed is increased by 3 frames per second compared with the model with three detection scales. Therefore, two detection scales are adopted as the final structure.

5.3 Test results using G/D/CIoU loss function

Replace the ShuffleNetv2-YOLOv3 model loss function, and average the results of multiple training and test models. The comparison results are shown in the Table 3.

The CIoU loss solves the problem of inconsistent width and height of the two frames when the predicted frame overlaps with the real frame and the centroids also overlap. It is more inclined to optimize in the direction of the increasing overlapping areas. From the data in the table, it can be verified that the mAP of the CIoU loss function of ShuffleNetv2-YOLOv3 can reach 99.1%, which is the highest among the four loss functions and the F1 score can reach 98.5%. They have the same model size. Therefore, the version of CIoU loss function of ShuffleNetv2-YOLOv3 is adopted as the final network structure. The detection results are shown in Fig. 5

5.4 Model effects on public datasets

To validate the effectiveness of the ShuffleNetv2-YOLOv3-1.5X-CIoU model, the Darknet53-YOLOv3

model was carried out on the datasets Microsoft Kinect and Leap Motion and Creative Senz3D as well. The comparison results are shown in Table 4.

The table verifies that there is no difference in the mAP and F1 score metrics between the two models in the Creative Senz3D dataset, But the size of the ShuffleNetv2-YOLOv3 model is 8.9MB, which is about

1/14 of the YOLOv3 model. The detection speed on GPU is 65 frames per second, which is 1.38 times faster than the YOLOv3 model. The detection speed of the ShuffleNetv2-YOLOv3 model on the CPU is 12 frames per second, the detection speed of the YOLOv3 model is 5 frames per second. The detection speed of the ShuffleNetv2-YOLOv3 model on the CPU is 2.4 times faster than that of the YOLOv3 model.

In the Microsoft Kinect and Leap Motion dataset, the detection image size is 1280×960, which is larger than the 640×480 in the Creative Senz3D dataset, so the detection speed of ShuffleNetv2-YOLOv3-1.5X-CIoU is only 15 frames per second on the GPU. The detection speed of YOLOv3 is only 12 frames per second. The improved network detection speed is 1.5 times faster than YOLOv3. The detection speed of ShuffleNetv2-YOLOv3-1.5X-CIOU is 8 frames per second on the CPU, while YOLOv3 detects 4 frames per second. The improved network detection speed is twice as fast as YOLOv3. This verifies the effectiveness of the ShuffleNetv2-YOLOv3-1.5X-CIoU improvement.

Table 4
Model effects on public datasets

Network models	**Dataset**	**mAP/ %**	**Weight size/MB**	**F1 score/ %**	**FPS**
ShuffleNetv2-YOLOv3-1.5X-CIoU	Creative Senz3D	99.5	8.9	98.0	65
Darknet53-YOLOv3	Creative Senz3D	99.5	123.5	98.0	52
ShuffleNetv2-YOLOv3-1.5X-CIoU	Microsoft Kinect and Leap Motion	99.6	8.9	97.9	15
Darknet53-YOLOv3	Microsoft Kinect and Leap Motion	99.6	123.5	97.8	12

5.5 Comparison of different network models

The recognition accuracy and detection speed on GPU of SSD, YOLOv3-tiny, ShuffleNetv2-YOLOv3-1.5X-CIOU and MobileNetv2-YOLOv3 under the self-made dataset were compared. The comparison results are shown in Table 5 to verify the feasibility and superiority of the proposed method. The proposed ShuffleNetv2-YOLOv3 model mAP can reach 99.1%, F1 score is 98.4%, the model size is 8.9MB, and the detection speed can reach 54 frames per second. In the comparison model, The detection speed of YOLOv3-tiny is higher than that of ShuffleNetv2-YOLOv3-1.5X-CIOU 6 frames, but the model is about twice the size of the improved model and has slightly lower mAP and F1 scores. The Mobilenetv2-YOLOv3 model is only 0.2 MB smaller than the improved model, but its detection speed, mAP and F1 score are slightly lower. The YOLOv4-tiny, Fatser-RCNN, and SSD models are inferior to the shuffnetv2 model in all aspects. This validates the superiority of the proposed ShuffleNetv2-YOLOv3-1.5X-CIOU model.

Table 5
Comparison of different network models

Network models	mAP/ %	F1 score/ %	Weight size/MB	FPS
ShuffleNetv2-YOLOv3-1.5X-CIoU	99.1	98.4	8.9	54
YOLOv3-tiny	98.6	98.0	17.4	60
MobileNetv2-YOLOv3	98.8	98.1	8.7	47
SSD	97.4	90.4	98.7	39
Faster-RCNN	98.7	91.3	113.7	18
YOLOv4-tiny	99.0	95.0	23.6	44

6 Conclusion

To achieve sign language recognition higher detection speed and accuracy on removable devices, the novel Shufflenetv2-YOLOv3 lightweight is presented. The traditional MSE bounding box loss is replaced with the CIoU bounding box loss, which further improves the detection accuracy. The test results show that the mAP and F1 scores of the model can reach 99.1% and 98.4% respectively on the homemade datasets, with a model size of only 8.9MB. For 640×480 images, the detection speed on the GPU is 54 frames per second. Compared with SSD, Faster-RCNN, YOLOv4-tiny and other models, it has improved in all aspects, and has a comparative advantage in similar gesture recognition. The proposed ShuffleNetv2-YOLOv3 network structure can be transplanted to embedded platform or mobile terminals for real-time multi-target sign language gesture recognition.

Declarations

Acknowledgments

This study was supported by Research and Development project of Key Core Technology and Common Technology in Shanxi Province (Grant No. 2020XXX001, 2020XXX009) and Major science and technology project of Shanxi Tiandi Coal Machine Equipment Co., LTD M2020-ZD03.

Declaration of conflicting interests

The authors declare that there is no conflict of interest.

References

1. Cortes C, Vapnik V. Support-Vector Networks[J]. Machine Learning, 1995, 20(3):273–297.
2. Davis, J, Shah, et al. Visual gesture recognition[J]. Vision Image & Signal Processing Iee Proceedings, 1994.

3. Zhang Liangguo, Gao Wen, et al. Chinese Sign Language Visual Recognition System for Medium Vocabulary [J]. Computer Research and Development, 2006,43(3):476–482.
4. Munib Q, Habeeb M, Takruri B, et al. American sign language (ASL) recognition based on Hough transform and neural networks[J]. Expert Systems with Applications, 2007, 32(1):24–37.
5. Pigou L, Dieleman S, Kindermans P J, et al. Sign Language Recognition Using Convolutional Neural Networks[J]. Workshop at the European Conference on Computer Vision, 2014.
6. Nakjai P, Katanyukul T. Hand Sign Recognition for Thai Finger Spelling: an Application of Convolution Neural Network[J]. Journal of Signal Processing Systems, 2018.
7. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
8. Girshick R. Fast R-CNN[J]. Computer Science, 2015.
9. Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 39(6):1137–1149.
10. Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[J]. IEEE, 2016.
11. Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger[J]. IEEE Conference on Computer Vision & Pattern Recognition, 2017:6517–6525.
12. Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J]. arXiv e-prints, 2018.
13. Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[C]// European Conference on Computer Vision. Springer, Cham, 2016.
14. Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications[J]. 2017.
15. Zhang X, Zhou X, Lin M, et al. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices[J]. 2017.
16. Xu Liukai et al. "Human gesture recognition algorithm based on convolution neural network based on surface EMG signal energy kernel phase diagram." Journal of Biomedical Engineering 38.4 (2021): 9.
17. Huang, Y., and J. Yang. "A multi-scale descriptor for real time RGB-D hand gesture recognition." Pattern Recognition Letters 144.10(2020).
18. Tan, Y.S., Lim, K.M., Tee, C. et al. Convolutional neural network with spatial pyramid pooling for hand gesture recognition. Neural Comput & Applic 33, 5339–5351 (2021).
19. Fan Jingjing, et al. "Gesture recognition algorithm with ghosting feature mapping and channel attention mechanism." Journal of computer aided design and graphics 34.03 (2022): 403–414
20. Ma N, Zhang X, Zheng H T, et al. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design[C]// European Conference on Computer Vision. Springer, Cham, 2018.
21. Jiang B, Luo R, Mao J, et al. Acquisition of Localization Confidence for Accurate Object Detection[J]. 2018.

22. H Rezatofighi, Tsoi N, JY Gwak, et al. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression[C]// 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2019.
23. Zheng Z, Wang P, Liu W, et al. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression[C]// AAAI Conference on Artificial Intelligence. 2020.
24. G. Marin, F. Dominio, P. Zanuttigh, "Hand gesture recognition with Leap Motion and Kinect devices", IEEE International Conference on Image Processing (ICIP), Paris, France, 2014
25. G. Marin, F. Dominio, P. Zanuttigh, "Hand Gesture Recognition with Jointly Calibrated Leap Motion and Depth Sensor", Multimedia Tools and Applications, 2015
26. A. Memo, L. Minto, P. Zanuttigh, "Exploiting Silhouette Descriptors and Synthetic Data for Hand Gesture Recognition", STAG: Smart Tools & Apps for Graphics, 2015
27. A. Memo, P. Zanuttigh, "Head-mounted gesture controlled interface for human-computer interaction", Multimedia Tools and Applications, 2017

Figures

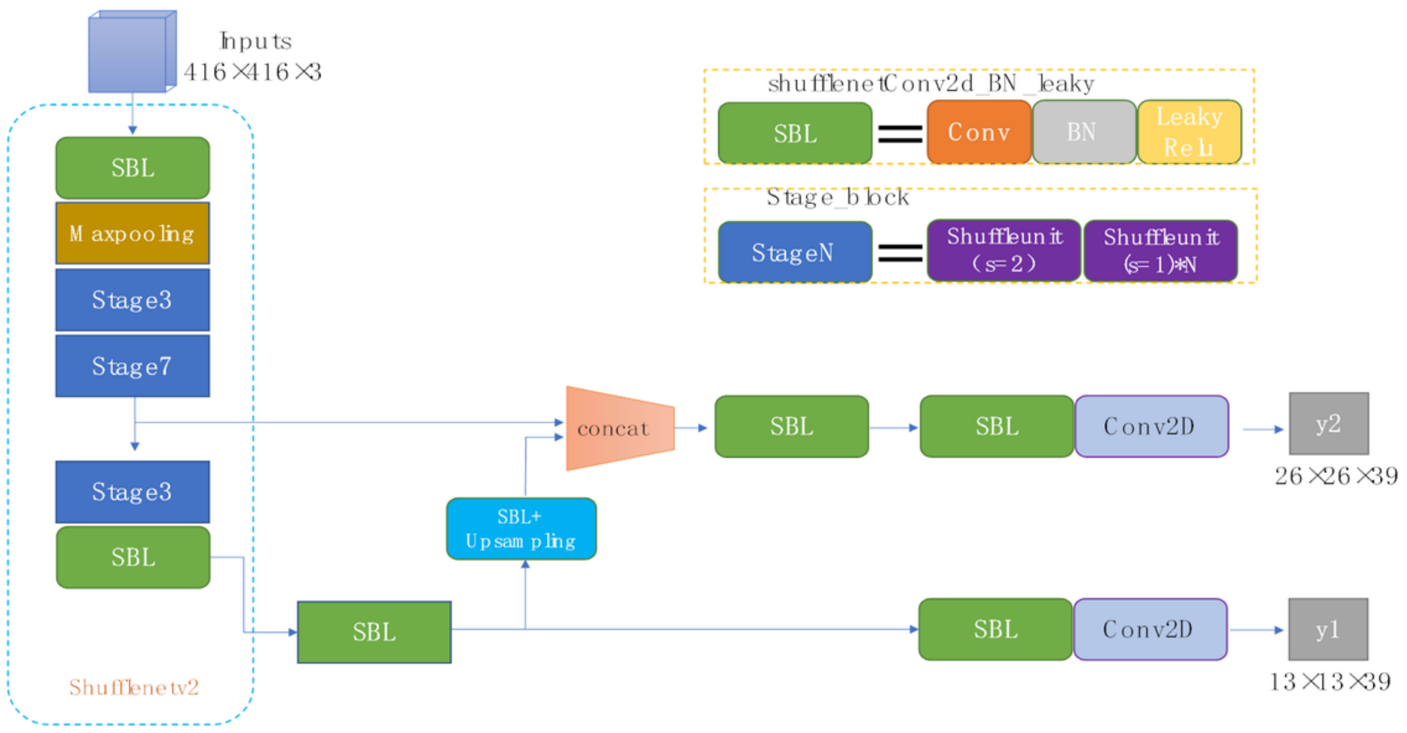

Figure 1

ShuffleNetv2-YOLOv3 structure

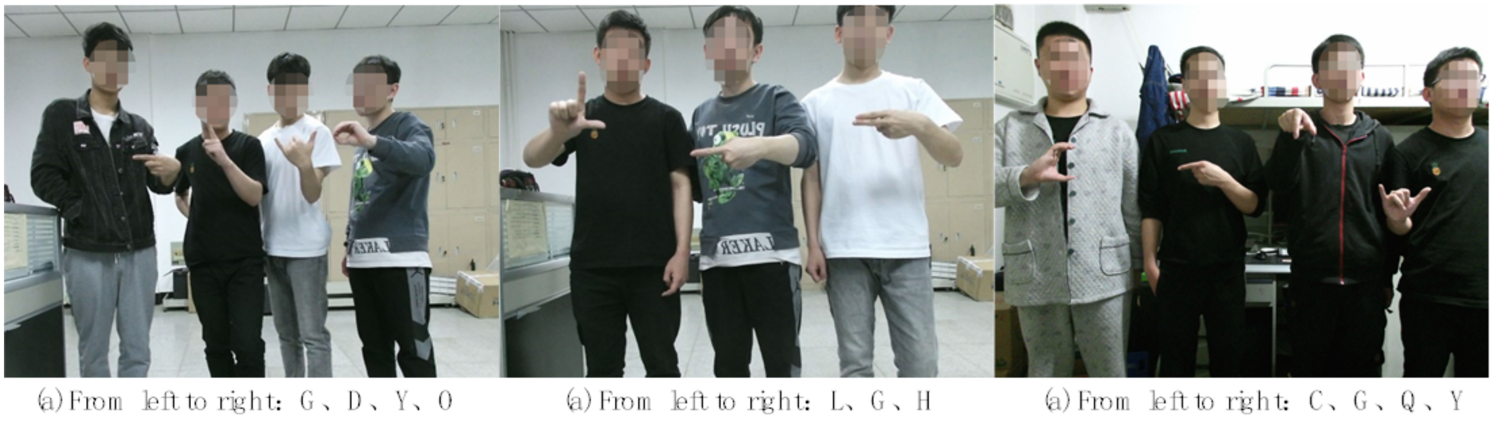
(a) From left to right: G、D、Y、O
(a) From left to right: L、G、H
(a) From left to right: C、G、Q、Y

Figure 2

Figure legend not available with this version.

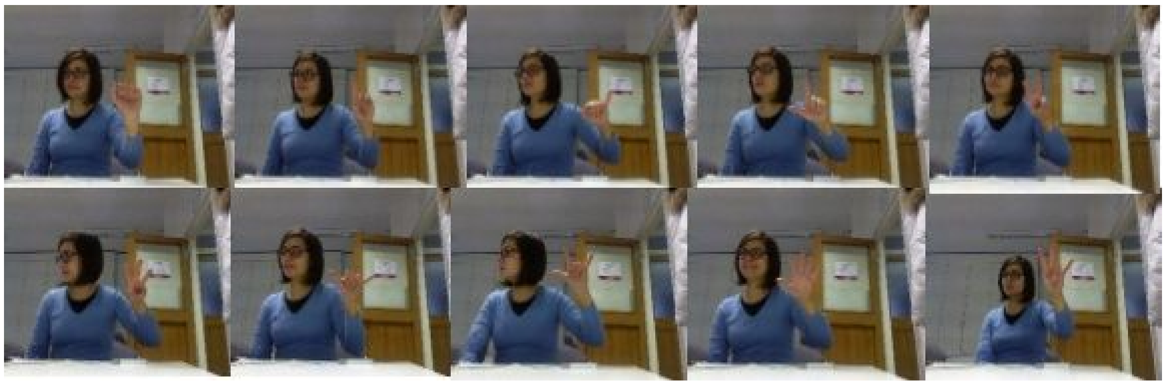

Figure 3

Microsoft Kinect and Leap Motion data

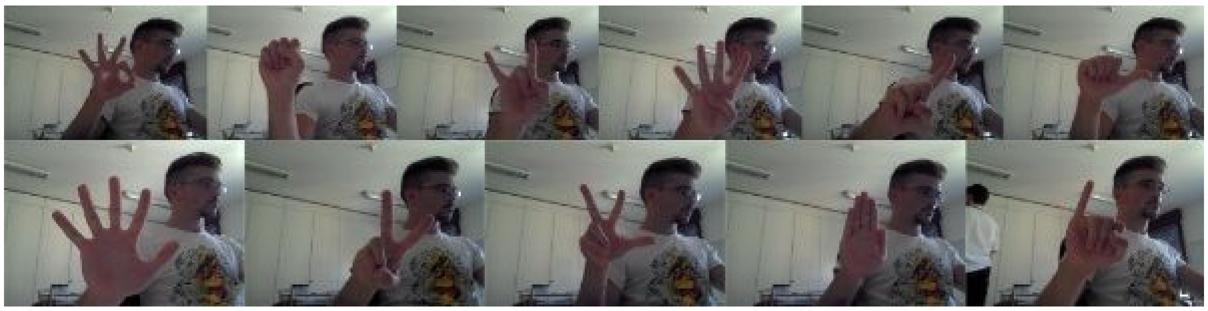

Figure 4

Creative Senz3D data

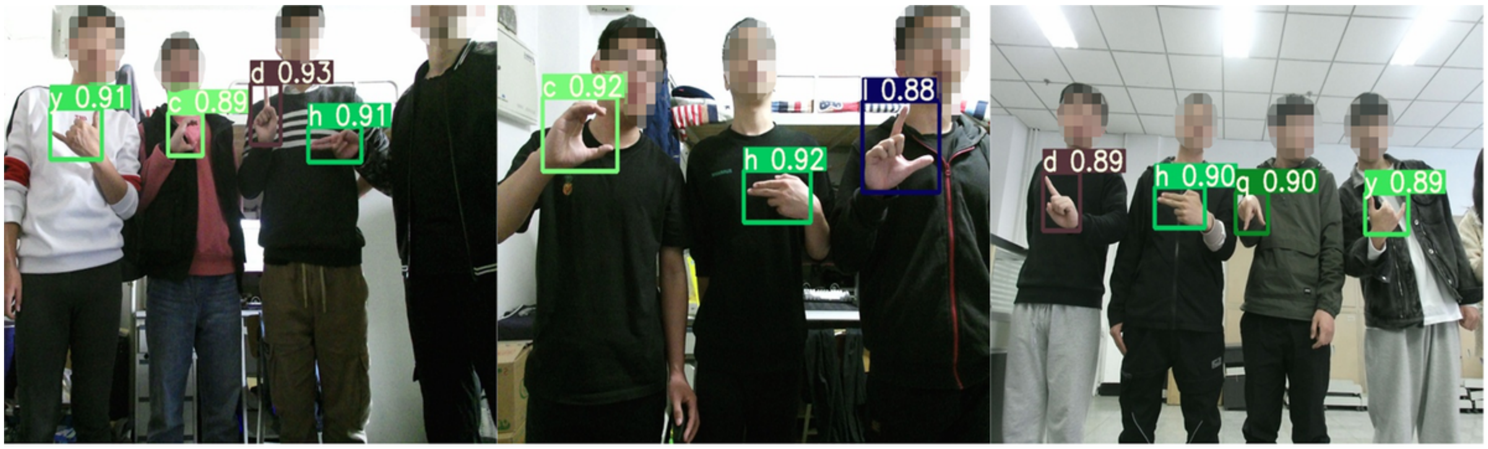

Figure 5

Actual detection effect