2019 IEEE 6th International Conference on Industrial Engineering and Applications

YOLO-Based Valve Type Recognition and Localization

Vahid Behtaji Siahkal Mahalleh, Tareq Aziz ALQutami, Iskandar Al-Thani Mahmood
Facility of Future
PETRONAS RESEARCH SDN BHD
Selangor, Malaysia
e-mail: vahid.behtajisiahka@petronas.com, Tareqazizhasan.al-q@petronas.com, Iskandar.mahmood@petronas.com

***Abstract*—In this paper, we present a YOLO model to recognize valve types and localize their positions on an experimental test bench unit in real-time frames taken from a continuous stream of video data, which are captured by a camera in a hydrogen sulfide (H2S) laboratory. The model required input frames after a particular period. In additional, the model was able to determine the type of valve labels based on a single frame. We showed that the pre-processing of training images plays an important role in the performance of YOLO model. We also included the concept of anchor and intersection over union (IOU) for good accuracy. Valve type results were demonstrated over a specific time of the video stream. The achieved average loss error value of the model was less than 6 percentage and the IOU between the ground-truth bounding box and predicted bounding box for all regions in the final iterations was almost 95 percentage.**

Keywords-valve type recognition; anchor; IOU; average loss error value

I. INTRODUCTION

Hydrogen sulfide (H_2S) poses a high risk for operators assisting robot systems in H_2S laboratories[1]. In this situation, laboratory automation plays a significant role to minimize the need to assist robots and improve the processes of controlling valves in order to be more precise. One of the solutions of automation is computer vision ,which replaces manual inspection [2]. Computer vision is an approach to enable a robot to “look” and thus, allows the robot to represent human eyes for object detection [3]. Object detection is a branch of computer vision and plays an important role in a wide range of robotic applications such as autonomous robot manipulation (ARM) and collision avoidance, which requires the need to recognize the presence of both stationary and moving objects in a specific area of the robots to perform corresponding actions such as interaction and braking[4]. Accordingly, the general purpose of object detection should be both fast and accurate.

In this present research, the ARM setup as illustrated in Fig. 1 was used to automate H_2S experiments. The robot manipulator requires a high degree of autonomy that is able to detect and localize a wide range of objects in order to perform certain number of manipulation tasks such as opening and closing switch valves as well as adjusting control valves fast and accurately.

From the perspective of speed and accuracy, object detection based on deep learning is much better compared to using traditional machine learning algorithm [5]. Deep learning is a forward-feedback neural network and has special privilege in image detection with its unique structure of local weights sharing[6]. Deep learning based object detection methods include conventional layers, pooling, drop-out, and fully-connected layers. Classification and localization are two main aspects of object detection. Traditional detectors require the use of feature extraction methods. These methods often increase the computational cost of the model[7, 8]. One of the region-based object detection algorithms is Fast-RCCN in which classification is achieved by CNN [9]. In F-RCCN, the process of extracting features is selective and it is time consuming [2]. On the other hand, The You Only Look Once (YOLO) approach is extremely fast [10]. The implementation of location and classification in this method can be performed in a single CNN. As such, this research proposed the use of the YOLO method.

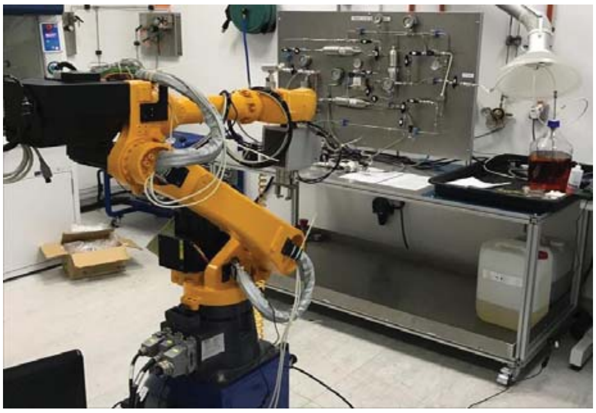


Figure 1. H_2S panel test and ARM.

To develop the automation of our robotic manipulation in an H_2S laboratory for the detection of valves, the YOLO technique, as a real-time object detection algorithm was proposed. Image dataset was collected by taking videos of valves in a H_2S panel to build our own training and testing set. The camera used was 12-megapixel model and the recording was set 1080 pixels at 25 frames per second (fps).

The remaining sections of this paper are organized as follows. Section II presents the pre-processing of training images and the design of the whole system. The implementation and results are illustrated in section III. Lastly, the conclusion and potential opportunities for future research are presented in section IV.

II. System Design

The accuracy of YOLO is similar to that of Fast-RCNN [10]. In addition, YOLO is the fastest object detection, in contrast to other current algorithms such as Fast-RCNN and R-CNN. The good performance of YOLO is dependent on training images. Given that, training images reflect the characteristic of YOLO [10].

The YOLO object detection algorithm is based on CNN that performs object recognition in a box called an anchor. An anchor is centred on the 'S × S' grid cell in an image. Each grid cell can predict 'B' bounding boxes, confidence scores for those boxes and 'C' conditional class probabilities. In Fig. 2, the ground-truth bounding box was drawn in black, while the predicted bounding box was drawn in red. The IOU between these bounding boxes should be computed. The computation of the IOU can be determined via (1).

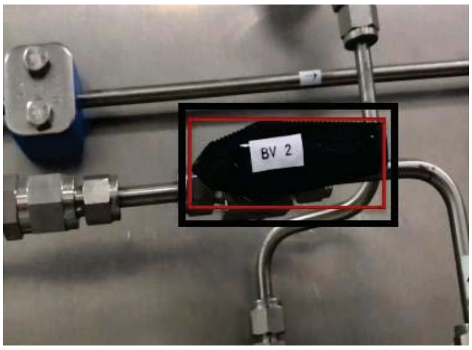


Figure 2. Red is predicted bounding box and black is ground-truth bounding box.

In this study, there were two types of objects; switch and control valves as shown in Fig. 2. Therefore, the class probability of output layer is 2 categories. An anchor was centred on the S= 52, B=3 and C=2 and the final output was a 52 × 52 × (B × (5+C)) = 52 × 52 × 21 three-dimensional array. TABLE I demonstrate the final output of our network structure.

According to the first stage of convolution (0 conv) input column in TABLE I, the size was reduced to 416 × 416 regardless of the original image size which was to be trained. Major distortion occurred on the objects in the resizing process, when there was a large difference in the ratio of width to height of the training images. However, if there was a same ratio between the recognition and learning images, a good performance could be expected. YOLOv3.cfg file from original YOLO was used for network architecture. The number of filters in the cfg file was derived from (2).

$$\text{IOU} = \frac{\text{Area of overlap}}{\text{Area of union}} \tag{1}$$

TABLE I. Network Structure

Layer Name	Filter	Size/ stride	Input	Output
0 conv	32	3 x 3 / 1	416 x 416 x 3	416 x 416 x 32
1 conv	64	3 x 3 / 2	416 x 416 x 32	208 x 208 x 64
2 conv	32	1 x 1 / 1	208 x 208 x 64	208 x 208 x 32
3 conv	64	3 x 3 / 1	208 x 208 x 32	208 x 208 x 64
4 res	1		208 x 208 x 64	208 x 208 x 64
5 conv	128	3 x 3 / 2	208 x 208 x 64	104 x 104 x 128
6 conv	64	1 x 1 / 1	104 x 104 x 128	104 x 104 x 64
7 conv	128	3 x 3 / 1	104 x 104 x 64	104 x 104 x 128
8 res	5		104 x 104 x 128	104 x 104 x 128
9 conv	64	1 x 1 / 1	104 x 104 x 128	104 x 104 x 64
10 conv	128	3 x 3 / 1	104 x 104 x 64	104 x 104 x 128
11 res	8		104 x 104 x 128	104 x 104 x 128
⋮	⋮	⋮	⋮	⋮
105 conv	21	1 x 1 / 1	52 x 52 x 256	52 x 52 x 21

The characteristic of the anchor plays a significant role in achieving good accuracy of YOLO. Anchors are used in the training process and defined as a fixed ratio. Computing good candidate anchor boxes is the first step in YOLO. It is achieved via k-means by looking at the shape of the objects in the training images. The direct Euler distance matric in k-means contributes to minimum error for larger bounding boxes. Therefore, IOU was applied as a Euler distance metric. All bounding boxes were assumed to locate at one point in the calculation of IOU, and only the height and width were utilized as features.

$$\text{Filters} = (\text{classes} + 5) \times 3 \tag{2}$$

During pre-processing, the ratio of area, which is occupied by each object in the image in both training and testing, must be the same for good accuracy. Furthermore, the sizes of all training and testing images must be similar. According to the principles of pre-processing, training dataset should be annotated properly. The YOLO-Annotation-Tool was used for this purpose. The YOLO-Annotation-Tool performed the following operations:

- Randomly selected the images from the image dataset. Each image had one or more than one object and all the objects were annotated with the location, size and class.
- The size of all objects (switch and control valves) was reduced to an appropriate size. The proper size was achieved by considering the size of which the target appeared to be recognized.
- The annotation creator annotated the size and location of the newly located objects in the base image. Fig.3 demonstrates the three above steps in the process of generating training images in the YOLO system.

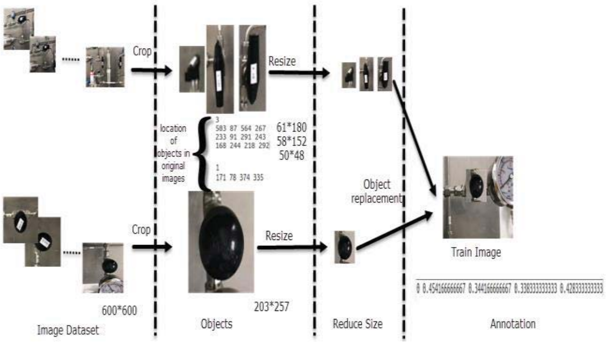


Figure 3. Image Pre-processing technique.

As shown in the first step of Fig. 3, objects should be cropped from the original image dataset. In the second step, the size of objects should be reduced to match both training and detection. In the next step, a trian image is selected and transfers the object at random location. In this step each object should not be overlapped. Eventually, the class, size and position of the relocated objects are recalculated to create annotations.

The pre-processing should be repeated for the image dataset to obtain a form suitable for YOLO training. This process can also be easily applied for a different dataset.

III. Implementation and Results

DARKNET framework was installed and used in Ubuntu operating system for this research. This research was done in GPU-enabled platform with YOLOv3. The dataset included 2 classes (switch and control valves). The dataset contains 350 images, 85% for training and 15% for testing. YOLOv3.cfg file was applied and the following changes were made in the cfg file. The batch was set at 16 which means 16 images are used for every training step. In addition, subdivisions parameter in the cfg file was set to 16, which means that the batch is divided by 16 to decrease the GPU VRAM requirements. According to (2), the Filter parameters were set to 21 in this case.

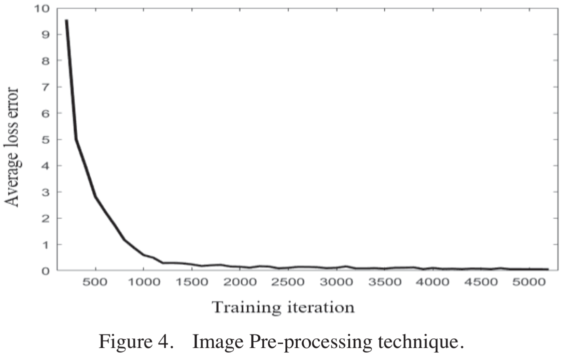
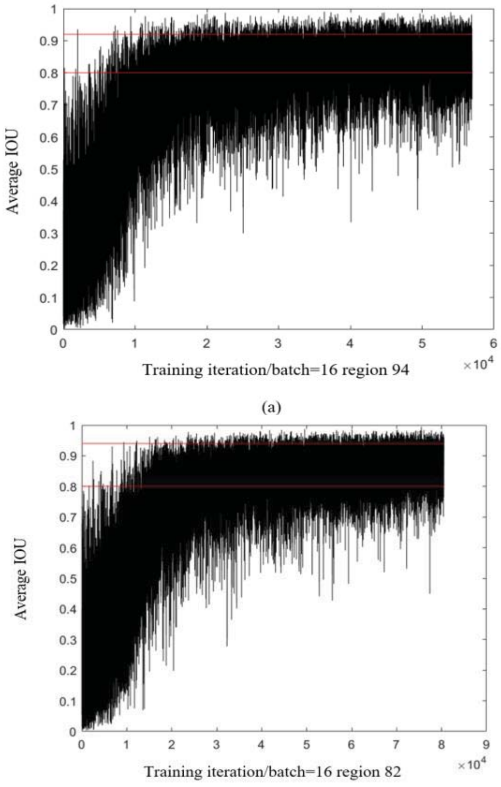


Figure 4. Image Pre-processing technique.

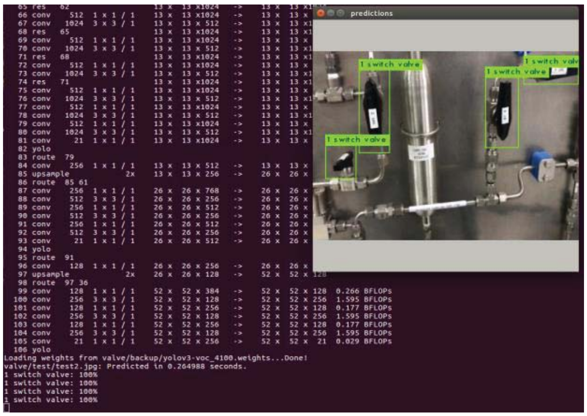
Fig. 4 shows the loss rate function over the number of iterations. After 4000 iterations, the average loss error decayed to less than 0.066173.

Fig. 5(a) shows the average IOU for region 94 in 5200 iterations of training in which the batch was equal to 16 and the total number of IOU was equal to 56960. The number of IOU was less than 16*5200=83200 since in some of the initial iterations, IOU was NaN. These NaNs decreased in the final iterations. General y,NaN values happen when the anchors are big or the object dimensions are too small. Fig. 5(b) illustrates the average of IOU for region 82 for 5200 iterations of training in which the batch was equal to 16 and the total number of IOU was equal to 80512. Based on Fig. 5, it was clear that as the number of iterations increased, the average IOU between the ground-truth bounding box and the predicted bounding box would get closer to one.



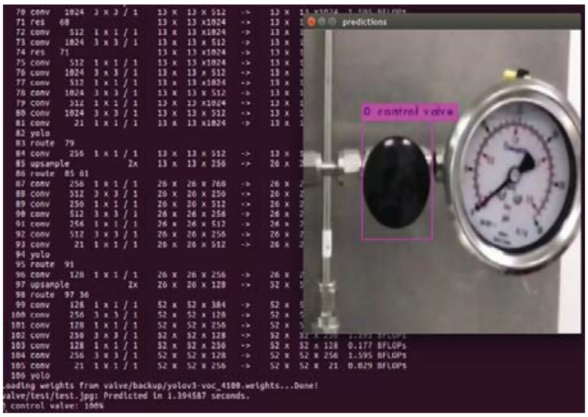
(b)

Figure 5. (a) average of IOU for region 94. (b) average of IOU for region 82.



```
 65 res   62                13 x  13 x1024   ->    13 x  13 x1
 66 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 67 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 68 res   65                13 x  13 x1024   ->    13 x  13 x1
 69 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 70 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 71 res   68                13 x  13 x1024   ->    13 x  13 x1
 72 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 73 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 74 res   71                13 x  13 x1024   ->    13 x  13 x1
 75 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 76 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 77 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 78 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 79 conv    512  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 80 conv   1024  3 x 3 / 1   13 x  13 x 512   ->    13 x  13 x1
 81 conv     21  1 x 1 / 1   13 x  13 x1024   ->    13 x  13 x
 82 yolo
 83 route  79
 84 conv    256  1 x 1 / 1   13 x  13 x 512   ->    13 x  13 x
 85 upsample            2x   13 x  13 x 256   ->    26 x  26 x
 86 route  85 61
 87 conv    256  1 x 1 / 1   26 x  26 x 768   ->    26 x  26 x
 88 conv    512  3 x 3 / 1   26 x  26 x 256   ->    26 x  26 x
 89 conv    256  1 x 1 / 1   26 x  26 x 512   ->    26 x  26 x
 90 conv    512  3 x 3 / 1   26 x  26 x 256   ->    26 x  26 x
 91 conv    256  1 x 1 / 1   26 x  26 x 512   ->    26 x  26 x
 92 conv    512  3 x 3 / 1   26 x  26 x 256   ->    26 x  26 x
 93 conv     21  1 x 1 / 1   26 x  26 x 512   ->    26 x  26 x
 94 yolo
 95 route  91
 96 conv    128  1 x 1 / 1   26 x  26 x 256   ->    26 x  26 x
 97 upsample            2x   26 x  26 x 128   ->    52 x  52 x 128
 98 route  97 36
 99 conv    128  1 x 1 / 1   52 x  52 x 384   ->    52 x  52 x 128  0.266 BFLOPs
100 conv    256  3 x 3 / 1   52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
101 conv    128  1 x 1 / 1   52 x  52 x 256   ->    52 x  52 x 128  0.177 BFLOPs
102 conv    256  3 x 3 / 1   52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
103 conv    128  1 x 1 / 1   52 x  52 x 256   ->    52 x  52 x 128  0.177 BFLOPs
104 conv    256  3 x 3 / 1   52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
105 conv     21  1 x 1 / 1   52 x  52 x 256   ->    52 x  52 x  21  0.029 BFLOPs
106 yolo
Loading weights from valve/backup/yolov3-voc_4100.weights...Done!
valve/test/test2.jpg: Predicted in 0.264988 seconds.
1 switch valve: 100%
1 switch valve: 100%
1 switch valve: 100%
1 switch valve: 100%
```

(a)



```
 70 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024  1.595 BFLOPs
 71 res    68                 13 x  13 x1024   ->    13 x
 72 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x
 73 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x
 74 res    71                 13 x  13 x1024   ->    13 x
 75 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x
 76 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x
 77 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x
 78 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x
 79 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x
 80 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x
 81 conv     21  1 x 1 / 1    13 x  13 x1024   ->    13 x
 82 yolo
 83 route  79
 84 conv    256  1 x 1 / 1    13 x  13 x 512   ->    13 x
 85 upsample           2x     13 x  13 x 256   ->    26 x
 86 route  85 61
 87 conv    256  1 x 1 / 1    26 x  26 x 768   ->    26 x
 88 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x
 89 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x
 90 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x
 91 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x
 92 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x
 93 conv     21  1 x 1 / 1    26 x  26 x 512   ->    26 x
 94 yolo
 95 route  91
 96 conv    128  1 x 1 / 1    26 x  26 x 256   ->    26 x
 97 upsample           2x     26 x  26 x 128   ->    52 x
 98 route  97 36
 99 conv    128  1 x 1 / 1    52 x  52 x 384   ->    52 x
100 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x
101 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x
102 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
103 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 128  0.177 BFLOPs
104 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
105 conv     21  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x  21  0.029 BFLOPs
106 yolo
Loading weights from valve/backup/yolov3-voc_4100.weights...Done!
valve/test/test.jpg: Predicted in 1.394587 seconds.
0 control valve: 100%
```

(b)

Figure 6. (a) switch valve detection . (b) control valve recognization.

As shown in Fig. 6, the results of object detection were achieved by the trained network in the proposed method. The results reveal that the high performance of object recognition depends on the quality of images that included objects and their notations. If the YOLO is trained from images in which the objects in pre-processing step were not annotated properly (such as overlap or size of object gets greater than one or less than zero), the YOLO is unable to detect anything.

IV. Conclusion

The process of YOLO-based real-time valve detection was described in this paper. The training images were collected by a camera in an H_2S laboratory. During pre-processing, it is important to ensure that the same size of area is occupied by each object throughout training and testing. As shown in Figures 5(a) and (b), the total numbers of IOU for the region 82 were more than that of region 94. Therefore, the size of anchor for region 82 was more properly selected, compared to region 94. In addition, the average loss error of the model converged to less than 6% after 4000 iterations and the IOU between the ground-truth bounding box and the predicted bounding box for all regions in the final iterations was almost 95%. The results of object detection were satisfactory.

However, in this research only two classes were used. Hence, further studies are required for a higher number of classes as well as to recognize the valve status.

References

[1] EE. Fabian-Wheeler, ML. Hile, and D. Murphy, “Operator exposure to hydrogen sulfide from dairy manure storages containing gypsum bedding,” Journal of agricultural safety and health, vol. 23, Jan. 2017, pp. 9–22, doi: 10.13031/jash.11563.

[2] J. Tao, H. Wang, X. Zhang, X. Li, and H. Yang, “An object detection system based on YOLO in traffic scene,” 6th International Conference on Computer Science and Network Technology (ICCSNT), Oct. 2017, pp. 315-319,
doi: 10.1109/ICCSNT.2017.8343709.

[3] S. Nishiguchi, K. Ogawa, Y. Yoshikawa, T. Chikaraishi, O. Hirata and H. Ishiguro, “Theatrical approach: Designing human-like behaviour in humanoid robots,” Journal of agricultural safety and health, vol. 89, Mar. 2017, pp. 158–166,
doi: https://doi.org/10.1016/j.robot.2016.11.017.

[4] A. Hourtash, M. Hingwe, P. Schena, S. Bruce Micheal and R.L. Devengenzo, “Manipulator arm-to-patient collision avoidance using a null-space,” Intuitive Surgical Operations Inc, vol. 23, Nov. 2016, pp. 3–16, doi: https://patents.google.com/patent/US9492235B2/en.

[5] R. Shaoqing, He. Kaiming, G. Ross and S. Jian, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” In Advances in neural information processing systems, Jan. 2016, pp. 91-99, doi: arXiv:1506.01497v3.

[6] L. Zheng, F. Canmiao, and Z. Yong, “Extend the shallow part of Single Shot MultiBox Detector via Convolutional Neural Network,” Jan. 2018, doi: arXiv:1801.05918.

[7] Y.H. Chen, T. Krishna, J.S. Emer, and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” IEEE Journal of Solid-State Circuits, vol.52, Jan. 2017, pp. 127-138, doi: 10.1109/JSSC.2016.2616357.

[8] A. Hajian and Y. Suet-Peng, “Feature Extraction from EEG Data for a P300 Based Brain-Computer Interface,” In Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Cham, Oct, 2017, pp. 39-50, doi: https://doi.org/10.1007/978-3-319-67274-8_4.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” InProceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 2016, pp. 779-788.

[10] H.J. Jeong, K.S. Park, and Y.G. Ha, “Image Preprocessing for Efficient Training of YOLO Deep Learning Networks,” In Big Data and Smart Computing (BigComp), 2018 IEEE International Conference on, Jan.2018, pp. 635-637, doi: 10.1109/BigComp.2018.00113.