IMPROVING THE ACCURACY OF SMALL OBJECT DETECTION ON YOLO BY INCREASING THE NUMBER OF INPUT GRIDS

Herdianto[1]*, Indri Sulistianingsih[2], Iwan Fitrianto Rahmad[3]
[1]Universitas Pembangunan Panca Budi, Jl. Gatot Subroto KM 4,5, Kota Medan, 20122, Indonesia
[2]Politeknik Negeri Medan, Jl. Almamater No.1, Padang Bulan, Kota Medan, 20155, Indonesia
[3]Universitas Potensi Utama, Jl. K.L. Yos Sudarso KM 6,5, Kota Medan, 20241, Indonesia

Keywords:
Object Detection, YOLO, Driver Autonomous Systems, Deep Learning.

***Correspondence Address:**
herdianto0108047703@gmail.com

Abstract: Object detection is one of every Driver Autonomous System (DAS) capabilities. However, the object detection results currently used are limited to detecting large objects, whereas for small objects less than 80 * 80 pixels, the detection accuracy can be less than 60% when using YOLO. Based on the low object detection accuracy results above, this research will try to increase the number of grids in the YOLO input image from 7*7, 10*10, 13*13, 16*16 and 19*19 in the YOLO input to improve object detection accuracy small in size. The image data obtained is divided into two parts: 70% for training data and 30% for testing. From the results of tests carried out on objects measuring 80 * 80 pixels with a grid of 7 * 7, it is known that the accuracy of the detection results reaches 90%. Meanwhile, the number of grids 10 * 10, 13 * 13, 16 * 16 and 19 * 19 is still under further testing.

INTRODUCTION

Object detection is a task in computer vision widely applied in the field, including in watersheds. This object detection must be done on the DAS to avoid collisions with surrounding objects. However, as is known, there are many objects in the environment around the DAS; complex and object detection that the DAS has must have a speed that matches the speed of the DAS and be able to detect small objects so that there is no delay in making decisions to avoid collisions.

To overcome the above problems, several studies related to object detection have been carried out, including those carried out by (Viola & Jones, 2001), (Viola & Jones, 2004). However, in this research, the object detected was specifically for human face detection, and the results obtained stated that the algorithm was very well designed, could be run on a low-specification computer (Pentium 3), and had high accuracy.

Furthermore, there is also other research conducted by (Navneet & Triggs, 2005) which tries to use the Histograms of Oriented Gradient (HOG) method to detect an object. Even though the HOG method can determine the vector and edge direction of the shape of an object, HOG cannot yet be used to decide whether the object is detected by a human or something else. For this reason, the HOG method for object detection is always equipped with the Support Vector Machine (SVM) method. Then, the research results (Navneet & Triggs, 2005) repaired by (Redmon et al., 2008), (F. Felzenszwalb et al., 2009), (Felzenszwalb et al., 2013) his research stated that the objects contained in the image could be recognized with training. Research related to object detection is still an interesting topic, so other researchers continue to try to improve the shortcomings of existing research results, as can be seen from the subsequent research carried out (Alex et al., 2012), (Herdianto & Nasution, 2023). The results of this research state that it can detect and classify objects in large numbers but still has the disadvantage of being slow > 0.010FPS in detecting objects. Then there were improvements made by (Girshick et al., 2014), (Girshick et al., 2016), (Girshick, 2015), (Ren et al., 2017), (He et al., 2017), (Redmon et al., 2016), (Redmon & Farhadi, 2018). Each study proposed improved methods known as Fast CNN, Faster CNN, Mask CNN, and You Only Look Once (YOLO). With the Fast CNN method, the detection speed increased to 0.5 FPS, then improved again by Faster CNN to 7 FPS and YOLO to 45 FPS.

Even though YOLO has the detection speed as expected, it still has shortcomings in detecting small objects. Therefore, in this research, YOLOV4 will be tried to detect small objects.

RESEARCH METHODS

YOLOV4 is a type of deep learning that can detect objects. The learning stages in YOLOV4 to detect objects are illustrated in Figure 1, where the input image is divided into a grid with a matrix size of 7*7. In this research, YOLOV4 will be tried with different numbers of grids in the input image: 7*7, 10*10, 13*13, 16*16, and 19*19. For each number of grids tested, the accuracy of the detection object will be evaluated.

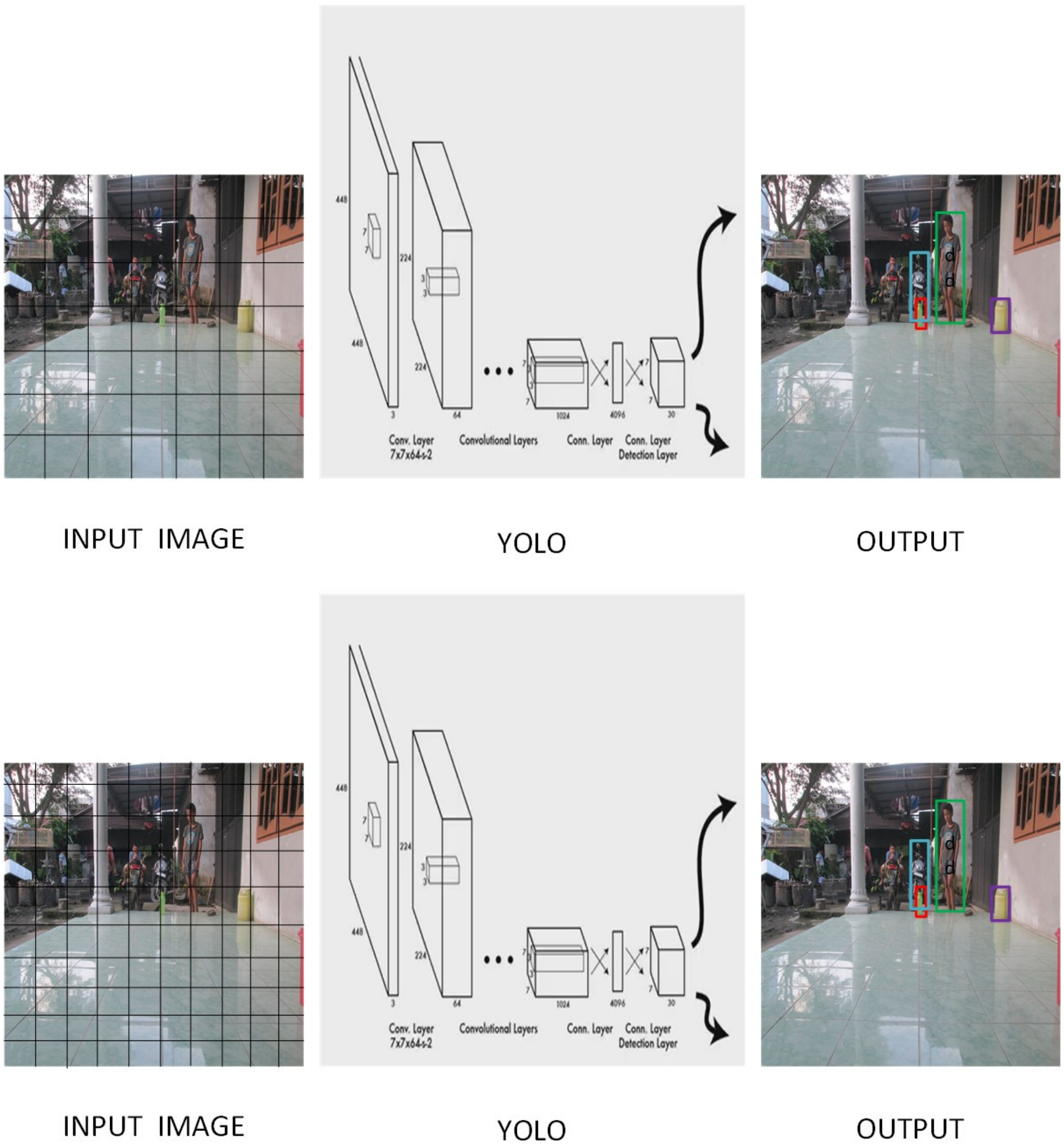
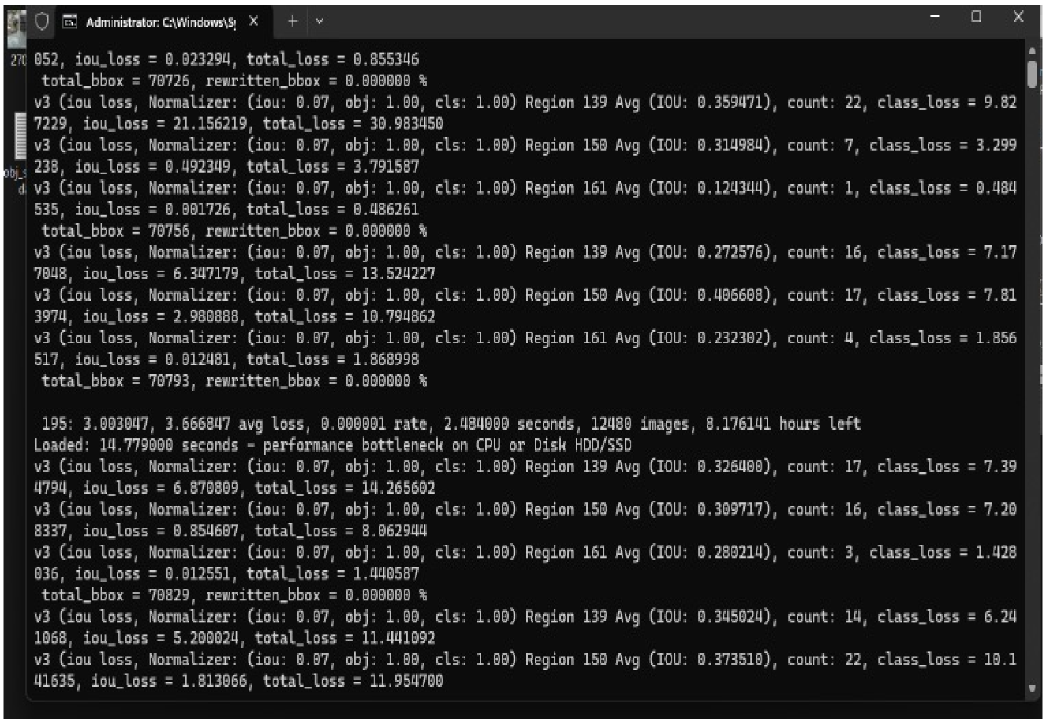


Figure 1. Object of research carried out

RESULTS AND DISCUSSION

The data used for training and testing was obtained from Kaggle with a total of 3 object classes (birds, fish, bees) and each class contained 2, 128, 49 objects. Then the data for the fish and bee classes was obtained in two parts. part namely 80% training and 20% testing. Next, so that the YOLO network used can correctly determine the class names and dividing boxes formed, the YOLO network is trained.

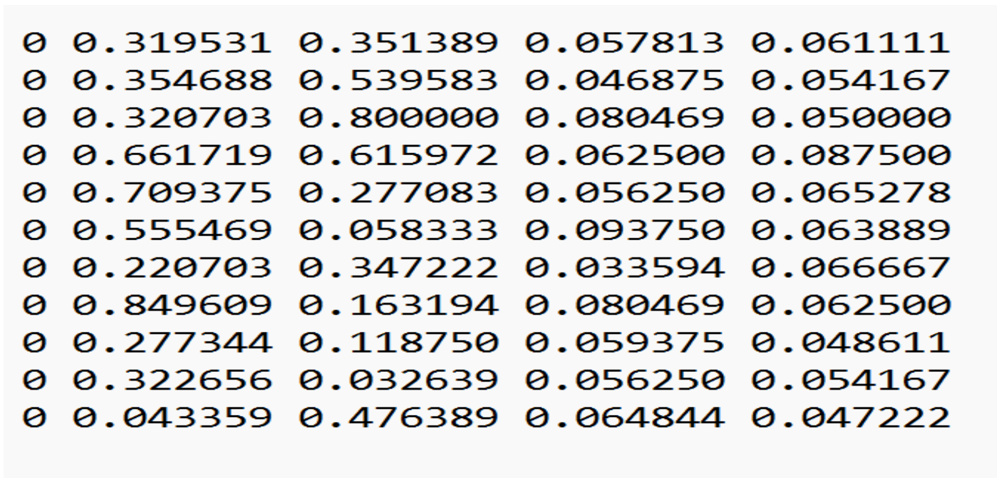


```
052, iou_loss = 0.023294, total_loss = 0.855346
 total_bbox = 70726, rewritten_bbox = 0.000000 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.359471), count: 22, class_loss = 9.82
7229, iou_loss = 21.156219, total_loss = 30.983450
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.314984), count: 7, class_loss = 3.299
238, iou_loss = 0.492349, total_loss = 3.791587
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.124344), count: 1, class_loss = 0.484
535, iou_loss = 0.001726, total_loss = 0.486261
 total_bbox = 70756, rewritten_bbox = 0.000000 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.272576), count: 16, class_loss = 7.17
7048, iou_loss = 6.347179, total_loss = 13.524227
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.406608), count: 17, class_loss = 7.81
3974, iou_loss = 2.980888, total_loss = 10.794862
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.232302), count: 4, class_loss = 1.856
517, iou_loss = 0.012481, total_loss = 1.868998
 total_bbox = 70793, rewritten_bbox = 0.000000 %

 195: 3.003047, 3.666847 avg loss, 0.000001 rate, 2.484000 seconds, 12480 images, 8.176141 hours left
Loaded: 14.779000 seconds - performance bottleneck on CPU or Disk HDD/SSD
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.326400), count: 17, class_loss = 7.39
4794, iou_loss = 6.870809, total_loss = 14.265602
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.309717), count: 16, class_loss = 7.20
8337, iou_loss = 0.854607, total_loss = 8.062944
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.280214), count: 3, class_loss = 1.428
036, iou_loss = 0.012551, total_loss = 1.440587
 total_bbox = 70829, rewritten_bbox = 0.000000 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.345024), count: 14, class_loss = 6.24
1068, iou_loss = 5.200024, total_loss = 11.441092
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.373510), count: 22, class_loss = 10.1
41635, iou_loss = 1.813066, total_loss = 11.954700
```

Figure 2. YOLO network training process

The working process of training the YOLO network can be explained as follows: before the image data is trained, the class is first labeled, the center point of the bounding box coordinates (x, y) along with the width and height which will later be changed to a value between 0 - 1 See Figure 3, these values are the output in YOLO.



```
0 0.319531 0.351389 0.057813 0.061111
0 0.354688 0.539583 0.046875 0.054167
0 0.320703 0.800000 0.080469 0.050000
0 0.661719 0.615972 0.062500 0.087500
0 0.709375 0.277083 0.056250 0.065278
0 0.555469 0.058333 0.093750 0.063889
0 0.220703 0.347222 0.033594 0.066667
0 0.849609 0.163194 0.080469 0.062500
0 0.277344 0.118750 0.059375 0.048611
0 0.322656 0.032639 0.056250 0.054167
0 0.043359 0.476389 0.064844 0.047222
```

Figure 3. Conversion results into 0 – 1 for class, x,y, width and height

0 represents the class name, 0.319531 0.351389 represents the center point of the x,y bounding box, and 0.057813 0.061111 is the width and height of the bounding box. Furthermore, every object with an input image that is larger than 448 * 448 will be

reduced to 416 * 416 * 3 because of Red Green Blue (RGB). Then, the convolution process is carried out for several layers with different window sizes for each layer used, and the output of the YOLO layer will be smaller than the layer above it. The purpose of this convolution is to obtain the characteristics of each object being trained.

Table 1. Image pixel values

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

The pixel values in Table 1 are convoluted with a 2 * 2 filter so that they become like Table 3.

Table 2. Filter values

1	0
0	1

Table 3. New pixel values resulting from convolution

7	5	9
4	7	9
32	2	5

At each YOLO convolution layer, the Rectified Linear Unit (RELU) activation function is used so that if there is a negative pixel value, the output will be 0, and if the pixel value is positive, the output will be the same as Table 3. Next, the pooling process is carried out from the RELU output if the image pixel value from RELU has a value like Table 4 then what is taken is the largest value from each filter so that the new pixel value is like Table 5.

Table 4. Pooling pixel values

1	4	3	2
2	5	10	17
10	32	24	20
15	20	11	6

Table 5. Pooling results

5	17
32	24

After the RELU process is complete, flattening is carried out which changes the image from 2 dimensions to 1 dimension. The illustration is shown in Figure 4.

Table 6. Flatten process

5	17
32	24

2 dimensions

$\longrightarrow$

5
17
32
24

1 dimensions

After the flatten process is complete, followed by fully connected, here the YOLO output is placed.

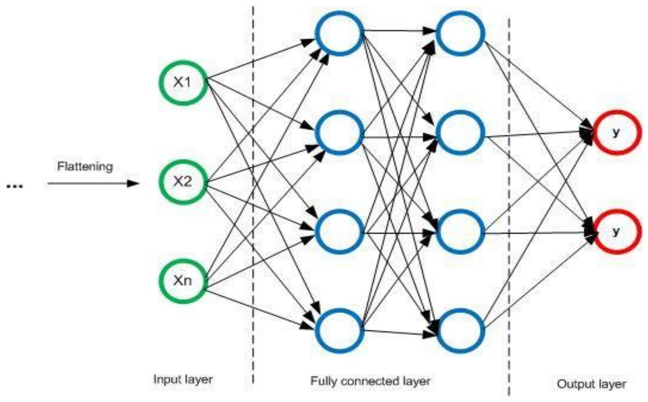
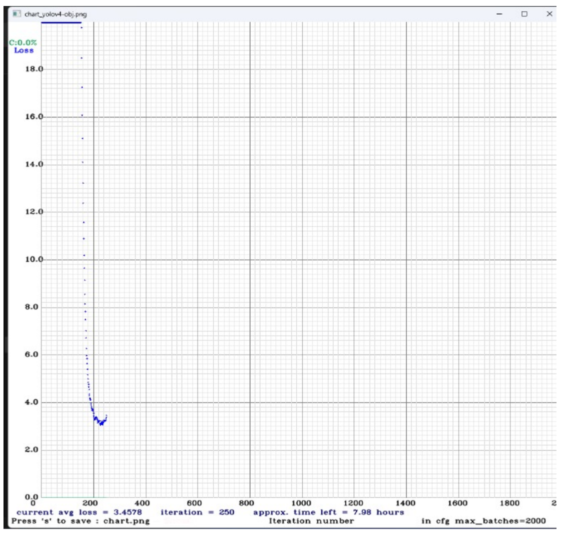


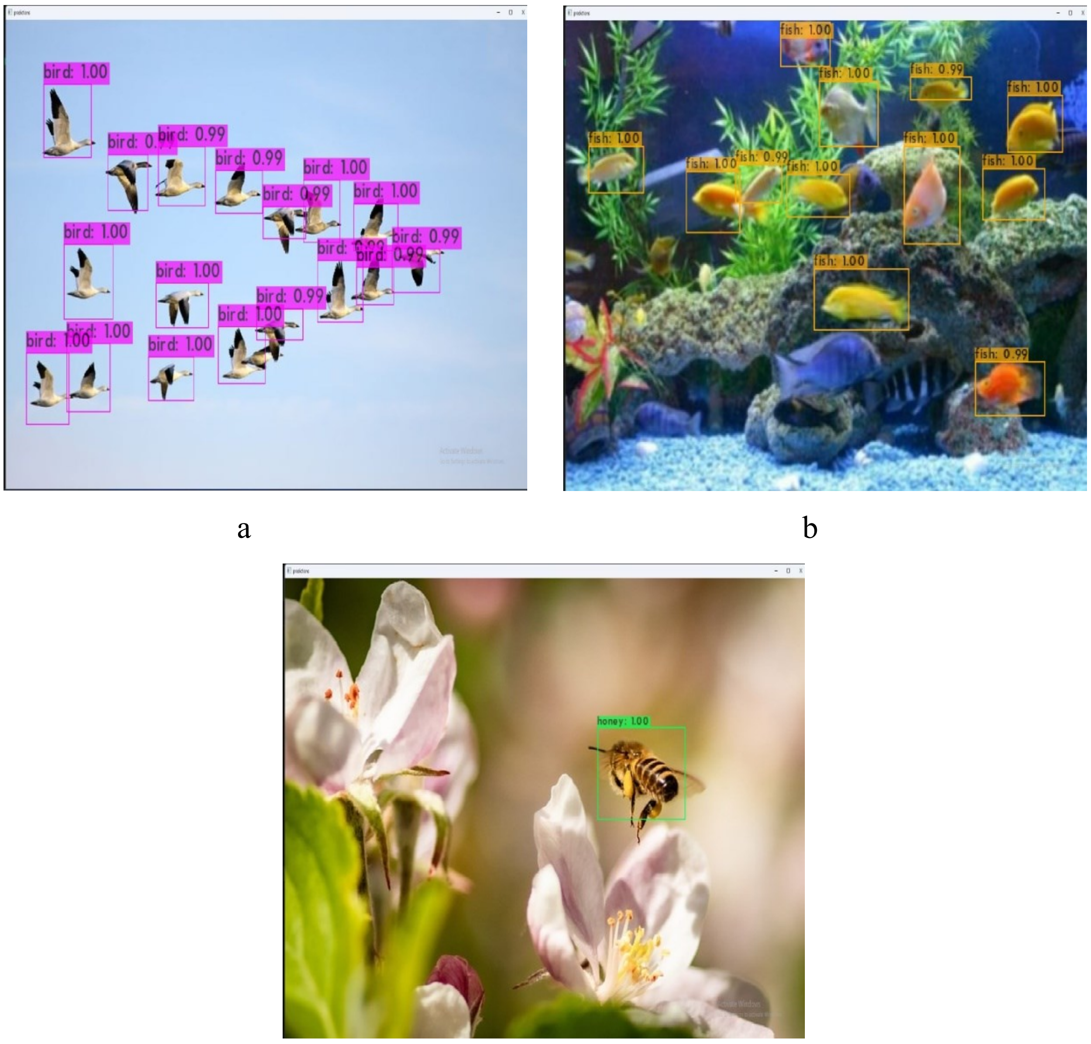
Figure 4. Fully connected



```
current avg loss = 3.4578    iteration = 250    approx. time left = 7.98 hours
Press 's' to save : chart.png                                          in cfg max_batches=2000
```

Figure 5. Error correction in YOLO training

Figure 5. Shows the error correction process during training where the training will stop when the predetermined error tolerance value has been reached.



c

Figure 6. Object detection results from YOLO predictions

Table 7. Accuracy of YOLO object detection prediction results

No	Object name	Accuracy of prediction results (%)
1	Bird	90
2	Fish	92
3	bee	92

CONCLUSION

The test results show that the method applied with a number of grips of 7*7 on objects measuring 80*80 produces object detection accuracy of up to 90% on three objects, namely birds, fish and bees. Meanwhile, testing for grids 10*10, 13*13, 16*16 and 19*19 is still in the testing phase.

REFERENCE

Alex, K., Ilya, S., & E Geoffrey, H. (2012). Imagenet classification with deep convolutional neural networks. *NIPS Conference*, 1097–1105.

F. Felzenszwalb, P., B. Girshick, R., McAllester, D., & Ramanan, D. (2009). Object Detection with Discriminatively Trained Part Based Models. *Computer*, *47*(2), 1–19.

Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2013). Visual object detection with deformable part models. *Communications of the ACM*, *56*(9), 97–105. https://doi.org/10.1145/2500468.2494532

Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 1440–1448. https://doi.org/10.1109/ICCV.2015.169

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 580–587. https://doi.org/10.1109/CVPR.2014.81

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 142–158. https://doi.org/10.1109/TPAMI.2015.2437384

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 2980–2988. https://doi.org/10.1109/ICCV.2017.322

Herdianto, H., & Nasution, D. (2023). Implementasi Metode Cnn Untuk Klasifikasi Objek. *METHOMIKA Jurnal Manajemen Informatika Dan Komputerisasi Akuntansi*, *7*(1), 54–60. https://doi.org/10.46880/jmika.vol7no1.pp54-60

Navneet, D., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 1–8. https://doi.org/10.1007/978-3-642-33530-3_8

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once:

Unified, real-time object detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Felzenszwalb, P., McAllester, D., Ramanan, D., An, W., … Zhang, L. (2008). A Discriminatively Trained, Multiscale, Deformable Part Model. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *330*(6), 1299–1305.

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. http://arxiv.org/abs/1804.02767

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*, 1–9.

Viola, P., & Jones, M. (2004). Robust Real-Time Face Detection Intro to Face Detection. *International Journal of Computer Vision*, *57*(2), 137–154.