Computer Vision with Deep Convolutional Neural Network Approach for Cold-Flow Casting Defect Detection

[1,2]Adhy Syaefudin
[1] PT. Astra Honda Motor,
Bekasi - Indonesia
[2] Master of Mechanical Engineering
Swiss German University
Tangerang City, Indonesia
adhy.syaefudin@gmail.com

[1,2] Fuad Widiatmoko
[1] Infinitigroup, Bekasi - Indonesia
[2] Master of Mechanical Engineering
Swiss German University
Tangerang City, Indonesia
fuadwidiatmoko7@gmail.com

Edi Sofyan
Master of Mechanical Engineering
Swiss German University
Tangerang City, Indonesia
edi.sofyan@lecture.sgu.ac.id

Widi Setiawan
Master of Mechanical Engineering
Swiss German University
Tangerang City, Indonesia
widi.setiawan@lecture.sgu.ac.id

Rifo Nurlaksana Restu
PT. Astra Honda Motor,
Bekasi - Indonesia
rifonurlaksanarestu@gmail.com

Abstract*—Quality of the engine parts is very important, because will be directly affected to motorcycle performance. If the defects from the Aluminum High-Pressure Die Casting (HPDC) injection process that are not detected and sent to the next process will cause production loss and will harm the customer. The research presents the implementation of object detection technology based on Deep Convolutional Neural Network (DCNN) to detect cold-flow defect. Computer vision with DCNN algorithm will be used to improve the result of visuals human inspections who have been detecting cold-flow defects that will in a result more stable and objective in carrying out quality assessments. This research will analyze the performance of the DCNN framework called YOLOv5s in Python. The analysis includes lighting conditions and the characteristics of the dataset. This research used cloud computing at Google-Colab during the training process of the DCNN, the computer specifications with graphic processing unit known as GPU (Tesla T4, 1.5 GB, 40 processor assigned by Google-Colab) are needed for faster training process. The Roboflow was very helpful tools in the dataset preparation phase of the development of this system. In conclusion, the developed system has proven to be very successful in assisting the HPDC part visual inspection, with mAP value 0.33, box loss value 0.06 and the average detection speed per object defect is 0.3 seconds.

Keywords*—cold-flow, DCNN, YOLO, roboflow, python, object detection.

I. INTRODUCTION

The most complicated engine part in the scooter-matic motorcycle type in the High Pressure Die Casting (HPDC) manufacturing process is Crankcase Left which is the larger part of engines. The HPDC process occurs at high temperatures (+ 650odegrees-celcius) and high speeds injection. There are many problems that occur in this process, including surface quality, dimensional, and inner quality.

HPDC (High Pressure Die Casting) is very vulnerable to produce defects, it should be considered that in HPDC filling stage extreme conditions, complexity of components leads to complex dies, and the high production rates required (up to 120 shots/h) lead to very high filling velocities for the molten alloy (up to 40 m/s) with strong generation of turbulence in the flow. Solidification takes place in few seconds, and the die is first in contact with a molten alloy at more than 700°C and, after 30-40 seconds, with a sprayed lubricant at room temperature, this condition potentially will generate defect product 5-10% in average. With a high production volume the defect of casting part means a very large loss in production costs [1].

To ensure the quality of crankcase left, we must check some quality parameter, such as dimensional quality, inner quality and surface quality. For dimensional quality check use CMM (Coordinate Measuring Machine) touching probe as touch probe base measurement machine or using 3D-scanner as optical processing base. To detect inner quality such as porosity can be done by X-Ray machine with special specification to detect Aluminum material.

And the third item is the surface quality, a defect on the surface of Aluminum casting currently checking for visual defect relies on human visual capabilities in mass production lines, this has the potential for error in making quality judgments, even though quality inspector have been provided with limit sample of goods that have been agreed visually, because there is a factor of fatigue and subjectivity. In this paper will focus how to detect surface defect of Aluminum casting part, specifically cold-flow defect (Fig. 1). Cold-flow also known as cold-shut, flow-lines, chill, laps and several other names of surface defects used in the HPDC industry, defects that occur on the surface of casting parts, and are visible to the eye as the leading edge of metal flow is too cold, laps together [2]. Cold flow cause not good

microstructure (Fig. 2) fusion this condition also has the potential to cause engine leaks after the assembling process [1].

Fig. 1. Typical of cold-flow surface defect [2]

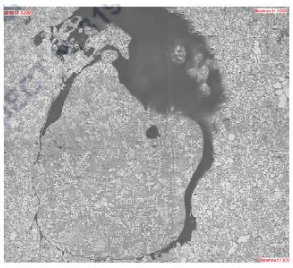


Fig. 2. Micro graph of cold-flow [1]

II. Literature Review

A. Casting Defect Detection

Aluminum die casting does not compromise on quality, unfortunately, most inspection systems in casting today are manually operated. The inspection quality depends on operator experience, and according to the variety of defects and their different features, the inspection process itself is time consuming and inefficient. In the development of casting defect detection technology, X-ray used for identification and evaluation of internal defects which is carried out to avoid human inconsistency in making quality decisions. And now x-ray inspection can be automated to detect internal defects such as porosity and material shrinkage. The original output or x-ray result sometime has a problem in reading the x-ray image because the defect line is very thin, pre-processing is needed [3].

Currently, computer vision to detect defects in casting products has begun to develop using consumer cameras, one of which is the implementation to detect surface defects in the automobile engine cylinder bore area. The CCD camera is used to capture images of the 3 segments of the cylinder bore after finishing the machining process. By comparing the annular shape of these three images through the threshold process and normalizing noise and lens distortion, it will be input into a Visual C ++ based computer system to be processed and measured [4].

B. Convolutional Neural Network

Convolutional neural network provides a solution by simplifying the input of pixels into the array, then through the feature detector array it will be simplified into feature mapping, then pooling and flattening are carried out and this is the input to the fully connected neural network [5].

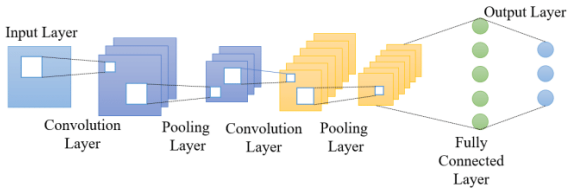


Fig. 3. Convolutional neural network basic architecture [6]

Fig. 3 illustrates how this CNN works, from the input image which can be in the form of photos or videos, the features will be extracted in the Convolutional layer, in this layer the image will be converted into numeric data with grayscale or in RGB (Red-Green-Blue) scale, then go to next layers and as the output from CNN is a classification of the output image of several categories that have been made [6]. The advantage of CNN is the simplification process in the convolutional layer, so that any size input image will easily shrink the image dimensions while still being able to recognize the features of the image, this can speed up the computation process and reduce the requirement of computer specifications [7].

CNN and later evolved into the DCNN algorithm which breaks the calculation into smaller segments. This DCNN system consists of two basic sections, namely Backbone and Head. Backbone extract feature from the input, to find the pattern of the object in train process. Head section to predicting the object, calculate the size of bounding box and locate by coordinate the bounding box to the object detected (R-CNN, FASTER R-CNN, YOLO), there is two types of head, one-step and two-step, where speed and accuracy as the trade of this types [8].

C. YOLO (You Only Look Once)

Compared to previous object detection algorithms, such as the R-CNN image processing is repeated to ensure and eliminate double detection, the main difficulty is in pipeline preparation. For example with using alexnet pre-trained data, need to resize the training input image to equal size of alexnet database [9]. YOLO (You Only Look Once) is an object detection algorithm that is very simple in preparation and fast in image processing, in general the concept of YOLO reduces the image size to 448x448 pixels (1st version is 224x224 pixels) as shown in Fig. 4, and only performs one convolutional network in the image processing to produce object detection in bounding box. With this speed YOLO is very suitable for objects with video input [10]. The second generation of YOLO efforts to improve mAP, which is the object detection framework's precision parameter by adding batch normalization to YOLOv2 and also being able to train with a larger image input resolution, from 224x224 pixels on early YOLO to 448x448 pixels on YOLOv2 and this is also claimed increase the accuracy improved by GoogleNet Architecture and Darknet-19 [11].

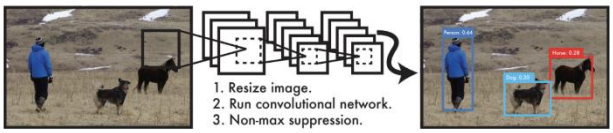


Fig. 4. The YOLO concept is resizing image and run only single convolutional network for faster calculation [10]

Latest version is YOLOv5 (Fig. 5), one of the improvement items is to eliminate weaknesses in the detection of small objects. Which is YOLOv4 uses a Darknet backbone in which the deep learning library used is the new Tensor-Flow and recently developed using PyTorch [12].

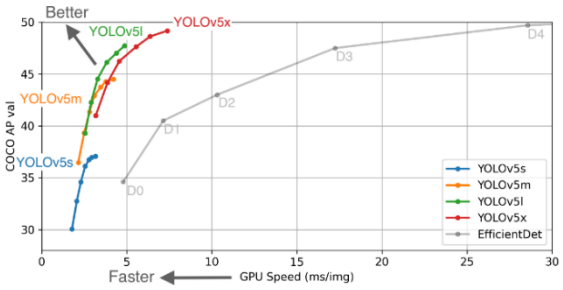


Fig. 5. YOLO compared with EfficientDet [13]

In this paper, base on the requirements in the HPDC Machine Injection Production line conveyor system, will focus to YOLOv5s version.

III. RESEARCH METHOD

In general, the process begins with sampling cold-flow defect data on the crankcase part, the process of retrieving data using a mobile phone, a natural light source from sunlight. The cold-flow defect image data is then entered in the annotation process using the "labelimg.py" tool and the results are entered into ROBOFLOW for data model preparation. Followed by the training process carried out on google-colab based on the python program language. And implement real-time object detection using a video webcam.

A. Collecting Image Data for Training Process

In the data collection process, focus on one areas of the defect cold-flow part crankcase left type scooter-matic (area inside red box). With a target of 300 cold-flow defect sample data from 300 crankcase sample parts will be input during the training process into DCNN-based YOLOv5s.

As previously stated, the process of shooting with natural light sources from sunlight is taken in the 9 to 11 am range, with light intensity in the 275-300 lux range. With the development trend of object detection as an application for computer vision, now it is easier and cheaper to use devices, one of which is to use the capabilities of a mobile phone camera as shown in Fig. 6 and Table 1 [14].

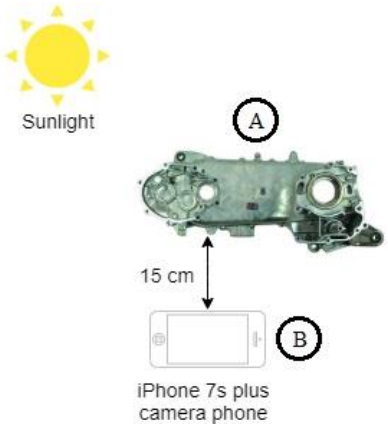


Fig. 6. Layout of image collecting

TABLE I
LIST OF COMPONENT AND SPECIFICATION

No	Component Name	Specification
1	Computer for run Object Detection with Python Program	Dell Inspiron 3541 : Intel Core Processor i7-9750H, Nvidia Quadro P620 graphics with 4GB GDDR5
2	Cloud computation for training process	Google-Colab: GPU processing by google assign
3	Mobile phone camera	iPhone 7s Plus: 12 MP (4032 x 3024 pixels
4	Webcam camera	Logitec C922 : 30 FPS at 1080P
5	Lighting	CFL Lamp : 2 X 9 watt, diffused LED lamp : 11 watt, diffused

B. Annotating Process

To be able to teach computer vision in machine learning, part conditions are defined as cold-flow defects are carried out in the annotation process, this process that will determine how CNN will analyse in the object detection process. The annotation process is carried out using open-source tools developed by Tzutalin (Fig. 7), this tool also supports object detection using YOLO which will be used in this research. LabelImg is based on Python and the results of the annotation process will be saved in TXT as the format used by YOLO [15].

As previously stated, in the train process it is only focused on one area, in the train process the area is defined with small rectangular boxes to provide a more detailed pattern per area of the NG part example.

Fig. 8 is an illustration of how annotation defines the cold-flow defect area, each rectangular box in the annotation process will provide a cold-flow defect pattern to the DCNN system, and this will be the data in the machine learning process.

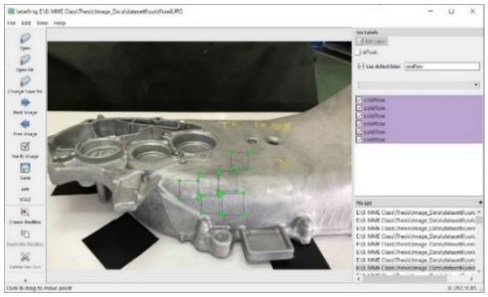
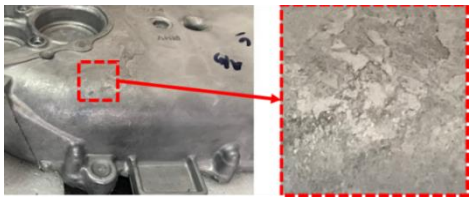


Fig. 7. Annotation process using labelImg by tzutalin



(a) (b)

Fig. 8. (a) Crankcase left image captured (b) cold-flow surface defect detail pattern result of cropping process and illustration of annotation process

C. Preparing Machine Learning Data Model

In this research, the prepare models process is carried out by cloud computing, using Roboflow which makes this process very simple. Unlike conventional processes, it uses complex code programs to convert from image data and annotations to models that are ready for use in machine learning in the training process as shown in Fig. 9. Roboflow already supports various machine learning platforms, and in this research using PyTorch-based YOLOv5 (Fig. 10).

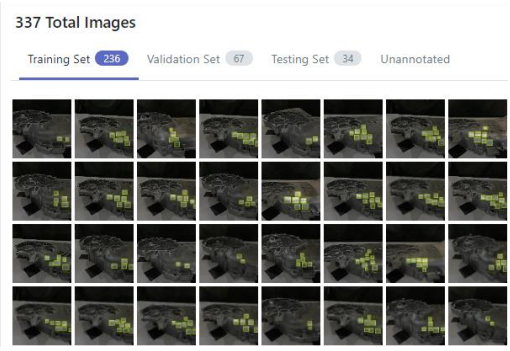


Fig. 9. Upload image data and annotation data to Roboflow

When uploading data to Roboflow, it will be divided into 3 types of data sets, namely: training set (70%), validation set (20%) and testing set (10%) for needs during the machine learning training process.

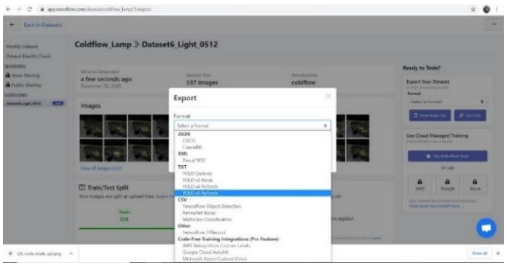


Fig. 10. Export format models as YOLOv5 PyTorch

After uploading the data, it is followed by generating data models to be included in the code program in the training process, this is the convenience provided by Roboflow compared to conventional processes which are entirely done using program code.

D. Training Process

The toughest step to building a machine learning is the train process, requiring high specification resources and a long duration of training process with high specification of physical computer [14]. One of the solutions taken in this research is to conduct a train process with cloud computing using Google-Colab resources, which provides a computing process using the GPU of course with limited processor time for free services.

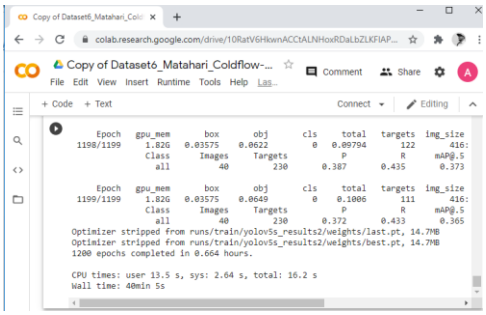


Fig. 11. Training process 1200 epoch using GPU processor by Google-Colab

In this paper only one class, namely "coldflow", for that it is necessary to do a small modification on the default YOLOv5 models file, where the data models and configuration are in the yaml file. The training process is carried out with 1200 epochs as shown in Fig. 11.

E. Live Trial using Video Webcam

In accordance with the conditions in the train process, including the distance between objects to the camera and light intensity, the live trial stage is to validate how the object detection system can detect cold-flow defects in parts that have not used in the train process. In the live trial using a video webcam with the specifications in the Table 2 and Fig. 12. Layout of live trial using video webcam and for the equalization of the resolution conditions is done by parsing the code "—img-size".

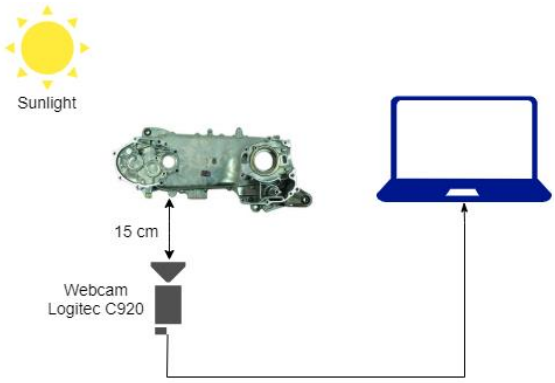


Fig. 12. Layout of live trial using video webcam

TABLE II
TRAINING PARAMETERS

Concheck	Specification
Input image to Roboflow	337
GPU spec for training with Google Collabs	Tesla T4 1.5 GB, 41 Core
Depth multiple	0.33
Channel multiple	0.5
Number of layers	283
Number of parameters	7.255.094
Average training iteration speed (Iteration/sec)	11.5
Total epoch	1200
Train processing time	40 min, 30 sec

IV. RESULT AND DISCUSSION

The result of the training process is the weight file that will be used in the object detection process using Python Script that runs YOLO. In addition to the weight file, performance analysis is also produced from the training results to be able to assess whether the training results have reached the optimum point so that they can be stable in the object detection process.

As a popular method to analize performance of the results of the development of the object detector is expressed in mAP (mean Average Precision). MAP is simply obtained from the area of the graph precision and recall [16]. In addition, to be more in-depth in analyzing, you can also analize box loss and object loss.

A. Training Process Result

From the trial results using the colf-flow defect dataset, with a training process of 1200 epochs using the YOLOv5s performance shown in Fig. 13 systems reached a convergent state at the 200th epoch, and a final value at 0.33 according to the literature in Fig. 5. Even though the value of an epoch doesn't mean anything, focus more on the trend.

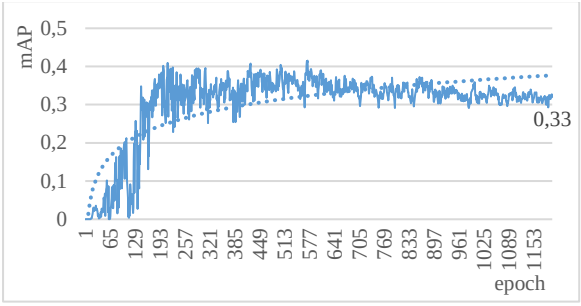


Fig. 13. mAP@0.5 result of training process

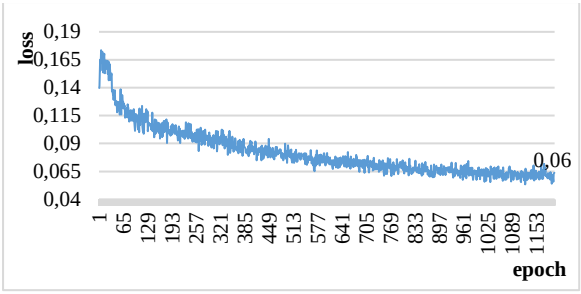


Fig. 14. Object loss result of training process

As shown in Fig. 14 the object loss value continued to decline, until at the end of the epoch it was at 0.06, from the results of the analysis of the possibility of the object loss occurring in the shadow area on the part due to uneven sunlight on the crankcase part, as shown in Fig. 15 shadowed area in red boxes.

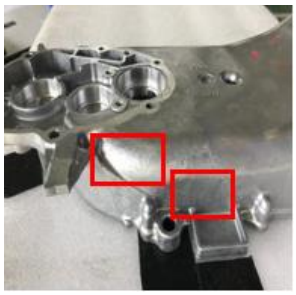


Fig. 15. Shadow area in natural lighting source condition

B. Live Trial using Video Webcam

As previously stated, object detection of cold-flow defects will be implemented live in the video webcam. In this paper, we tried it with 2 resolutions, n amely 512 pixels (Fig. 16) and 768 pixels (Fig. 17), we can change the resolution with parsing code in python program as *–img-size*.

By comparing both resolution, 768 pixels result at Fig. 17 produces more boxes but it looks more complicated, this is because the details of the cold-flow defect flow are held by the camera, in addition it takes a slower time to detect cold-flow defects compared to a resolution of 512 pixels.

With an average detection rate per object of 0.3 seconds at a resolution of 512 pixels and 0.7 seconds at 768 pixels, speed and precision are trade-offs.

Fig. 16. Object detection result sunlight dataset with 512 pixels

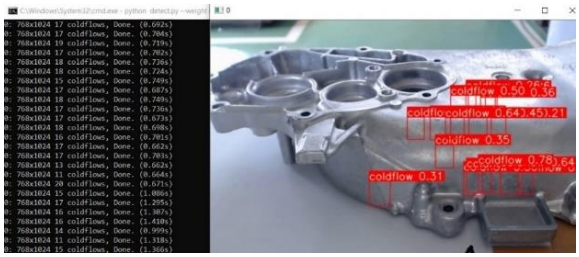


Fig. 17. Object detection result sunlight dataset with 768 pixels

C. Discussion

The process of collection defect images as the input data it training process is very important step in develop object detection system, preparing good lighting, and the more defect sample data will produce a more stable and precise system. As in this paper, there are still shaded areas due to uneven sunlight, for the next research it is recommended to use additional lamps or other lighting system to fill the light in the shaded areas. In addition, YOLO has the ability to detect more than one variety of visual defects of Aluminum casting, such as porosity, dent or scratches, by adding annotation processes as more than one class.

V. CONCLUSION

YOLOv5s provides satisfactory results in detecting cold-flow defects, with a mAP value of 0.33 at the 1200th epoch and a box loss value of 0.06 as reference data to show the performance of the detection object system created. Speed is also an advantage of YOLOv5s, at a video resolution of 512 pixels the average detection speed per object defect is 0.3 seconds and 0.7 seconds for a resolution of 768 pixels. Cloud computing provides convenience and efficiency in the process of object detection training based on DCNN, by using Roboflow and Google-Colab, the preparation and training process takes place easily and quickly. With the provision of computers for GPU-based computing (Graphic Processing Unit) Tesla T4, 1.5 GB, 41 Core, it only took 40 minutes 30 seconds to complete training on 337 cold-flow defect sample images, through 283 deep convolutional neural network layers, and 7.2 million of parameters.

REFERENCES

[1] F. Bonollo, G. Timelli, E. Fiorese, E. Gariboldi, P. Parona, and L. Arnberg, "Database on defects," *D ELIVERABLE*, 2 (1), pp. 1–25, 2013.

[2] G. William and (NADCA) Walkington, *Die Casting Defects: Troubleshooting Guide*, 2003.

[3] X. Li, S. K. Tso, X. P. Guan, and Q. Huang, "Improving automatic detection of defects in castings by applying wavelet technique," *IEEE Trans. Ind. Electron.*, 53 (6), pp. 1927–1933, 2006, doi: 10.1109/TIE.2006.885448.

[4] J. J. Ji and C. Ye, "The automatic detection technology for the surface defects of automobile engine cylinder," *Key Eng. Mater.*, 693, 2016, doi: 10.4028/www.scientific.net/KEM.693.1458.

[5] I. Hadji and R. P. Wildes, "*What Do We Understand About Convolutional Networks?*," Mar. 2018, [Online]. Available: http://arxiv.org/abs/1803.08834.

[6] H. A. O. Gu, Y. U. Wang, S. Hong, G. Gui, and S. Member, "Blind channel identification aided generalized automatic modulation recognition based on deep learning," XX, pp. 1–7, 2019, doi: 10.1109/ACCESS.2019.2934354.

[7] D. Juan, "Understanding of object detection based on CNN family and YOLO understanding of object detection based on CNN family and," *J. Phys. Conf. Ser.*, 2018, doi: 10.1088/1742-6596/1004/1/012029.

[8] M. Rezaei and M. Azarmi, "Deepsocial: social distancing monitoring and infection risk assessment in covid-19 pandemic," *Appl. Sci.*, 10 (21), pp. 1–29, 2020, doi: 10.3390/app10217514.

[9] T. R. Uetama, W. Setiawan, and E. Sofyan, "Performance comparation of real time image processing face recognition for security system," 2nd Proceedings of The Conference on Management and Engineering in Industry (CMEI 2020), vol. 2(1), pp. 21–25, Tangerang, Indonesia, September 2020, doi: 10.33555/cmei.v2i1.38.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "YOLOv1," *Cvpr*, vol. 2016-Decem, pp. 779–788, 2016.

[11] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, 2017-Janua, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "*YOLOv4: Optimal Speed and Accuracy of Object Detection*," 2020, [Online]. Available: http://arxiv.org/abs/2004.10934.

[13] Ultralytics, "*YOLOv5 in PyTorch*," pp. 1–9, 2020, [Online]. Available: https://github.com/ultralytics/yolov5.

[14] B. Wibowo, E. Sofyan, G. Baskoro, "Prototype design of speed detection mobile application for golfers swing movement using computer vision compared to portable radar and accelerometer systems," 1st Proceedings of The Conference on Management and Engineering in Industry (CMEI 2019), vol. 1, pp. 49-52, Tangerang, Indonesia, August 2019

[15] tzutalin, "*LabelImg a Graphical Image Annotation Tool and Label Object Bounding Boxes in Images*." 2018, [Online]. Available: https://github.com/tzutalin/labelImg.

[16] J. Hui, "*mAP (Mean Average Precision) for Object Detection*," pp. 1–10, 2018, [Online]. Available: https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173.