CNN-based Crowd Detection for COVID-19 Social Distancing Protocol from UAV Onboard Camera

[1]Leonard Matheus Wastupranata & [2]Rinaldi Munir

[1,2]School of Electrical Engineering and Informatics,

Institut Teknologi Bandung, Indonesia

[*1] leo.matt.547@gmail.com, [2]rinaldi@informatika.org

*Corresponding author

ABSTRACT

Social distancing is a feasible solution to break the chain of the spread of COVID-19. However, human crowds are the main problem for close contact between humans who are close to each other. A crowd detection model is needed that can estimate the distance between two or more to prevent social distancing violations with a safe limit of less than 1.5 meters. The CNN model training was conducted using 9,600 images of humans, cyclists, and motorcyclists. The pre-trained model used for the experiment of transfer learning method is of Single Shot Detector (SSD) type with MobileNet, ResNet50, and ResNet101 architectures. In addition, the measurement of the estimated social distance uses the Euclidian distance with the average Indonesian human as a reference, which is 1.6 meters. Social distance calibration is also conducted using the principle of projection from different angles of view of the UAV camera while flying. Based on the analysis of test results, MobileNet V2 was chosen as a crowd detection model with a lightweight size, which is only 19 Megabytes and the average detection runtime for a single image is only 0.606 seconds, in accordance with the load for the UAV companion computer. MobileNet V2 is also able to detect crowds of people well, as evidenced by the precision value reaching 84.9% (IoU=0.50:0.95) and the sensitivity (recall) value reaching 87.8% (MaxDets=100). In addition, a program has been successfully developed to count violations using social distance estimation.

Keywords: calibration, COVID-19, human detection, social distance estimation, UAV.

INTRODUCTION

The COVID-19 pandemic has swept across the world and changed the way people live in general. The spread of this virus is extremely fast and massive, especially in environments that are crowded with people. The World Health Organization (WHO) has suggested that steps to anticipate the spread of the virus, including close contact and crowds, should be minimized by implementing social distancing (Qian & Jiang, 2020). Several studies have proven that implementing social distancing has lowered the risk of spreading the virus and reduced mortality rates that may arise (Cowling et al., 2020; Greenstone & Nigam, 2020; IHME, 2020; Lee & Choe, 2021). Crowd is a condition that occurs when two or more people caught on camera are close together at less than 1.5 meters (Kemenkes RI, 2020). Crowd detection is a research topic regarding the observation of a large collection of people who violating social distance in a certain area.

In crowd detection, a sensor is needed that can capture data to be translated into other forms into information on the state of the crowd. The information can be received in various forms, such as the result of human enumeration in a crowd (Xu et al., 2022), geographical location in a density map (Ozcan et al., 2015), and estimation of social distance between humans and each other (Al-Sa'd et al., 2022). Computer Vision is a specific branch of science that extracts a digital image, then produces information that can be processed into several methods, such as counting methods, measuring distances, or navigation (Krishna, 2017). On the other hand, Convolutional Neural Network (CNN) is a type of neural network that forwards signals into a stack of convolutional layers. The output of the last layer will be spread into a stack called the fully-connected layer (Venkatesan & Li, 2017). Currently, crowd detection using CNN and computer vision is the best solution to anticipate close contact between people in a crowd.

Research on crowd detection in relation to social distancing monitoring has been conducted with different results and conditions. Papaioannidis (Papaioannidis et al., 2021) have created an image segmentation model that can detect crowds to determine the safe flying altitude of UAV's with accuracy

rate ranging from 85% to 98%. However, the image segmentation results cannot be used to estimate social distance. Rezaee et al. (2021) have also trained an image segmentation model for each case of contact between humans using UAVs. The accuracy of detection reaches 97.5% of the 100% scale, but the social distancing violation model is displayed in the same detection box and the detection is not able to estimate the distance between people. Another approach to crowd detection was designed by Shao et al. (2021), which detects pedestrians using UAVs and transforming human head images. The detection accuracy reaches 88.5% for video processing with 75 FPS (frames per second). However, there is no indication of the specific location where a violation occurred, either in the form of contact lines between people or the entire human image.

The objective of this paper is to create a CNN model that can detect crowds of people from the point of view of the UAV camera. The main goal is to build a lightweight model so that the computational process is below the memory capacity of the companion computer in the UAV with high precision and recall values. Next, the new CNN model will measure the estimated distance between humans who are close to each other, with the help of computer vision. From the results of the estimated social distance measurement, a program will be developed to display the counting of social distancing violations between two or more humans who are close together.

This paper is divided into four parts, starting with an introductory chapter that explains the background of this research and related works that explain the crowd detection research that was previously researched. Then, the results and discussion chapter will discuss the human crowd detection model testing output and its analysis. Finally, there are conclusions and future works for the further development of this research.

RELATED WORKS

There are two object detection approaches, which are one-stage object detection and two-stage object detection. The one-stage object detection approach was chosen because high detection inference speed was prioritize (Lohia et al., 2021). Models that belong to the one-stage object detection approach are SSD (Single Shot Detection), RetinaNet, EfficientDet, and others. SSD architecture is based on a

feed-forward convolution network approach. A collection of bounding boxes with a fixed size along with their values will be generated to predict the existence of the object class in the box, followed by a non-maximum suppression step to generate predictions at the final detection stage. (Liu et al., 2016). The main difference between SSD architecture training methods and other detection architectures is that there is no need for a region proposal on the SSD, only a small convolution filter is needed. This filter will be run after the feature mapping layer in the convolution stage to get the class prediction results. Therefore, detection processing can be conducted quickly, and the resulting detection box will be more accurate.

Human crowd detection models have been generated in the study of Papaioannidis et al. (2021) to be used as a sensor for determining the safe flight altitude of a UAV. The method used to perform the detection is a training model based on image segmentation on a convoluted neural network. The experiment was conducted by inserting the model into the embedded system. This model has a high detection accuracy, which is at a confidence value of 85% to 98% depending on the size of the image to be processed. The disadvantage of this study is that it can only detect crowds at general locations and is only intended as a UAV flight altitude determination device. Unfortunately, this model is not suitable for detecting humans at close range and cannot capture detailed images of human objects.

Another crowd model training was conducted by Rezaee et al. (2021) using ShuffleNet, an image segmentation model to capture human objects and create a detection box for humans who are caught in close proximity to each other. Humans will be detected using the Kalman filter method to track human movements from above the UAV. The accuracy obtained is quite high, which is around 97.5% with an average processing time of about 84 milliseconds for each video frame. However, the model cannot calculate the distance between two or more captured human objects. This is because in image segmentation, the coordinate center of one human object cannot be determined from another human object. In addition, the resulting FPS is also quite low, it can only process videos with a size of 11 FPS. The optimization is needed to improve the performance of the related human detection model.

Another approach to detecting crowds of people from above the UAV is to detect human head images and perform transformations so that they are like the representation of location coordinates on

a two-dimensional matrix (Shao et al., 2021). This model is trained using PeleeNet with a high level of precision, which is about 88.22% in video processing at 76 FPS. In experiments using UAV, the level of precision obtained is about 88.5% in video processing of 75 FPS. This model can also be applied to the human object counting system for those who violate social distancing rules. Unfortunately, the model cannot capture the entire human body image. As a result, the results of image extraction will only get human heads and make identification of violators difficult. In addition, there are no markers where the social distancing violations occurred, such as lines or other signs. Therefore, a complete human detection is needed and can display the location of the violation in the image set to be processed.

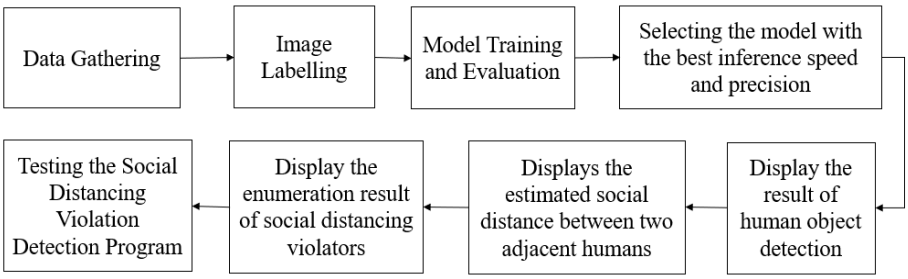
PROPOSED METHOD

Proposed Workflow

In the data gathering process (as shown in Figure 1), the video containing human detection will be extracted into a single image set, each of which will be labeled. Next, a model training and evaluation process will be conducted to produce a model that can adapt to the human crowd image captured from the UAV camera. After that, the model that could detect human objects precisely with the fastest inference time will be selected. Then, the program will be developed to process the model's detection results into a detection box that can be displayed to the screen using Computer Vision. The process of counting camera-captured social distancing violators and storing captured images of social distancing violations will be managed by an algorithm developed in Python. Both the enumeration results and the captured images will be stored in a log in a specific folder that can also be accessed by the user. Finally, an overall program test will be conducted to ensure that the detection program runs well.

Figure 1

Proposed Workflow for Crowd Detection System

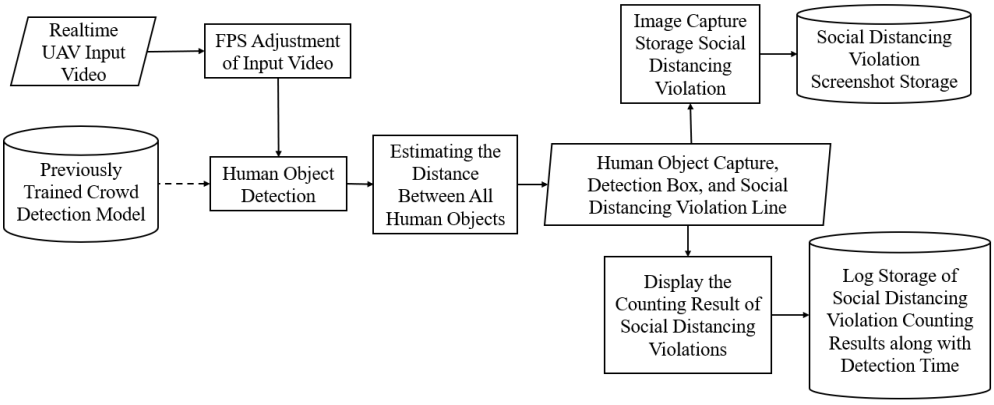


System Architecture Design

When developing a whole crowd detection program, there is a system that helps few processes (as shown in Figure 2). In the initial stage, the video obtained in real-time from the camera on the UAV will be adjusted to FPS at certain frame intervals. After obtaining the image capture in a frame, the human object detection process will then be conducted. The detection model that will be used has previously been stored in the Internal Storage embedded in the UAV's companion computer to make processing the detection results easier. The next step is to get all the detection boxes contained in the captured image of the detection results, the detection box to be processed is the classification of objects that indicate the "human" class. By estimating the distance between all detection boxes, a human object that has violated social distance will be obtained in the form of specific coordinates of the detection box and the estimated number of social distance violations for each detection box. The data will be processed to produce a capture of human objects, detection boxes, and social distance violation lines.

Figure 2

System Architecture Design for Crowd Detection



For the social distancing violation capture image, an image containing two human object violators and one social distancing violation line will be produced. The more detailed image processing results will be stored in the internal storage. On the other hand, the social distancing violation line data will be collected and then the number of lines generated will be enumerated. The number of lines represents the social distancing violations that occur in one image capture at a given time. The results of this enumeration will be written into a file containing a log of the number of social distancing violation events along with the specific time of the event. This log will be stored in the internal storage dedicated

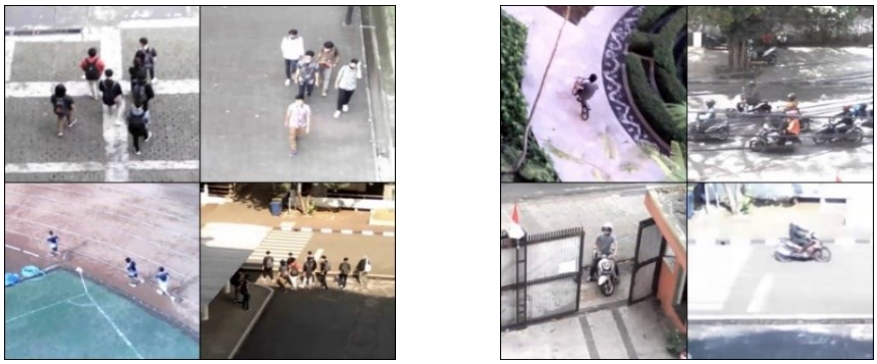
to the number of social distancing violations. The entire process previously described will run continuously until the UAV stops operating. The stop condition is occurred when the state of the UAV after getting the landing command and returning to Home.

Datasets

Images that represent human objects will be collected to conduct model training. Image data retrieval process uses a video extraction process for each frame to be saved in PNG format. PNG format was chosen is because the compression type in this format is lossless. This format can improve the accuracy of object detection at the representation of small pixels (Rahman & Hamada, 2021). Video recording of the crowd is generated by the UAV camera in flight, at an altitude of about 5 to 10 meters from the ground. Positive dataset (can be seen in Figure 3 part a) contains human objects when standing or walking and is categorized as a "human" class. On the other hand, negative dataset (can be seen in Figure 3 part b) contains objects such as cyclists or motorcyclists which are categorized as "non_human" class. Therefore, the amount of positive data with negative data is balanced so that there is no oversampling in certain classes.

Figure 3

Example of Positive Dataset (a) and Negative Dataset (b)



(a) **(b)**

Videos containing crowds were recorded in the morning to evening timeframe so that the camera could capture images with bright and clear conditions. After validating the objects contained in the image, 10000 images from video extraction in sizes of 640 × 480 pixels have been collected. Datasets that have been collected will be labeled according to their respective class categories using the labelImg

tool (Tzutalin, 2015). The output of the image data labeling process is in XML format following the PASCAL VOC convention (Everingham et al., 2010). Furthermore, the image datasets that have been assigned class category labels will be divided into three folders, more details can be seen in Table 1.

Table 1

Dataset Splitting

Folder	Positive Images	Negative Images	Total Images
train	4800 (48%)	4800 (48%)	9600 (96%)
dev	100 (1%)	100 (1%)	200 (2%)
test	100 (1%)	100 (1%)	200 (2%)
Total	**5000 (50%)**	**5000 (50%)**	**10000 (100%)**

Transfer Learning from Pre-trained Model

For the selection phase of the pre-trained model, considerations are made from the side of the most optimal speed with high enough precision (Table 2, which have bold font). SSD MobileNet V2 FPNLite is selected because the model provides a high detection speed and a better level of precision than other MobileNet models. With the addition of this FPNlite feature, objects with small sizes will be able to be detected better than the standard MobileNet V2 model (Li et al., 2019). Moreover, the SSD ResNet50 and the SSD ResNet101 were chosen with a layer size of 640×640 compared to 1024×1024 because they fit the needs of model training and testing. The SSD ResNet152 was not selected due to the excessive number of layers and the final model memory being inefficient for the case of crowd detection using the UAV companion computer.

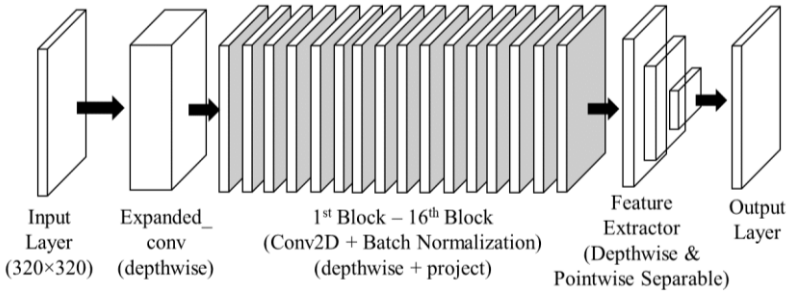
Table 2

Single Shot Detection Pre-trained Model in TensorFlow Model Zoo (TensorFlow, 2022)

Model Name	Speed (ms)	COCO mAP	Output
SSD MobileNet v2 320×320	19	20.2	Boxes
SSD MobileNet V1 FPN 640×640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320×320	**22**	**22.2**	**Boxes**
SSD MobileNet V2 FPNLite 640×640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640×640 (RetinaNet50)	**46**	**34.3**	**Boxes**
SSD ResNet50 V1 FPN 1024×1024 (RetinaNet50)	87	38.3	Boxes
SSD ResNet101 V1 FPN 640×640 (RetinaNet101)	**57**	**35.6**	**Boxes**
SSD ResNet101 V1 FPN 1024×1024 (RetinaNet101)	104	39.5	Boxes
SSD ResNet152 V1 FPN 640×640 (RetinaNet152)	80	35.4	Boxes

Figure 4

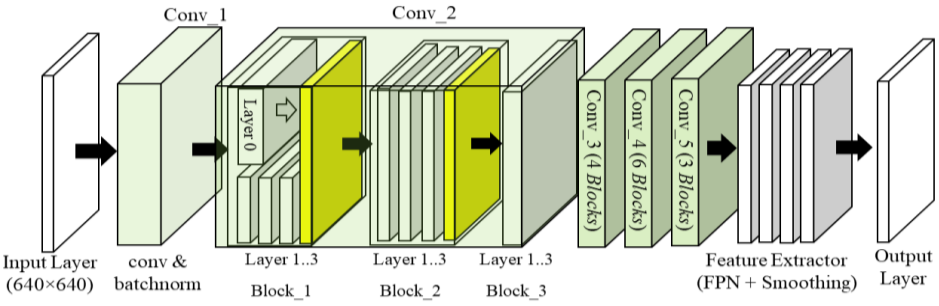
Layer of SSD MobileNet V2 FPNLite 320×320



In the MobileNet Architecture (as shown in Figure 4), the very first layer will receive inputs with dimensions of 320 × 320. After that, there is an additional convolution which is the base layer of the SSD architecture. Furthermore, the output of the additional convolution layer will enter 16 blocks which are useful for depthwise separable process along with batch normalization process. This normalization is useful for scaling the output of each previous layer so that the input to the convolutional layer afterwards becomes more adaptive. Finally, depthwise and pointwise feature processing will be performed. For depthwise, the process is performed at a kernel size of 3 × 3, while at pointwise, the kernel size used is simply 1 × 1 dimension.

Figure 5

Layer of SSD ResNet50 V1 FPN 640×640 (RetinaNet50)

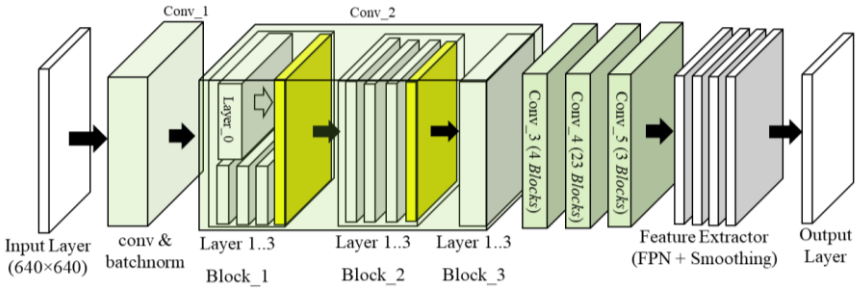


In the ResNet50 Architecture, the input layer dimension size is 640×640, and will be processed in conv_1 as the base layer of the SSD Architecture. Furthermore, the output of conv_1 will enter conv_2. In the conv_2 section, there are 3 main blocks, and each block has 3 layers. In the first block, there is a Layer_0 that will directly send the results to the temporary output in the same block (can be seen in Figure 5 in the yellow layer). This is in accordance with the residual learning principle on operations

within a particular block. For other blocks, the process runs sequentially like any other process. After all blocks have been processed, the output of the convolution will be input to the next convolution process. While the ResNet convolution process has been completed, the final feature extraction and normalization process will be conducted to determine the class classification along with the location of the detection box. The result of this process will be forwarded to the output layer.

Figure 6

Layer of SSD ResNet101 V1 FPN 640×640 (RetinaNet101)



The layer architecture in ResNet101 is quite like ResNet50. First, the input data is received with dimensions of 640×640 and will be processed in conv_1 as the base layer of SSD Architecture. Next, the residual learning system will be applied to block_1 in layer 0 whose output will be stored while waiting for other block processing. The number of blocks in the convolution layer is more than the ResNet50 model. As seen in Figure 6, the number of blocks in conv_4 is 23 units, more than the ResNet50 model (only 6 blocks). After the convolution process is complete, the feature extraction and final normalization process is conducted to determine the class classification and the location of the detection box.

Hyperparameter Tuning

For all pre-trained models, hyperparameter adjustments will be made to fit the training process on a larger dataset. For tuning phase, development dataset (about 200 images) will be used to support the best-fit method armed with experience from numbers obtained in related research (Makirin et al., 2021; Wastupranata & Munir, 2021). There are several hyperparameters that need to be tuned, such as Learning Rate Base, Cosine Decay Step (Huang et al., 2017; Loshchilov & Hutter, 2017; Smith, 2018), Exponential Decay Step (López, 2020), Warmup Learning Rate (Gotmare et al., 2019; Goyal et al.,

2017), and Warmup Step (Phong et al., 2022). The best combination of hyperparameters is expected to increase the speed of the training process on a larger amount of data without worrying about the precision and sensitivity of the model (Lyon, 2017; Ruder, 2016). Hyperparameter tuning is using GPU computing environment with a total epoch of 100000 steps. The hyperparameter combination in each model is selected based on the lowest total loss value, the highest precision value, and the highest sensitivity (recall) value, as can be seen in Table 3 with bold font.

Table 3

Hyperparameter Testing

Optimizer	Batch Size	Learning Rate Base	Cosine Decay Step	Exp Decay Step	Warmup Learning Rate	Warmup Step	Total *Loss*	*Precision* IoU= 0,50:0,95	*Recall AR@100*
SSD MobileNet V2 FPNLite 320×320 Configuration									
Momentum	12	0.040	100000	-	0.013	5000	0.074	0.821	0.858
Momentum	**12**	**0.080**	**100000**	**-**	**0.027**	**5000**	**0.047**	**0.832**	**0.866**
Momentum	12	0.040	100000	-	0.040	0	0.071	0.823	0.862
Momentum	12	0.040	100000	-	0.027	5000	0.068	0.814	0.850
RMS_Prop	12	0.004	-	5000	-	-	1.959	0.000	0.036
RMS_Prop	12	0.040	-	500	-	-	0.879	0.389	0.577
SSD ResNet50 V1 FPN 640×640 (RetinaNet50) Configuration									
Momentum	12	0.040	100000	-	0.013	5000	0.046	0.821	0.854
Momentum	12	0.080	100000	-	0.027	5000	0.043	0.793	0.828
Momentum	12	0.040	100000	-	0.040	0	0.050	0.780	0.813
Momentum	**12**	**0.040**	**100000**	**-**	**0.027**	**5000**	**0.039**	**0.807**	**0.835**
RMS_Prop	12	0.004	-	5000	-	-	0.578	0.569	0.662
RMS_Prop	12	0.040	-	500	-	-	0.981	0.258	0.547
SSD ResNet101 V1 FPN 640×640 (RetinaNet101) Configuration									
Momentum	8	0.040	100000	-	0.013	5000	0.062	0.801	0.835
Momentum	8	0.080	100000	-	0.027	5000	0.066	0.803	0.831
Momentum	**8**	**0.040**	**100000**	**-**	**0.040**	**0**	**0.061**	**0.794**	**0.829**
Momentum	8	0.040	100000	-	0.027	5000	0.053	0.786	0.826
RMS_Prop	8	0.004	-	5000	-	-	0.608	0.599	0.678
RMS_Prop	8	0.040	-	500	-	-	0.726	0.579	0.700

Ratio of Social Distance to Human Height

A digital image consists of constituent elements in the form of pixels with a limited size and has a defined value for each pixel. Digital image representation is two-dimensional matrix with the elements represented by pixel values at each location (Sonka et al., 2014). To measure the distance between two defined pixels, the Euclidian distance formula can be used as shown in Equation 1.

$$D_E\big((i,j),(h,k)\big) = \sqrt{(i-h)^2 + (j-k)^2} \quad (1)$$

where,

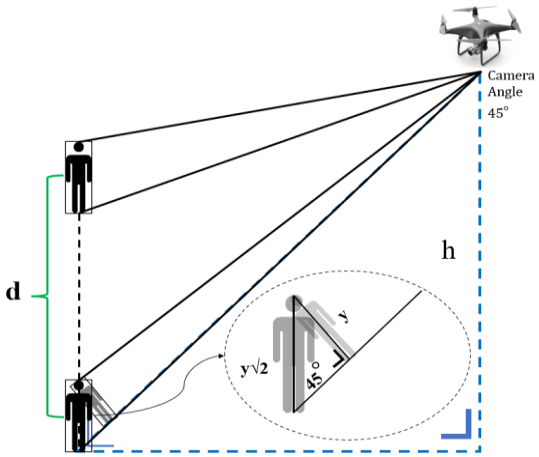
(i, j) = pixels at the starting point,

(h, k) = pixels at the target point

To estimate the social distance between two or more people, the human height will be used as a basis for measurement. Distance measurement using the human height reference value should be considered further from the point of view of a particular UAV camera. Social distance reference according to human height caught by UAV camera can be seen in Figure 6.

Figure 7

Illustration of Human Height Captured as a Reference for Estimating Social Distance



By using the ratio of height y to get d, the value of d will be generated as shown in Equation 2.

$$d = \frac{\Delta pixel_d}{\Delta pixel_y} \times y \tag{2}$$

$$y = \frac{y\sqrt{2}}{\sqrt{2}} = \frac{1.6\ m}{\sqrt{2}} \approx 1.13\ m \tag{3}$$

where,

d = social distance estimation (meters),

$\Delta pixel_d$ = the number of pixels between two humans that are d apart (pixels),

$\Delta pixel_y$ = the number of pixels that represents the human height from the UAV camera (pixels),

y = human height projection (meters)

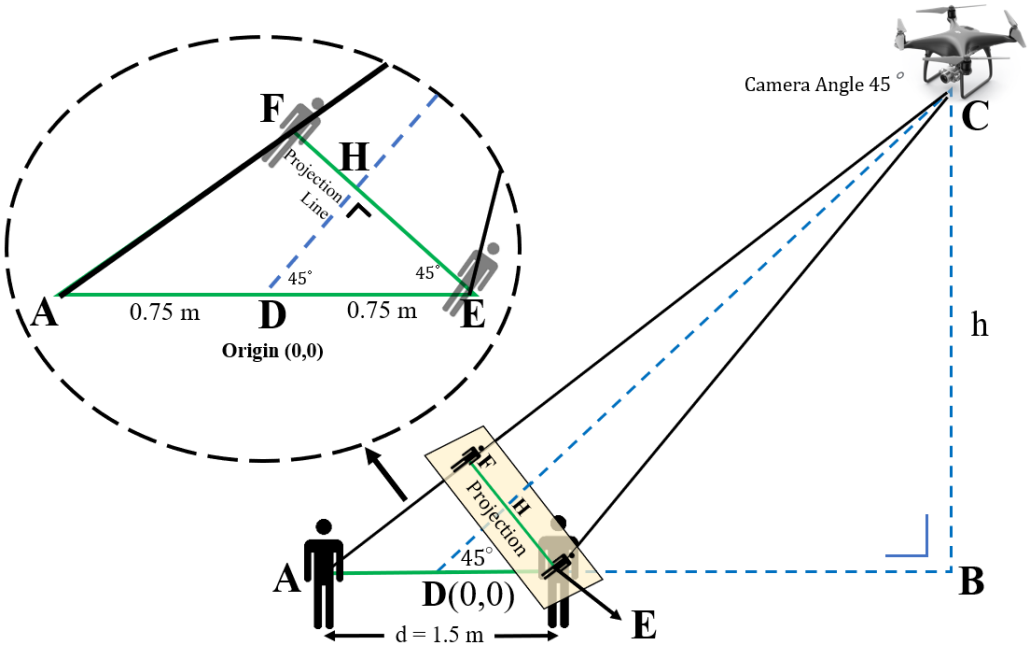
Thus, the human height captured by the camera will be approximated as high as 1.13 meters to the projection of the camera angle on the UAV by 45° (as shown in Equation 3). The value of y will be substituted into Equation 2 to estimate the social distance.

Calibration of Human Coordinate Components Parallel to UAV Camera Viewpoint

The vertical angle of view of the camera determines the human height reference due to the projection on the object that will appear on the detection screen. The flying height of the UAV will be used as a basis in determining the projection to determine the distance of humans who are close to each other from the parallel side of the UAV camera's point of view. Therefore, it is necessary to calibrate the component of human coordinates that are parallel to the point of view of the UAV camera and then substitutes into the Euclidian distance formula in Equation 1. Illustration of adjusting the projection of human object coordinates parallel to the UAV camera's point of view can be seen in Figure 8.

Figure 8

Illustration of Adjusting Human Object Coordinate Projection with UAV Camera Parallel View Angle



The calculation to get the human coordinates projection that is parallel to the camera's point of view is in the letter FE to the AE line. The AE line is 1.5 meters long (social distancing requirement) so that the AD Line and DE Line have a length of 0.75 meters to the center of the camera axis. Since the camera angle has a 45° angle to the UAV's maneuverability, the HDE angle also has a 45° angle. Since the projection is always perpendicular to the plane on which it is projected, the angles at point H are all 90°. Next, the Euclidean equation will be used to determine the social distance for the angle of view that is parallel to the UAV camera while flying, as shown in Equation 5.

$$d_{\overline{FE}} = \sqrt{\left(\frac{9}{4(8h+3)} - \frac{3}{4}\right)^2 + \left(\frac{6h}{8h+3}\right)^2} \quad (5)$$

where,

$d_{\overline{FE}}$ = distance projection estimation (meters),

h = UAV altitude

It is necessary to calibrate and adjust the value of the y-axis to the flying height of the UAV (h variable) in the coordinates of the estimated social distance. Therefore, the calculation of the new Euclidian formula previously seen in Equation 1, can be seen in Equation 6.

$$d_h = \sqrt{(x_1 - x_2)^2 + \left(\frac{\sqrt{\left(\frac{9}{4(8h+3)} - \frac{3}{4}\right)^2 + \left(\frac{6h}{8h+3}\right)^2}}{\frac{3}{2}}(y_1 - y_2)\right)^2} \quad (6)$$

where,

d_h = social distance estimation from UAV altitude of h (pixels),

x_1, x_2 = pixel of x in starting point and target point (pixels),

y_1, y_2 = pixel of y in starting point and target point (pixels)

Euclidian distance in Equation 6 depends on the value of h variable, so the calibration of the y variable will be different for any given height. The UAV flight test is conducted at a constant altitude, so that changes in the value of h will not occur during the program compilation process. To get the original social distance in meters, substitute again in Equation 2 as $\Delta pixel_d$.

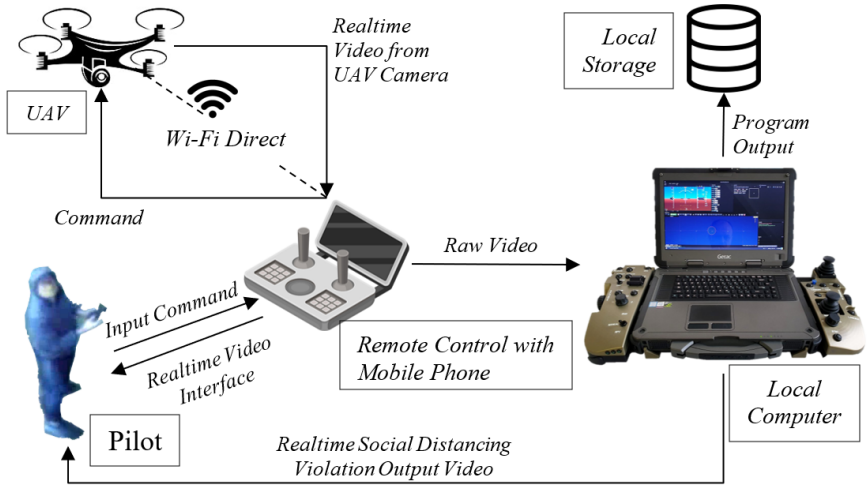
Experimental Setup

The object detection test will be conducted with the Test dataset which has a different image set from the Train and Development dataset. Then, runtime detection test will be conducted on a single image for different resolutions. One of the best models will be selected which will then be implemented in the program to display the results of human object detection with the approximate distance between the human objects. The pilot will turn on the UAV along with the remote control which is the main component in the crowd detection system. The Mobile Phone will be connected to the remote control

as a tool in handling video matters. The pilot will maneuver the UAV and search for crowd points via video transmitted from the UAV camera via the Wi-Fi Direct system. The input video is not processed directly on the UAV but is done on a local computer in real time. This processing method is to overcome the UAV specifications that are not able to embed a companion computer (microprocessor). Furthermore, evaluating the entire human detection system can be seen in Figure 9.

Figure 9

Architecture diagram of crowd detection system testing



The UAV must be in a radius of less than 50 meters from the remote-control location to avoid loss of contact. After the UAV has successfully flown at a certain altitude, the mobile phone will send raw video to a local computer that contains a program to calculate the number of social distancing violations. The result of the violation will be displayed on the Pilot and first stored on local storage. The testing process will keep looping until the UAV has landed perfectly and program execution has been stopped. To minimize detection delays, a detection will be performed for each specific frame interval. For every N frame, one detection will be performed, including the measurement of the distance between humans, and other operations. This interval setting is also useful for getting objects with other positions that may be caught on camera, so time does not run out just to detect at the same position. In addition, the memory used to store the image of the violator will also be less, so the memory can be used for other purposes.

Hardware and Resources

The UAV which is used for crowd video recording is the TXD 8S(L) Drone Wi-Fi HD Camera. For training phase, Google Colab provides a single 12GB NVIDIA Tesla K80 GPU that can be used up to 6 hours continuously. The program development and testing phase are using a Computer with Intel® Core™ i7-9750H CPU @ 2.60GHz, with 8.192 MB RAM.

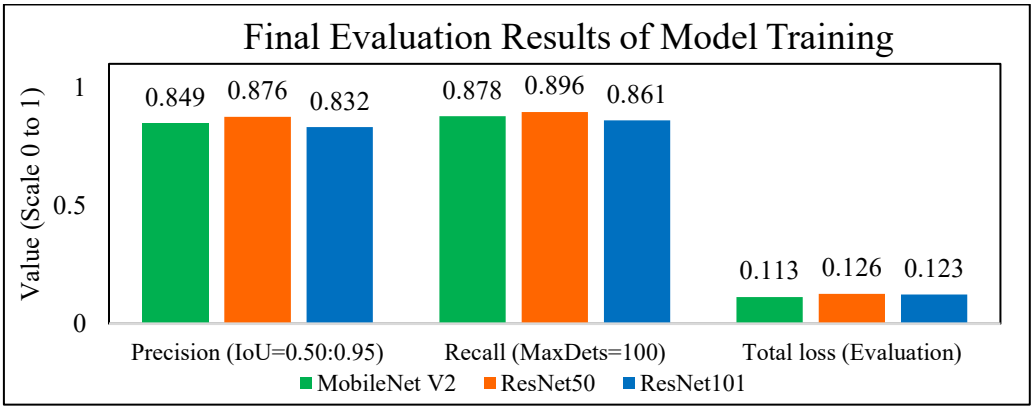
RESULTS AND DISCUSSION

Crowd Detection Model Test Results

There are three test metrics conducted at the final evaluation stage of the model, depicted by a bar graph as can be seen in Figure 10.

Figure 10

Graph of Final Evaluation Results of Model Training



The IoU metric represents how big the wedge area is between the detection box on ground truth and the detection box formed from the prediction data. The equation of the IoU metric can be seen in Equation 6.

$$IoU(p,a) = \frac{(Box_T \cap Box_P)}{(Box_T \cup Box_P)} \quad (6)$$

where,

$IoU(p,a)$ = Introduction over Union [0..1],

Box_T = pixel of x in starting point and target point (pixels),

Box_P = pixel of y in starting point and target point (pixels)

All human crowd detection models yield more than 80% precision and recall values. In addition, all human detection models also have a low loss value, which is below the 0.2 scale. Precision metric is used to determine the ratio between the correct prediction data detection boxes compared to the overall detection results as shown in Equation 7. In addition, the recall metric is used to determine the ratio between the correct prediction data detection boxes compared to the overall label data that should be formed (ground truth) as shown in Equation 8.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{7}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{8}$$

where,

$True\ Positive$ = positive class detection results are in accordance with the basic truth,

$False\ Positive$ = detection result is a positive class, but the basic truth should be a negative class,

$False\ Negative$ = detection result is a negative class, but the ground truth should be a positive class

Precision value exceeding 80% indicates that the number of True Positives is four times greater than the number of False Positives (Equation 7). That is, the number of images detected as "human" corresponds to the ground truth of the category which is also "human". Only less than 20% of “human” images were incorrectly detected as “non_human”. Furthermore, if the recall value exceeds 80%, the number of True Positives is four times greater than the number of False Negatives (Equation 8). This indicates that the number of images detected are of the "human" class and identical. with the basic truth which is also “human” category. Less than one-fifth of the image is detected as “non_human” even though it has the basic truth of “human”. In determining True Positive and False Negative, IoU will be involved for certain value limits. If the IoU between the predictive data detection box and the ground truth is higher than 0.5, the area can be defined as TP. Otherwise, it will be FP (Shen et al., 2020).

Regarding the loss value which has a scale of less than 0.2, there are two variables that have an impact on the total loss calculation. This loss function impacted by the classification loss value and the localization loss value.

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \quad (9)$$

where,

N = number of detection boxes in a detection,

x = image to be detected,

c = the predictive value of the predictive confidence,

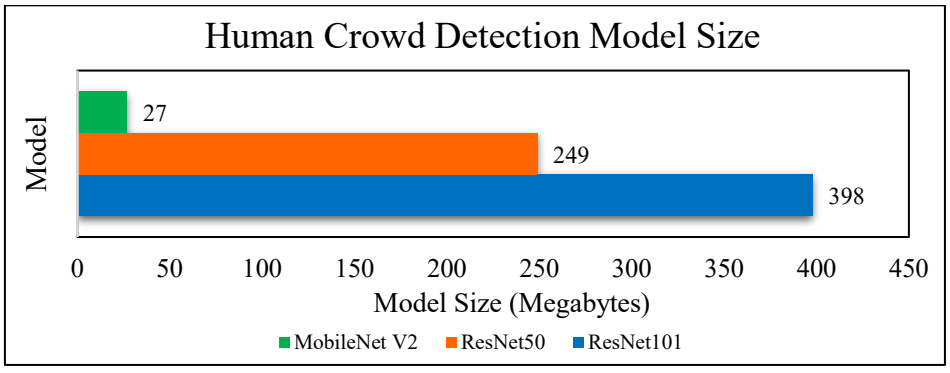
l = the detection box formed from the results of object detection,

g = label data detection box for ground truth,

As seen in Equation 9, the error caused in the category classification phase is ridiculously small. The low value of classification loss can also be caused by a high detection confidence level. On the other hand, the low value of localization loss is supported by the appearance of a detection box that matches the ground truth coordinates that have been defined in the image data set in XML format. The best precision and sensitivity values were obtained by the ResNet50 model, with a precision value reaching 87.6% and a sensitivity (recall) value reaching 89.6%. However, the lowest loss value among all crowd detection models is obtained from the MobileNet V2 metric model, which is 0.113. However, MobileNet is a crowd detection model with the smallest size compared to the other two models, only having a size of 27 Megabytes (Figure 11).

Figure 11

Human Crowd Detection Model Size Chart



Furthermore, the testing of the human crowd detection model was conducted by entering two images, namely the first image containing humans and the second image containing motorcycle riders. The results of the runtime testing of each model can be seen in Table 4.

Table 4

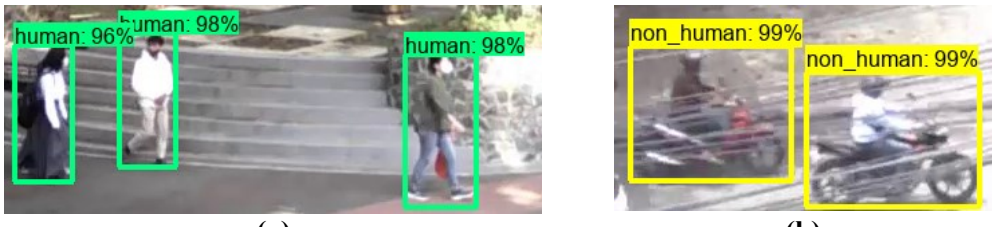
Human Detection Runtime Test Results

Model	Resolution	Load Time (s)	Image #1 Runtime (s)	Image #2 Runtime (s)	Detection Runtime Average (s)
MobileNet V2	480 × 360	7.387	1.044	0.117	0.581
	640 × 480	6.863	1.099	0.119	0.609
	960 × 720	7.471	1.101	0.129	0.615
	1440 × 1080	7.286	1.102	0.141	0.621
	Average	7.252			0.606
ResNet50	480 × 360	7.201	1.478	0.389	0.934
	640 × 480	7.515	1.501	0.392	0.947
	960 × 720	9.123	1.685	0.430	1.058
	1440 × 1080	9.438	1.701	0.444	1.073
	Average	8.319			1.003
ResNet101	480 × 360	12.472	1.936	0.541	1.239
	640 × 480	13.685	1.972	0.541	1.256
	960 × 720	13.748	2.192	0.615	1.403
	1440 × 1080	15.022	2.207	0.661	1.434
	Average	13.732			1.333

Model setup time is the time measured during the model initiation phase until the model is ready for use. Detection time is the time measured when the human crowd detection module is run until the output is the coordinates of the detection box and the number of objects detected (Figure 12). The duration of image processing is very dependent on the size of the image resolution received by the human crowd detection model. The larger the resolution size of the human crowd image, the greater the human detection time. For this reason, it is necessary to pay attention to the size of the input video resolution when using the human crowd detection model if it has a time constraint.

Figure 12

Example of "human" Detection (a) and "non_human" Detection (b) result



(a) **(b)**

Considering the results of precision values, sensitivity, total loss, and model size, MobileNet V2 was chosen as the best human crowd detection model. Furthermore, the testing of the social distance

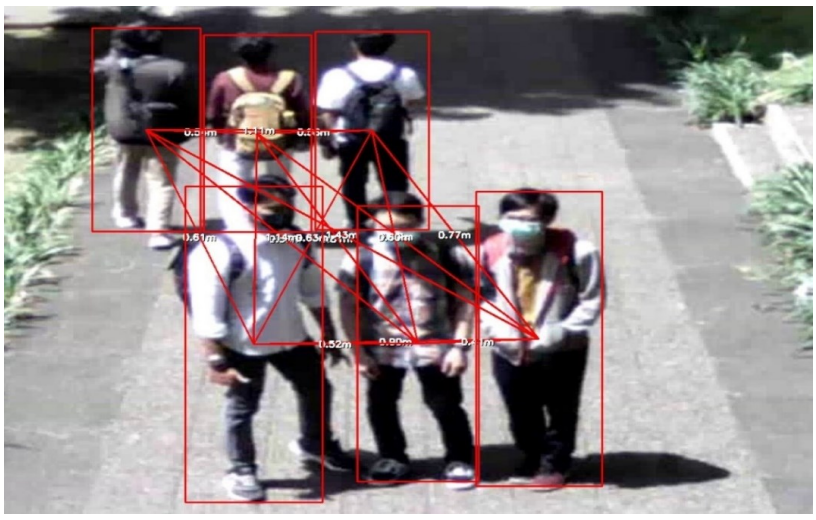
estimation module and the program for calculating the number of social distancing violations will use the MobileNet V2 model.

Social Distance Estimation and Calibration Test Results

Furthermore, the test is conducted at different altitude. The flying altitude of the UAV is used as one of the factors to determine social distance calibration and has an indirect impact on the sensitivity of the human detection model. Social distance estimation is conducted for every two people detected in proximity. An example of an image from the estimation of social distance at a height of ±5 meters can be seen in Figure 13. The lines of social distancing violations are interconnected with each other. For this reason, it is necessary to calculate the number of violators and the number of social distancing violations to find out the number of people that participate in the crowd. Calibration will make a correction to the difference in the angle of the UAV camera while flying. By calibrating, the excess distance caused by the illusion of a viewing angle can be overcome as shown in Equation 6.

Figure 13

Example of the Estimation of Social Distance at a Height of ±5 Meters



Evaluating the Program for Calculating the Number of Social Distance Violations

With the formation of a crowd of people in one place, the potential for social distancing violations is extremely high. The program will be evaluated to ensure that the integration between the human detection module, the social distance estimation and calibration module, and the social distance

enumeration module is not a problem. Human crowd video in MP4 format will be input to the program and produce video output in MP4 format, as seen in the architecture of the crowd detection system testing system (Figure 9). An example of the calculation results of violators and social distancing violations can be seen in Figure 14.

Figure 14

Example of Violation Calculation Results (Red Text) and Social Distance Violation (Orange Text)

The program succeeded in calculating the number of violators and social distancing violations. As seen in Figure 14, five humans were detected as social distancing violators. In addition, there are also 5 social distancing violations for every two humans who are close together. Because all human objects are within the padding limit, the total human height has been calculated and can be used as a reference to calculate social distance. By using the Oxford Town Center Dataset (Benfold & Reid, 2011), this paper has advantages compared to previous studies (see Table 5).

Table 5

The Comparison Between This Paper and Other Previous Studies on Oxford Town Center Dataset

Research	Backbone	Precision	Measured Social Distance	Counting Social Distancing Violators
Rezaee et al. (2021)	ShuffleNet	88.45%	-	-
Elbishlawi et al. (2021)	DETR + ResNet50	43.4%	-	-
Özbek et al. (2021)	Darknet-53 + YOLOv3	55.3%	✓	-
Madane & Chitre (2021)	ResNet50	94.23%	-	-
Ahmad et al., (2022)	YOLOv3	97%	✓	-
Wastupranata & Munir **(Proposed)**	MobileNet V2	82%	✓	✓

CONCLUSIONS

Three human detection models were successfully created using the MobileNet, ResNet50, and ResNet101 pre-trained models. All models can detect humans, cyclists, and motorcyclists with precision and sensitivity values above 80%. All trained models also did not experience overfitting during training, as evidenced by the loss function value below the 0.2 scale. MobileNet V2 was chosen as the detection model for further implementation in the social distance calculation program. This is because the MobileNet V2 model has a file size only 19 Megabytes, so the detection process can be conducted smoothly according to the computational load that can be managed by the UAV companion computer. The precision value of MobileNet V2 reaches 84.9% (IoU=0.50:0.95), with a sensitivity value (recall) reaching 87.8% (MaxDets=100).

The estimation of social distance was successfully conducted by using the average human height in Indonesia as a reference, which is 1.6 meters. The social distance calibration formula for the social distance component that is parallel to the UAV camera's point of view has been successfully implemented in the program so that the estimated social distance is close to the original distance. However, the flying height of the UAV must be determined in advance so that the estimated social distance can be properly calibrated. The social distancing violation calculation program has been successfully integrated with the crowd detection model and the social distance estimation and calibration calculation module.

FUTURE WORKS

In the future, an improved architectural model will be conducted that can detect crowds of people more quickly. In addition, hyperparameter tuning can be done with other variables to increase the accuracy and sensitivity of the resulting model. Furthermore, the number of images for model training should be increased, so that the UAV can detect human crowds in a more heterogeneous environment. Thus, the program can be further developed to conduct human tracking so that the movements of social distance violators can be further traced. The measurement of social distance will be developed using proximity sensors that are integrated with the UAV companion computer. The UAV's flight altitude

measured from the ground can also be determined using the proximity sensor. It is also possible to develop a crowd detection model in darker places, using a thermal sensor. The detection of social distancing violators can also be conducted on humans with temperatures higher than the normal reference so that it can be seen whether the human being is suspected of being a COVID-19 suspect.

ACKNOWLEDGMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Ahmad, I., Xu, S. J., Khatoon, A., Tariq, U., Khan, I., Rizvi, S. S., & Ullah, A. (2022). Analytical Study of Deep Learning-Based Preventive Measures of COVID-19 for Decision Making and Aggregation via the RISTECB Model. *Scientific Programming*. https://doi.org/10.1155/2022/6142981

Al-Sa'd, M., Kiranyaz, S., Ahmad, I., Sundell, C., Vakkuri, M., & Gabbouj, M. (2022). A Social Distance Estimation and Crowd Monitoring System for Surveillance Cameras. *Sensors*, *22*(2), 1–21.

Benfold, B., & Reid, I. (2011). Stable Multi-Target Tracking in Real-Time Surveillance Video. *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, 3457–3464. https://doi.org/10.1109/CVPR.2011.5995667

Cowling, B. J., Ali, S. T., Ng, T. W. Y., Tsang, T. K., Li, J. C. M., Fong, M. W., Liao, Q., Kwan, M. Y., Lee, S. L., Chiu, S. S., Wu, J. T., Wu, P., & Leung, G. M. (2020). Impact assessment of non-pharmaceutical interventions against coronavirus disease 2019 and influenza in Hong Kong: an observational study. *The Lancet Public Health*, *5*(5), e279–e288.

Elbishlawi, S., Abdelpakey, M. H., & Shehata, M. S. (2021). SocialNet: Detecting Social Distancing Violations in Crowd Scene on IoT devices. *7th IEEE World Forum on Internet of Things, WF-IoT 2021*, 801–806. https://doi.org/10.1109/WF-IoT51360.2021.9595383

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The PASCAL

Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Gotmare, A., Shirish Keskar, N., Xiong, C., & Socher, R. (2019). A Closer Look at Deep Learning Heuristics: Learning rate restarts, Warmup and Distillation. *7th International Conference on Learning Representations, ICLR 2019*.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., & He, K. (2017). *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*.

Greenstone, M., & Nigam, V. (2020). Does Social Distancing Matter? *SSRN Electronic Journal*.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., & Murphy, K. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 3296–3305. https://doi.org/10.1109/CVPR.2017.351

IHME. (2020). Forecasting COVID-19 impact on hospital bed-days, ICU-days, ventilator-days and deaths by US state in the next 4 months. *MedRxiv*.

Kemenkes RI. (2020). *Protokol Pemicuan dan Verifikasi 5 Pilar STBM*.

Krishna, R. (2017). *Computer Vision: Foundations and Applications*. Stanford University.

Lee, J. K., & Choe, Y. J. (2021). The Impact of Social Distancing on the Transmission of Influenza Virus, South Korea, 2020. *Osong Public Health and Research Perspectives*, *12*(1), 91–92.

Li, X., Lai, T., Wang, S., Chen, Q., Yang, C., & Chen, R. (2019). Feature Pyramid Networks for Object Detection. *Proceedings - 2019 IEEE Intl Conf on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking, ISPA/BDCloud/SustainCom/SocialCom 2019*, 1500–1504.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., & Berg, A. C. (2016). SSD : Single Shot MultiBox Detector. *European Conference on Computer Vision*, 21–37.

Lohia, A., Kadam, K. D., Joshi, R. R., & Bongale, A. M. (2021). Bibliometric Analysis of One-stage and Two-stage Object Detection. *Library Philosophy and Practice (LPP)*, *February*.

López, J. G. (2020). *Geometric computer vision meets deep learning for autonomous driving*

applications.

Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 1–16.

Lyon, R. F. (2017). Neural Networks for Machine Learning. In *Human and Machine Hearing* (pp. 419–440).

Madane, S., & Chitre, D. (2021). Social Distancing Detection and Analysis through Computer Vision. *2021 6th International Conference for Convergence in Technology, I2CT 2021*, 1–10. https://doi.org/10.1109/I2CT51068.2021.9418195

Makirin, M. K., Wastupranata, L. M., & Daffa, A. (2021). Onboard Visual Drone Detection for Drone Chasing and Collision Avoidance. *AIP Conference Proceedings*, *2366*.

Özbek, M. M., Syed, M., & Öksüz, I. (2021). Subjective analysis of social distance monitoring using YOLO v3 architecture and crowd tracking system. *Turkish Journal of Electrical Engineering and Computer Sciences*, *29*(2), 1157–1170. https://doi.org/10.3906/ELK-2008-66

Ozcan, A. H., Unsalan, C., & Reinartz, P. (2015). Sparse people group and crowd detection using spatial point statistics in airborne images. *RAST 2015 - Proceedings of 7th International Conference on Recent Advances in Space Technologies*, 307–310.

Papaioannidis, C., Mademlis, I., & Pitas, I. (2021). Autonomous UAV Safety by Visual Human Crowd Detection Using Multi-Task Deep Neural Networks. *2021 IEEE International Conference on Robotics and Automation (ICRA 2021)*, 11074–11080.

Phong, N. H., Santos, A., & Ribeiro, B. (2022). PSO-Convolutional Neural Networks With Heterogeneous Learning Rate. *IEEE Access*, *10*(1), 89970–89988.

Qian, M., & Jiang, J. (2020). COVID-19 and social distancing. *Canadian Journal of Addiction*, *11*(2), 4–6.

Rahman, M. A., & Hamada, M. (2021). PCBMS: A Model to Select an Optimal Lossless Image Compression Technique. *IEEE Access*, *9*, 167426–167433.

Rezaee, K., Mousavirad, S. J., Khosravi, M. R., Moghimi, M. K., & Heidari, M. (2021). An

Autonomous UAV-Assisted Distance-Aware Crowd Sensing Platform Using Deep ShuffleNet Transfer Learning. *IEEE Transactions on Intelligent Transportation Systems*.

Ruder, S. (2016). *An Overview of Gradient Descent Optimization Algorithms*. 1–14.

Shao, Z., Cheng, G., Ma, J., Wang, Z., Wang, J., & Li, D. (2021). Real-time and Accurate UAV Pedestrian Detection for Social Distancing Monitoring in COVID-19 Pandemic. *IEEE Transactions on Multimedia*, 1–16.

Shen, H., Du, L., Zhang, L., & Gong, W. (2020). Priority Branches for Ship Detection in Optical Remote Sensing Images. *Remote Sensing*, 1–19.

Smith, L. N. (2018). *A disciplined approach to neural network hyper-parameters: Part 1 -- learning rate, batch size, momentum, and weight decay*. 1–21.

Sonka, M., Hlavac, V., & Boyle, R. (2014). *Image Processing, Analysis, and Machine Vision*. Cengage Learning.

TensorFlow. (2022). *TensorFlow 2 Detection Model Zoo*. GitHub. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

Tzutalin. (2015). *LabelImg*. Git Code. https://github.com/heartexlabs/labelImg

Venkatesan, R., & Li, B. (2017). *Convolutional Neural Networks in Visual Computing: A Concise Guide*. CRC Press.

Wastupranata, L. M., & Munir, R. (2021). UAV Detection using Web Application Approach based on SSD Pre-Trained Model. *Proceedings of the 2021 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology, ICARES 2021*.

Xu, J., Zhao, H., Min, W., Zou, Y., & Fu, Q. (2022). DGG: A Novel Framework for Crowd Gathering Detection. *Electronics (Switzerland)*, *11*(1).