IMPLEMENTATION TRANSFER LEARNING CONVOLUTIONAL NEURAL NETWORK FOR POTHOLE DETECTION ON DRONE VIDEO

Miftahul Muhaemen[1], Mohammad Reza Faisal [2], Dodon Turianto Nugrahadi[3], Andi Farmadi[4], Rudy Herteno[5]
[12345]Ilmu Komputer FMIPA ULM
A. Yani St. KM 36 Banjarbaru, South Kalimantan
Email : miftahulmuhaemen@gmail.com

Abstract

Pothole is one of the problems that can cause harm to a person or a lot of people and can even cost lives. So a lot of research has been done to detect potholes, especially image-based. This research uses Unmanned Aerial Vehicle (UAV) to get aerial video dataset and train Convolutional Neural Network (CNN) with the dataset. However, instead of doing learning from the beginning, transfer learning can be used to train CNN to recognize the object of a pothole and measure the value of its performance and what the optimal frame rate is. Then the results of this study indicate that the CNN model, ssd_resnet_50_fpn_coco gets an average performance value of 48.90 mAP. And the optimal frame rate with the average highest performance value at a frame rate of 30FPS with a value of 49.43 mAP, followed by 1FPS with a value of 48.36 mAP .

Keywords*: Performance, Transfer Learning, Convolutional Neural Network, Pothole Detection, Aerial Video*.

1. INTRODUCTION

Potholes become one of the problems that can cause harm to a person or a large number of people and can even take lives, so a lot of research has been done to detect potholes, especially image-based such using road contour imagery taken with a digital camera resolution of 640x480 pixels and the intensity of bright light at midday and identification made using edge detection [8] can identify damage to road contours especially potholes with an accuracy above 80% , but researchers still have to manually retrieve the required image data.

The length and the number of roads make it difficult to capture images, this can be solved by using Unmanned Aerial Vehicle (UAV). UAV systems have the ability to capture the images of roads faster and inspect more accurately than humans [7].

One method that can be used to recognize potholes is the Convolutional Neural Network (CNN) [1, 2]. CNN is one method of deep learning that is capable of conducting independent learning for the introduction, extraction and classification of objects such applying object recognition using CNN on rice, onion, coconut, banana and chilli plants [2].

CNN can also be trained to recognize pothole objects using UAV aerial video datasets. However, instead of doing learning from the beginning, it can use transfer learning [1]. Imports pre-trained models that have been trained with similar and larger datasets and then retrains the model with their image datasets using best practices, cyclic differential learning and data augmentation. This process is called transfer learning, making models that have been previously trained to be able to recognize new objects without having to create a model from scratch.

These CNN models can then be tested using mean Average Precision (mAP) [3]. For example conducts object detection and calculations on things that move, for example pedestrians, cars, bicycles, motorbikes, buses and trucks using CNN and embedded devices. The model is then tested with mean Average Precision using the captured data, either image or video and get the performance score.

So the research is to test the performance of the application of transfer learning on the CNN model using a potholes video dataset from the UAV with mean Average Precision and determine the optimal frame rate.

2. METHODOLOGY

The procedure of this research is as follows :

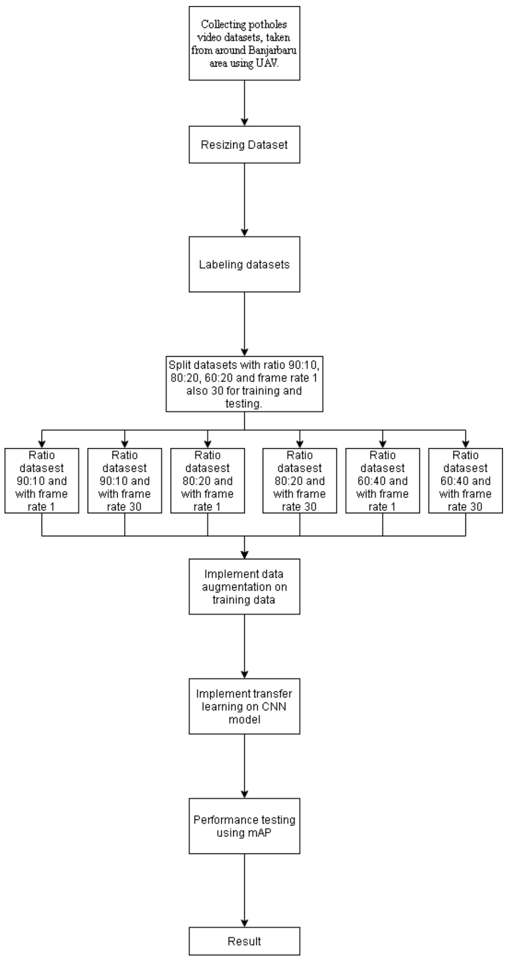

Figure 1. Research Flow

2.1 Collection of Datasets

The dataset was collected in the form of a potholes video, taken from around the Banjarbaru area using a UAV. The dataset is then resized [1]. After that the dataset is labeled 'Pothole' frame to frame and divided based on the ratio of training data and test data (90:10, 80:20, and 60:40) and frame rate (1FPS and 30FPS).

2.2 Model Making

The model is created by applying transfer learning to the pre-training model of the Tensorflow Object Detection API [5]. Furthermore, transfer learning is carried out with pothole video dataset, which is applying data augmentation before training and testing the model [1] on the dataset based on the ratio of training data and test data and its frame rate , which is 90:10 and 1FPS, 90:10 and 30FPS, 80:20 and 1FPS, 80:20 and 30FPS, 60:40 and 1FPS, 60:40 and 30FPS.

2.3 Model Testing

The test results on each data ratio and frame rate in the form of mean Average Precision as the performance value of object detection models [6, 4].

3. RESULTS AND DISCUSSION

3.1 Result

3.1.1 Collection of Datasets

The dataset used in this study was in the form of an aerial video of the Banjarbaru city road from a UAV. First, make a list of the existing potholes in the city of Banjarbaru, by surveying the location directly and saving the coordinates. The coordinates are shown in table 1.

Table 1 Pothole coordinates

No.	Location
1	-3.501994, 114.838416
2	-3.500725, 114.793675
3	-3.499039, 114.797023
4	-3.498997, 114.796965
5	-3.495192, 114.850902
6	-3.476789, 114.858246
7	-3.473052, 114.852233
8	-3.459165, 114.827042
9	-3.458692, 114.846099
10	-3.455179, 114.841893
11	-3.454330, 114.834918
12	-3.452528, 114.833838
13	-3.452018, 114.831272
14	-3.451997, 114.831419
15	-3.451988, 114.831447
16	-3.451875, 114.839298
17	-3.451780, 114.837237
18	-3.451767, 114.837422
19	-3.451755, 114.837281
20	-3.451079, 114.837998
21	-3.450346, 114.838629
22	-3.450323, 114.838655
23	-3.450072, 114.831150
24	-3.449592, 114.830982

25	-3.449121, 114.841276
26	-3.449083, 114.841263
27	-3.448639, 114.838868
28	-3.448133, 114.848771
29	-3.448022, 114.848747
30	-3.447997, 114.850492
31	-3.447909, 114.837641
32	-3.447906, 114.837614
33	-3.447882, 114.837546
34	-3.447749, 114.850512
35	-3.447449, 114.850517
36	-3.446434, 114.850693
37	-3.446133, 114.851349
38	-3.446115, 114.851199
39	-3.446047, 114.851116
40	-3.446027, 114.851052
41	-3.446004, 114.851218
42	-3.446001, 114.851132
43	-3.445991, 114.851082
44	-3.445988, 114.851051
45	-3.445949, 114.851045
46	-3.445932, 114.851140
47	-3.445927, 114.852288
48	-3.445913, 114.850976
49	-3.445850, 114.852244
50	-3.444671, 114.863389
51	-3.444127, 114.851444
52	-3.442551, 114.851247
53	-3.441603, 114.839861
54	-3.441543, 114.851254
55	-3.441395, 114.855808
56	-3.441162, 114.851147
57	-3.441159, 114.851389
58	-3.441121, 114.851153
59	-3.441087, 114.850959
60	-3.441029, 114.850078
61	-3.440567, 114.848428

After conducting the survey, further data were collected using the DJI Phantom 4 Pro UAV, from one point to another, the collection was carried out for approximately one hour starting at 9.30 to 10.30 in the morning. The results are as in table 2 and the capture of pothole frame from each data in figure 2.

Table 2 Results of Data Collection

Name	Location
DJI_0030	-3.448639+114.838868+12.000
DJI_0031	-3.449121+114.841276+9.700
DJI_0032	-3.451079+114.837998+8.600
DJI_0033	-3.451839+114.836787+8.800
DJI_0034	-3.448133+114.848771+9.100
DJI_0036	-3.446434+114.850693+8.800
DJI_0039	-3.445988+114.851051+12.700
DJI_0040	-3.446133+114.851349+8.700
DJI_0041	-3.445927+114.852288+8.500
DJI_0042	-3.442551+114.851247+8.600
DJI_0044	-3.441159+114.851389+8.800
DJI_0045	-3.441087+114.850959+8.800

Figure 2. Pothole capture from each video data

All collected data is resized from 1280 pixels by 720 pixels to 720 pixels by 480 pixels. Then frames extracted automatically based on constant frame rate using Visual Object Tagging Tool (VoTT) and given a label `LubangJalan` frame by frame. The number of frames in each frame rate can be seen in table 3, 1FPS has 73 frames and 30FPS with 1623 frames.

Table 3 Number of frames in each frame rate

Framerate Per Second (FPS)	
1	**30**
73 frames	1623 frames

After all the collected data divided based on frame rate, it is need to be divided again based on the ratio of training data and test data starting from 90:10, 80:20 and 60:40. But this division does not apply to the total number of frames at each frame rate, but rather to the number video data, this is done to prevent models from being trained and tested with the same video data, even though the used frames are different.

Of the twelve collected video data, for a ratio of 90:10 the number of training data was 10.8 rounded up to 11 and the test data was rounded up to 1, 80:20 the number of training data was 9.6 rounded up to 10 and 2.4 test data were rounded up into 2, and 60:40 the amount of training data was 7.2 rounded up to 7 and 4.8 test data was rounded up to 5. The number of training data and test data sharing based on the ratio can be seen in table 4.

Table 4 Number of training data and test data based on ratio

	Ratio		
	90:10	**80:20**	**60:40**
Training	11	10	7
Test	1	2	5

Then apply the division based on the ratio of training data and test data to the frame rate distribution. The results of dividing the dataset by one model based on ratio and frame rate can be seen in table 5 showing the number of frames in the distribution of training data and table 6 showing the number of frames in the distribution of test data.

Table 5 Training data based on ratio and frame rate division

		Framerate Per Second (FPS)	
		1	**30**
	90:10	65 frames	1420 frames
Ratio	**80:20**	62 frames	1311 frames
	60:40	49 frames	989 frames

Table 6 Test data based on the ratio and frame rate division

		Framerate Per Second (FPS)	
		1	**30**
	90:10	8 frames	203 frames
Ratio	**80:20**	11 frames	312 frames
	60:40	24 frames	634 frames

3.1.2 Model Making

Transfer learning can be applied to the Tensorflow Object Detection API pre-training model and this transfer learning does not need to be applied to all models [5], but rather, just one, the ResNet50 model [1]. This model can be seen in table 7. The application of transfer learning to selected models is carried out at Google Collaboratory (Google Colab) with hardware specifications as shown in table 8.

Table 7 Pre-trained model

No.	Nama Model
1	ssd_resnet_50_fpn_coco

Table 8 Google Colab Specifications

GPU	1xTesla K80, 12GB GDDR5 VRAM
CPU	Xeon Processors@2.3Ghz
RAM	12.6GB
DISK	320GB

Transfer learning is implemented in Google Colab. The application involves data augmentation such as horizontal flip (mirror) and random crop which is done by Tensorflow Object Detection API. After transfer learning is done, following rule in table 5 and table 6, Tensorflow Object Detection API then produce 9 new models.

3.1.3 Model Testing

While the Tensorflow Object Detection API model undergo the training, per number of steps , the new model is tested with test data and gets a test value. This test repeated until it reaches maximum step, which is 20.000 steps. The test results use the Mean Average Precision variable [4, 6] to assess the performance of the detection model.

3.1.3.1 Ssd_resnet_50_fpn_coco

3.1.3.1.1 The test results are based on a ratio of 90:10

The test results show that the best performance value of the ssd_resnet_50_fpn_coco model with ratio dataset of 90:10 is at 1FPS with a value of 25.74 mAP, followed by 30FPS with a value of 47.21 mAP as shown in table 9 and figure 3.

Table 9 Performance comparason model in ratio 90:10

Model	mAP 1FPS	mAP 30FPS
ssd_resnet_50_fpn_coco	25,74	47,21

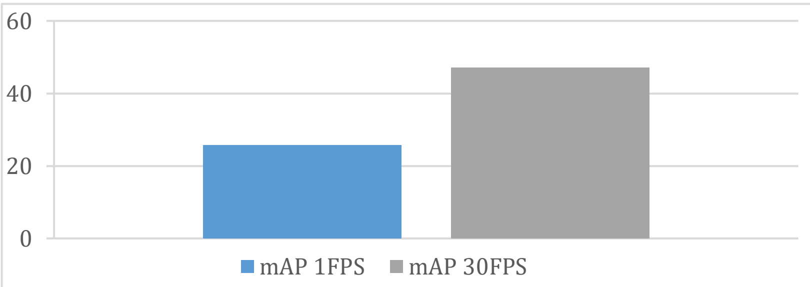

Figure 3. Performance comparison graph model in ratio 90:10

3.1.3.1.2 The test results are based on a ratio of 80:20

The test results show that the best performance value of the ssd_resnet_50_fpn_coco model with ratio dataset of 80:20 is at 1FPS with a value of 57.27 mAP, followed by 30FPS with a value of 49.28 mAP as shown in table 10 and figure 4.

Table 10 Performance comparison model in ratio 80:20

Model	mAP 1FPS	mAP 30FPS
ssd_resnet_50_fpn_coco	57,27	49,28

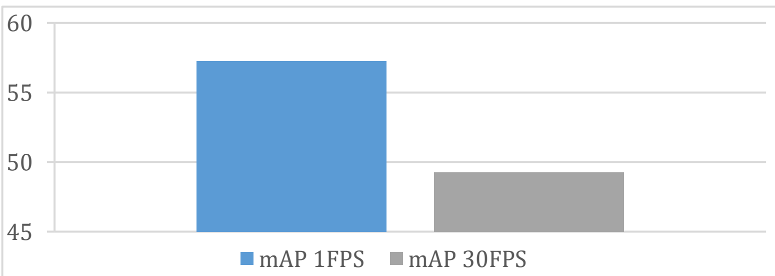

Figure 4. Performance comparison graph model in ratio 80:20

3.1.3.1.3 The test results are based on a ratio of 60:40

The test results show for the best performance values of the ssd_resnet_50_fpn_coco model with ratio dataset of 60:40 is at 1FPS with a value of

62.09 mAP, followed by 30FPS with a rate of 51.82 mAP as shown in table 11 and figure 5.

Table 11 Performance comparison model in ratio 60:40

Model	mAP 1FPS	mAP 30FPS
ssd_resnet_50_fpn_coco	62,09	51,82

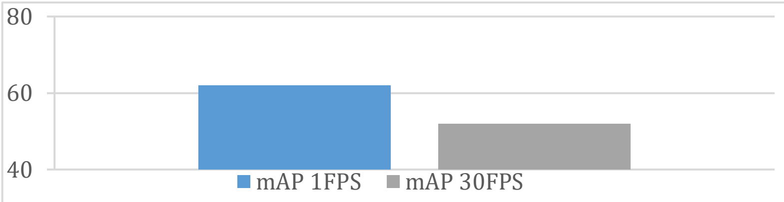

Figure 5. Performance comparison graph model in ratio 60:40

3.2 Discussion

The pothole detection model was created using aerial video dataset from UAV, taken in Banjarbaru. By conducting a survey and taking twelve data from 9:30 to 10:30 in the morning, the dataset is resized from 1280 pixels by 720 pixels to 720 pixels by 480 pixels and the dataset is labeled using VoTT from frame to frame based on a frame rate, where the result is 73 frames for 1FPS and 1623 frames for 30FPS.

After that the division is done based on the ratio with the amount of video data obtained, the results of the ratio distribution for 90:10 are 11 pieces of training data and 1 piece of test data, the ratio for 80:20 is 10 pieces of training data and 2 pieces of test data, and the ratio for 60:40 is 7 training data and 5 test data.

Then apply the division of both to get the training data and test data sharing at a ratio of 90:10, 80:20, 60:40 and a frame rate of 1FPS and 30FPS. Then apply transfer learning to Tensorflow Object Detection API model, ssd_resnet_50_fpn_coco as many as 20,000 steps and tested per number of steps using the mean Average Precision as a performance value.

Table 12 Test and training data based on ratio and frame rate

Frame Rate	Ratio	Test Data (Frame)	Training Data (Frame)
1FPS	90:10	8	65
	80:20	11	62
	60:40	24	49
30FPS	90:10	203	1420
	80:20	312	1311
	60:40	634	989

The results of the test in mean Average Precision (mAP) will later become a reference to find out the model performance on the detection of potholes and the optimal frame rate of aerial video for the detection model on pothole detection, in accordance with the research objectives described in chapter I. The highest test of the ssd_resnet_50_fpn_coco model can be seen in table 13 and figure 6.

Table 13 The highest test results of ssd_resnet_50_fpn_coco

Model	Ratio	mAP 1FPS	mAP 30FPS
	90:10	25,74	47,21
ssd_resnet_50_fpn_coco	80:20	57,27	49,28
	60:40	62,09	51,82

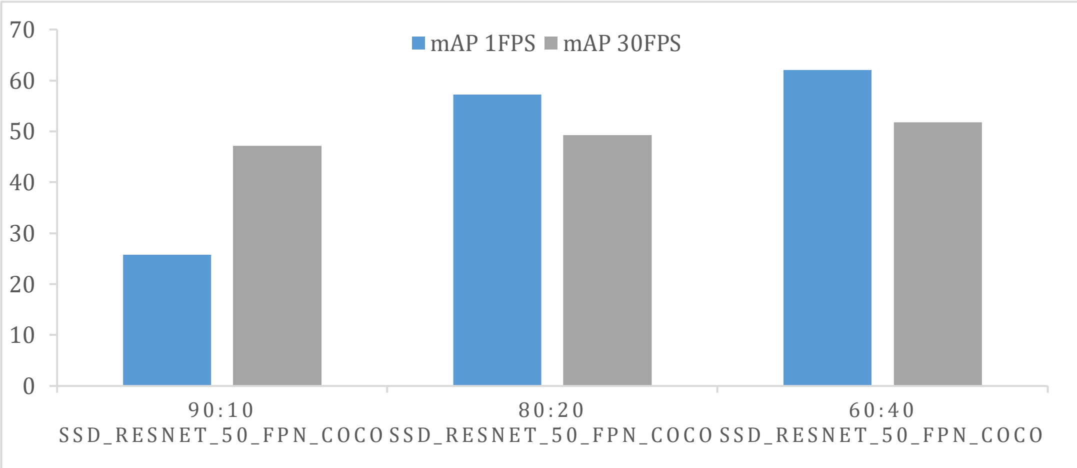

Figure 6. Performance model on each ratio and frame rate

Then the average value of the performance of the model is calculated by adding up 6 results then dividing it by the same amount. The average performance value of the model can be seen in table 14, where the value of the ssd_resnet50_fpn_coco model's performance is 48.90 mAP.

Table 14 Average model performance

Model	Rata-rata (mAP)
ssd_resnet_50_fpn_coco	48,90

And to get the optimal frame rate with the best performance, the average performance value of the model is calculated by adding up the 6 results then dividing it by the same amount. The average value of the model can be seen in table 15.

Comparison of average test results shows for optimal frame rates with the best average work values is at 30FPS with a value of 49.43 mAP, followed by 1FPS with a value of 48.36 mAP.

Table 15 Comparison of performance values with frame rates

Frame Rate	Average (mAP)
1 FPS	48,36
30 FPS	49,43

4. CONCLUSION

The results showed that from the CNN model applied by transfer learning, the ssd_resnet_50_fpn_coco model highest average performance value is 48.90 mAP. It can also be seen that the optimal frame rate highest average performance value at 30FPS with a value of 49.43 mAP followed by 30FPS with a value of 48.36 mAP.

REFERENCES

[1] Akula, Aparna & Bhatia, Yukti & Rai, Rachna & Gupta, Varun & Aggarwal, Naveen & Aukla, Aparna. 2019. "***Convolutional Neural Network*** **Based Potholes Detection Using Thermal Imaging**". Journal of King Saud University - Computer and Information Sciences.

[2] Arrofiqoh, Erlyna & Harintaka, Harintaka. 2018. "**Implementasi Metode** ***Convolutional Neural Network*** **Untuk Klasifikasi Tanaman Pada Citra Resolusi Tinggi**". Geomatika. 24. 61.

[3] Heredia, A. & Barros-Gavilanes, G. 2019. "**Video processing inside embedded devices using SSD-Mobilenet to count mobility actors**". 2019 IEEE Colombian Conference on Applications in Computational Intelligence (ColCACI), 1-6.

[4] Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. 2017. "**MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications**".

[5] Karlsson, P., & Johansson, E. 2018. "**Object Identifier System for Autonomous UAV : A subsystem providing methods for detecting and descending to an object. The object is located in a specified area with a coverage algorithm**". (Dissertation).

[6] Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & Berg, Alexander. 2016. "**SSD: Single Shot MultiBox Detector**".

[7] Petkova, Mia. 2016. "**Deploying Drones For Autonomous Detection of Pavement Distress**".

[8] Suryowinoto, A. 2017. "**Penggunaan Pengelohan Citra Digital Dengan Algoritma** ***edge detection*** **Dalam Mengidentifikasi Kerusakan Kontur Jalan**".