ZANCO Journal of Pure and Applied Sciences
The official scientific journal of Salahaddin University-Erbil
https://zancojournals.su.edu.krd/index.php/JPAS

ISSN (print):2218-0230**, ISSN (online):** 2412-3986**, DOI: http://dx.doi.org/10.21271/zjpas**

RESEARCH PAPER

Road Pothole Detection Using Unmanned Aerial Vehicle Imagery and Deep Learning Technique

Dana Mohammed Ali[1], Haval A.Sadeq[1]

[1]Department of Geomatics (Surveying), College of Engineering, Salahaddin University- Erbil, Kurdistan Region, Iraq

A B S T R A C T:

Potholes are considered the main factor for road defects, which leads to road status deterioration, which, consequently will lead to increased road accidents. The first step in road maintenance is to inspect the road surface and then accurately detect potholes. However, manually identifying them is costly and time-consuming. In this study, unmanned aerial vehicle (UAV) imagery was used to create orthophotos of the roads and, using deep learning methods, potholes were detected. The used deep learning method in this study is the "you only look once" (YOLO) algorithm. YOLO is one of the "deep learning-based approaches" to detecting objects and is a single-stage network which requires only one forward propagation across the neural network and focuses on the entire image. The fourth version of YOLO is YOLOV4, which has two different architectures (YOLOv4 and YOLOv4-tiny). Two roads were chosen as the study areas, and to generate the orthophotos of the roads, UAV was used to acquire images. To train both methods in the process of detecting potholes using deep learning, 5300 images were used, 90% used for training and 10% applied for testing. The two used architectures were trained for 6000 iterations. Both methods were evaluated based on the average loss, mean average precision (mAP), and training and testing time. The results showed that the (mAP) values for YOLOv4 and YOLOv4-tiny were 91. 2% and 85.7%, respectively. At the end of the 6000 iterations, the average loss for YOLOv4 is 0.30% and for YOLOv4-tiny is 0.34%. In the training process, YOLOv4 needs 29 seconds for each iteration, while YOLOv4-tiny requires only 8 seconds. In the test process, YOLOv4-tiny is faster at detecting potholes than YOLOv4. The approaches were tested on orthophotos created by processing UAV photos. When comparing the detection of both architectures with visual detection, the results showed that YOLOv4 was able to detect most of the potholes on roads, but YOLOv4-tiny detected a lower number of potholes.

KEY WORDS: UAV, Deep Learning, YOLO, Pothole Detection
DOI: http://dx.doi.org/10.21271/ZJPAS.34.6.12
ZJPAS (2022) , 34(6);107-115 .

1. INTRODUCTION:

For the economic growth of a country, road systems are an important infrastructure that provides transportation services. Road maintenance is putting pressure on the budget of the country when the major portion of the road infrastructure reaches the end of its useful life (Salini et al., 2017). Monitoring the road is the main important work that should be done before repairing the road. The road's condition is influenced by a variety of factors, such as overloaded big vehicles, terrible weather, etc.

In the transportation network, the most popular form of road surface is asphalt road pavement. During the usage of the road, some sort of distress will always emerge on the pavement (Koch et al., 2015). The most prevalent types of road surface degradation are potholes and cracks. They have a substantial impact on the vehicle's performance (Tedeschi and Benedetto, 2017). Before road repair and reconstruction, the road department should conduct a road condition inspection. However, due to the fast increase in kilometers of the road networks, particularly highways, it is currently a difficult task for the road management department to inspect the condition of roads sufficiently (Pan et al., 2018).

*** Corresponding Author:**
Dana Mohammed Ali
E-mail: dana.ali@kissr.edu.krd
Article History:
Received: 28/03/2022
Accepted: 10/08/2022
Published: 20/12 /2022

UAV systems have become a helpful tool for academics and practitioners dealing with challenges in a variety of sectors in terms of quick decision-making, and maintenance (Zeybek et al., 2020). Transportation engineering is one of the fields that has just begun to employ UAV technology. Road maintenance and repair are highly serious tasks, and comprehensive data collection about the road's state is required to determine whether or not these actions are required (Tan and Li, 2019).

Traditional manual road inspection is time-consuming, labor-intensive, and subjective (Attoh-Okine and Adarkwa, 2013). Some automated techniques, such as various types of road survey vehicles equipped with stereo cameras, light detection and ranging (LiDAR) technology, laser profilers, and so on, have also been developed and deployed in road surveys, with the potential to greatly improve the survey's efficiency and objectivity (Choi et al., 2016, Bar Hillel et al., 2014, Laurent et al., 2012). However, because just the vehicle's footprint is inspected in a single trip, numerous visits may be required to cover the whole width of the road by the vehicle. Furthermore, the survey procedure has an impact on traffic flow, which is particularly undesirable for high-traffic roads (Kim et al., 2014). UAVs offer a lot of flexibility, are very inexpensive compared to survey vehicles, are easy to handle, and need less field effort, so they're quite hopeful for pavement condition monitoring (Chen et al., 2014).

With the advancement of science and technology, as well as the popularity of the deep learning model in the engineering area, powerful and low-cost intelligent systems can be employed to identify potholes instead of workers (Patra et al., 2021). Thanks to advanced technology, which leads to a decrease in the time and cost required for detecting potholes, furthermore the potholes will be identified with more precision.

You only look once (YOLO) is one of the "deep learning-based approaches" to detecting objects. To detect objects faster, Redmon et al. (Redmon et al., 2015) created the YOLO object identification algorithm, which requires only one forward propagation across the neural network. YOLO focuses on the entire image, generating predictions in the global context, unlike other conventional classifiers that learn about particular portions of the image to generate models (Zhao and Li, 2020). It splits the picture into a grid of S × S cells, with each grid cell predicting B bounding boxes based on the object's center (x, y), with dimensions (w, h), conditional class probabilities C, and related confidence score values. The probability of the bounding box containing the object and the precision of the boundary box combine to provide the confidence score. Box confidence score and conditional class likelihood are used to get the class confidence score (Du et al., 2021).

The main aim of this study is to automatically detect potholes on roads using deep learning methods when the orthophotos of the roads are created using UAV images. The deep learning algorithm used in this study is the fourth version of YOLO. YOLOv4 has two architectures: YOLOv4 and YOLOv4-tiny. Both methods were trained for 6000 iterations. The models were compared based on mAP, average loss, and time consumption for training and testing. The UAV images were used to create orthophotos of the two study areas, which have many potholes on the roads. To evaluate which method can detect potholes accurately, the obtained last weights of both methods were tested on both study areas. Then the potholes are detected manually using visual inspection and the results are compared with those of automatic detection.

2. MATERIALS AND METHODS

A comparison between the two different architectures of YOLOv4 (YOLOv4, and YOLOv4-tiny) was conducted based on the average loss, training and testing time, and mAP. In section three, which shows the results of the study, average loss, mAP, and training and testing time, will be explained clearly with their mathematical formula. The results in detecting potholes were compared based on deep learning methods with manual detection technique.

2.1. Study area

In this study, two roads were used as the study areas. Both roads are located in Sulaimaniah province. The first road connects Tanjero and Glazarda villages, the positional center of the study area (45° 25′ 27″ E, 35° 27′ 24″ N). The second study area is a part of a road that connects Sulaymaniah and Qaradaq cities with the position center (45° 23′ 40″ E, 35° 20′ 36″ N).

2.2. UAV Specifications

In this study, a DJI MAVIC 2 Pro was employed to explore the chosen roads. It's a light platform, weighting just 907 grams, with a flying duration of roughly 30 minutes. It has a navigation system connected with GPS and GLONASS positioning systems. The UAV camera has a 20-megapixel CMOS sensor with an image size of 5472 x 3648 pixels (DJI MAVIC 2 Pro MANUAL).

2.3. Data Acquisition

Data acquisition was planned using Pix4Dcapture software. By adding the critical flight parameters such as area extent, image overlaps, flight altitude, and velocity, Pix4Dcapture lets the user undertake a completely autonomous flying mission. The UAV is equipped with an RGB camera with a resolution of 20 megapixels. In this study, oblique pictures were acquired by altering the camera angle to 70 degrees in order to increase image alignment accuracy and gather a comprehensive point cloud. The UAV was configured to fly at low altitude, the front and side overlaps were 80 percent, and the drone's speed was set to medium. In the first study area, the altitude of the UAV was 19 meters, and it acquired 97 images, while in the second study area, the altitude was decreased to 10 meters, and it captured 61 images.

2.4. 3D Reconstruction Workflow

To generate the orthophotos, Agisoft Metashape software was used. The software is capable of reconstructing three-dimensional images. The Agisoft Metashape uses structure from motion (SfM) algorithm to generate the orthophotos. SfM photogrammetry is a method of constructing a three-dimensional structure using two-dimensional images. Photographs are stitched together using photogrammetry software to make a three-dimensional (3D) model and other product like orthophoto. Using SfM, internal and external camera geometry can be determined without the need for a pre-defined set of ground control points. To generate the orthophoto, the software should perform some steps such as feature detection and aligning photos, building sparse point clouds, building dense clouds, mesh model, generating DSM, and finally generating the orthophoto (Wu, 2013).

2.5. Dataset

A custom dataset is produced for the model to train on. To create the custom dataset, different sources were used to acquire pothole images, such as downloading from the internet, smartphone camera, and UAV camera. The pavement pothole dataset utilized in this study is comprised of around 5300 images of various sizes, with multiple potholes in each photo. Potholes of all sizes and shapes, both wet and dry, are seen in the images. The images in the dataset were downloaded from Google Images and Kaggle. UAV was used to capture all of the images downloaded from Google Images and Kaggle. Kaggle is an online community platform for data scientists and machine learning enthusiasts. Kaggle allows users to collaborate with other users, find and publish datasets. Some images in the dataset have been captured using a smartphone camera. In addition, some images have been acquired using UAV for areas that are difficult to photograph with smartphone cameras, such as highways.

2.6. YOLOv4

YOLOv4 (Bochkovskiy et al., 2020) is an updated version of YOLOv3(Chitale et al., 2020) with a few models' architectural changes and increased overall performance. The feature pyramid network (FPN) structure is maintained as the training approach in YOLOv4. The backbone has been replaced with CSPDarkNet-53. When compared to YOLOv3, Spatial Pyramid Pooling (SPP) and a Path Aggregation Network (PAN) are introduced, which employ multiscale feature concatenation to improve picture representation learning. The SPP structure enhances the receptive field by pooling on four scales to isolate the most critical context feature, whereas PANet employs up-sampling and down-sampling to extract features repeatedlyYOLOv4 consists of 162 layers, and the final three layers are detected layers (Huang et al., 2020).

2.7. YOLOv4-tiny

The YOLOv4-tiny approach is based on the YOLOv4 method, but it is supposed to be faster at object detection. YOLOv4-tiny uses the same backbone as YOLOv4. It considerably improves the likelihood of using object detection on embedded systems or mobile devices (Silva et al., 2020).

YOLOv4 and YOLOv4-Tiny divide the input image into S x S cells. The convolutional layers of both methods downscale the picture by a factor of 32, so if the input image is 416 x 416 pixels, it generates a 13 x 13 output feature map. Every cell generates three predicted bounding boxes, and each bounding box has five values and another value related to class number. The values are (pc, px, py, pw, ph, and c) if the number of classes to be detected is one. If the number of classes is greater than one, the values of the bounding box change with the number of classes. The below example illustrates this case.

$$Y = \begin{bmatrix} pc \\ bx \\ by \\ bw \\ bh \\ c0 \\ c1 \\ c2 \\ c3 \\ . \\ . \\ . \\ cn \end{bmatrix}$$

Number of filters = (5 + c) x 3

Pc: probability of the class

Px: predicted X coordinate

Py: predicted Y coordinate

Pw: the width of the predicted bounding box

Ph: the height of the predicted bounding box

c: class

In this study, the input image is 416 X 416 X 3, so the output image in the convolutional layer is 13 X 13 X 18 by using the following formula. The study is related to detecting one class, and the class is pothole.

$$S \times S \times [(5 \times c) * 3]$$

2.8. Annotation and Labeling

The technique of indicating potholes in images is called annotation and labeling. Create a bounding box manually around the pothole's outside edge and then label it. The images are annotated according to the YOLO criteria using an open-source program called "LabelImg" software, which is available at https://sourceforge.net/projects/labelimg.mirror. All the research data was annotated and labeled. In the annotation and labeling procedure, the id-class, bounding box center (x and y), and width and height (w and h) of the bounding box are all saved. The bounding box information is in decimal format from 0–1 scale, and the id class is an integer number starting from 0. Each jpg image will be accompanied by a txt file including pothole information.

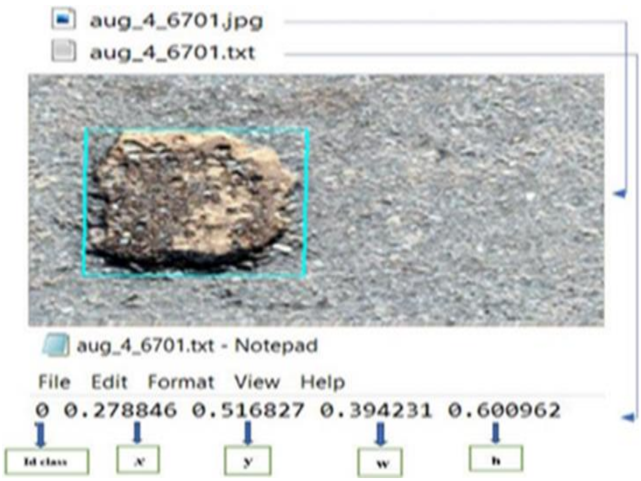

Figure 1: Annotation and Labeling

As shown in figure 1, the id class is zero because the study involves detecting only one class. The x value is the ratio between the distance from the upper left of the image toward the right direction to the center of the bounding box and the width of the image. The y value is the ratio between the distance from the upper left of the image downward to the center of the bounding image box and the height of the image. The values of w and h are the ratios between the width and height of the bounding box and the width and height of the image, respectively.

2.9. Training

The DarkNet network serves as a model training environment. This neural network structure is written in C and CUDA and may be directly performed on the GPU (Bochkovskiy et al., 2020). DarkNet is fast, easy to install, and supports CPU and GPU computation. It is installed differently depending on the GPU; the processing was performed using a laptop with the Windows 10 Professional 64-bit operating system, a 2.9 GHz Intel processor, 16 GB of RAM, and a GPU of NVIDIA GEOFORCE 930MX with 2 GB of RAM. The parameter settings are: learning rate configuration is 0.001. The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the learning rate is challenging as a value too small (0.0001) may result in a long training process that could get stuck, whereas a value too large (0.01) may result in learning a sub-optimal set of weights too fast or an unstable training process. The batch size is equal to 64 with 16 subdivisions. The photos in the dataset were split into two groups: 90% for training and 10% for testing the model. A total of 6000 iterations were performed to train the model. The models were saved after each one hundred iterations and analyzed to see how well they performed. This savings is good when the training process is ending for any reason. To continue the training process, it can be started at the stopped iteration. The model was stored every 1000 iterations during the training process. The best and most recent weights were also saved in the training folder. The final result analysis is based on the 6000 iterations as the last weight. The detection accuracy in 6000 iterations is better than in 5000 iterations, and the result of 5000 iterations is better than that of 4000 iterations and so on. If the YOLO algorithm trains for more than 6000 iterations, the results will be a little better, but not a big change will occur.

3. RESULTS

In this section, the results that have been obtained from the training and testing processes are listed.

3.1. Average Loss

Loss is a significant outcome of the pothole modeling process. It is an index to evaluate the accuracy. It is used to lead the training iteration. The training will be terminated if there are no changes in the loss value. Every modeling process produces its own loss. If a method produces a low loss, it is determined that the method has good performance in detecting objects. As shown in figure 2, the two losses are combined to investigate how the modeling process performs differently. When compared to the YOLOv4 architecture, it can be seen that YOLOv4-tiny produces a larger loss. The loss for the two architectures is relatively high at the start of iteration, but after 3400 iterations, the loss of YOLOv4 is less than 0.5, whereas YOLOv4-tiny produced an average loss of 0.5 at iteration 4000. At the end of the training process, the loss of YOLOv4 was less than 0.3, while YOLOv4-tiny produced an average loss of around 0.4. Loss value has not any unit. The YOLO algorithm uses the following formula to compute average loss:

$$average\ loss = loss\ (coord) + loss(conf) + loss(class)$$

$$loss(coord) = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} W_{ij}^{obj} (2 - w_i * h_i)\ [L_{CIOU}]$$

$$loss(conf) = -\sum_{i=0}^{S^2} \sum_{j=0}^{B} W_{ij}^{obj} [\hat{c}_i^j \log(C_i^j) + (1 - \hat{c}_i^j) log(1 - C_i^j)] \\ - \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} (1 - W_{ij}^{obj})\ [\hat{c}_i^j log(C_i^j) + (1 \\ - \hat{c}_i^j) log(1 - C_i^j)]$$

$$loss(class) = -\sum_{i}^{S^2} \sum_{j}^{B} W_{ij}^{obj} \sum_{c=1}^{C} \left[\hat{p}_i^j(C) log\left(p_i^j(C)\right) - \left(1 - \hat{p}_i^j(C)\right) \\ - log\left(1 - \hat{p}_i^j(C)\right)\right]$$

Where, S^2 is the number of grids in input image, B is the number of bounding boxes in a grid,
$obj\ W_{ij}^{obj}$ is merely a function of the object. If the *jth* bounding box of the *ith* grid is responsible for detecting the current object, $W_{ij}^{obj} = 1$, otherwise $W_{ij}^{obj} = 0$. The C_i^j and $\hat{c}_i^j$ are the confidence score of predicted box and confidence score of truth box, respectively. λ_{noobj} is a weight parameter. $p_i^j(C)$ and $\hat{p}_i^j(C)$ are predict probability and truth probability to which the

object belongs to *c* classification in the *jth* bounding box of the *ith* grid.

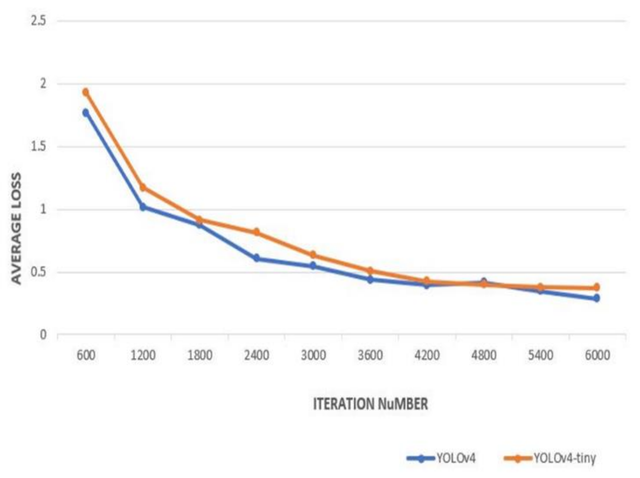

Figure 2: Loss of YOLOv4 and YOLOv4-tiny in the training process

3.2. Metric Evaluation

For the purpose of the evaluation in detecting potholes on the test images from the dataset, two indices have been used: Precision and Recall. Precision is the relationship between true positives (TPs) and true positives (TPs) along with false positives (FPs); recall, which is the probability that potholes could be detected as positive; and the relationship between the TPs and the TPs together with the false negatives (FNs). These two indices can be used to evaluate the detection accuracy of any algorithm. In the detection process, the (TP, FP, and FN) are produced. Precision uses (TP and FP) while recall uses (TP and FN).

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Where:

TP- objects that are actually potholes and predicted as potholes.

FP- objects that are actually not potholes but predicted as potholes.

FN- objects that are actually potholes but predicted as not potholes.

The results of the accuracy analysis of each method are depicted in Table 1. In the testing process, each image was investigated and calculated (TP, FP, and FN) individually. According to our tests, YOLOv4 achieved precision and recall of 0.95 and 0.98, respectively, while YOLOv4-tiny delivered 0.92 precision and a recall of 0.94.

Table 1: *Detection accuracy results for the applied testing data*

Architecture	Test data	TP	FP	FN	Precision	Recall	mAP
YOLOv4	500	532	27	13	0.95	0.98	91.2%
YOLOv4-tiny	500	498	43	31	0.92	0.94	85.7%

3.3. Time for training and testing

When using a model for detecting objects in the deep learning method, the speed of the training and testing process is an important factor that should be considered. In this study, YOLOv4 required more time in the training and testing process than YOLOv4-tiny. In the training process, YOLOv4 needs 29 seconds for each iteration, while YOLOv4-tiny requires only 8 seconds. In the test process, YOLOv4-tiny is faster than YOLOv4 and needs an average detection time of less than 500 milliseconds per image, while YOLOv4 needs approximately 900 milliseconds per image.

3.4. mAP

Mean Average Precision (mAP) is a metric used to evaluate object detection. Most of the deep learning algorithms use mAP to evaluate their performance in detecting objects. YOLO is one of the models that uses mAP. The mAP value is commonly used to determine the object detection accuracy. In order to obtain a high mAP value in the training process, the number and size of images should be increased. The number of iterations should also be increased. In the training process, mAP is shown continuously approximately every 400–600 iterations, and the progress can be seen in the detection model. At the iteration of 600, YOLOv4-tiny achieved a higher mAP than YOLOv4, but in the next thousand iterations, the mAP value of YOLOv4 increased significantly. At the end of 6000 iterations, the mAP value of YOLOv4 was higher than the mAP value of YOLOv4-tiny by around six percent. When a deep learning algorithm calculates mAP, it uses the same formula of precision, but its computations are based on different threshold ranges from 50 to 95. But precision is calculated based on a particular threshold value. The mAP values between the two used architectures are illustrated in figure.3.

$$mAP_{50:95} = \frac{1}{10}(AP_{50} + AP_{55} + AP_{60} + \ldots \ldots \ldots . + AP_{90} + AP_{95})$$

The AP is calculated using different thresholds example: 0.50, 0.55, 0.6, ……, 0.95.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

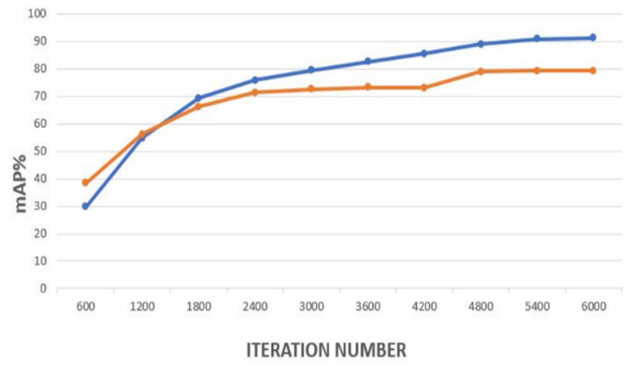

Figure 3: mAP of YOLOv4 and YOLOv4-tiny in the training process

3.5. Pothole detection accuracy

Measuring the detection accuracy of potholes is an important index in deep learning. For this purpose, potholes have been manually and visually inspected on the roads. The obtained last-weights, which were obtained in the 6000 iterations of YOLOv4 and YOLOv4-tiny, were applied to the orthophotos. Every iteration provides an updated weight, and the last weight is obtained in the last iteration. In this study, the last weight was obtained after 6000 iterations.

Manual detection is the process of detecting potholes manually by using just our eyes to know where the potholes have appeared on the road. In both study areas, the manual detection was done and the number of potholes was counted for each study area separately. The process of manual detection was tedious, dangerous, and time-consuming, but the results were more accurate. Through comparing the detection abilities of deep learning with manual detection, the results demonstrated that YOLOv4 can achieve very good accuracy in detecting potholes. As shown in figures (4 and 5), YOLOv4 has detected most potholes on the roads. In the first study area, as shown in figure 4, there were six potholes on the road, and YOLOv4 detected six potholes too. Figure 5, shows the second study area, where the road has ten potholes, but YOLOv4 has detected ten potholes and two false positions as potholes. This means that YOLO4 has the ability to detect every pothole but has also produced some errors. The detected potholes have confidence values ranging from 90 to 99 percent. When applied to the orthophotos, the results show that YOLOv4-tiny achieved lower accuracy in detecting potholes. As illustrated in figures (6 and 7), YOLOv4-tiny hasn't detected every pothole on the roads. As shown in figure 6, the road in the first study area has six potholes, but YOLOv4-tiny detected seven potholes, which shows that the method detected a wrong place as pothole. Also, the confidence scores of the detected potholes are too low, ranging from 34 to 99 percent. Figure 7, shows the second study area. There are ten potholes on the road, but YOLOv4-tiny has detected only seven potholes with high confidence values.

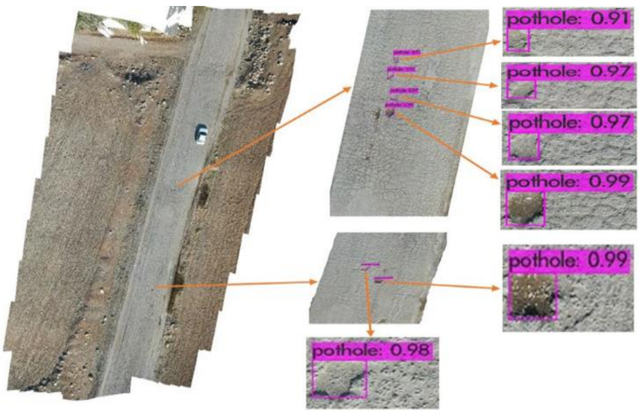

Figure 4: Applied YOLOv4 to the first study area's orthophoto

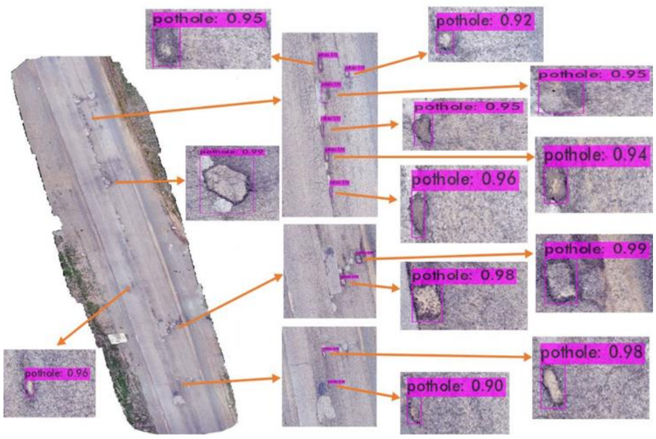

Figure 5: Applied YOLOv4 to the second study area's orthophoto

4. DISCUSSION

At the end of the training process, YOLOv4 produced a lower loss than YOLOv4-tiny. YOLOv4 consists of 162 layers, while YOLOv4-tiny has only 38 layers. YOLOv4 needs more time and uses more layers in the training process, so it produces a lower average loss. YOLOv4-tiny is fast and has a small number of layers, so it produces a little more average loss than YOLOv4. The difference in the produced loss of both methods was not significant in this study. The number of images in the dataset was too large; it affected the loss function. So, both methods can detect potholes accurately and, in the detection process, they have no big difference. Also, the precision and recall values of both methods were nearly equal and had no significant difference. YOLOv4 produced a higher mAP than YOLOv4-tiny because YOLOv4 has more layers than YOLOv4-tiny.

When using a model for detecting objects in the deep learning method, the speed of the training and testing process is an important factor that should be considered. In deep learning, the required time for the training and testing process depends on the image size in the dataset and the device being used. If a computer has a good graphic processing unit (GPU), it needs less time, and vice versa. The used algorithm and the number of layers also have an effect on the training and testing time. YOLOv4 has a complex architecture and needs more time, but can achieve high accuracy. YOLOv4-tiny is simple and needs less time, but it can achieve lower accuracy. In our experiment, YOLOv4 consists of 162 layers, so it needs more time for training and testing. YOLOv4-tiny has only 38 layers, so it requires less time in the training and testing processes.

When YOLOv4 was used to detect potholes on the roads, the results can be seen that YOLOv4 detected every pothole on the road in the first study area and most potholes in the second study area. That means the using of YOLOv4 for road pothole detection is the best option. YOLOv4's confidence scores in detecting potholes are high. This indicates that the results have more certainty that potholes have been appropriately identified. When YOLOv4-tiny was tested to detect potholes in the first study area, it detected potholes with a low confidence value and detected a place as a pothole, but in reality, the place was not a pothole. In the second study area, the road has ten potholes, but YOLOv4-Tiny detected only seven potholes and missed some potholes on the road. So, YOLOv4 is preferred over YOLOv4-tiny because of its high accuracy in detecting potholes.

5. CONCLUSIONS

Road maintenance is a difficult task that is also a significant issue throughout the world. The identification of road abnormalities such as potholes is one of the most important road monitoring and maintenance tasks. To avoid road accidents, it is essential to identify potholes accurately. Manual pothole detection is costly and time-consuming. Using deep learning to detect potholes is a new and fast method. The results show that the YOLOv4 and YOLOv4-tiny are able to obtain mAP values of 91.2% and 85.7%, respectively. The achieved accuracy of YOLOv4 architecture to detect potholes on the asphalt road surface is excellent, but YOLOv4-tiny achieved lower accuracy in detecting potholes. YOLOv4 can detect potholes with a higher confidence score than YOLOv4-tiny. The results of this study can be performed on asphalt roads because the used images in the training process belong to asphalt roads. In conclusion, it is possible to use UAV images to generate an orthophoto of any road and detect potholes on it successfully using deep learning, which is considered to be safer, cost-effective, and faster than the traditional manual method.

Acknowledgements

The authors would like to thank the Kurdistan Institution for Strategic Studies and

Scientific Research for granting permission to use their UAV for the research purposes.

REFERENCES

ATTOH-OKINE, N. & ADARKWA, O. J. D. C. F. T., UNIVERSITY OF DELAWARE: NEWARK, DE, USA 2013. Pavement condition surveys–overview of current practices.

BAR HILLEL, A., LERNER, R., LEVI, D., RAZ, G. J. M. V. & APPLICATIONS 2014. Recent progress in road and lane detection: a survey. 25**,** 727-745.

BOCHKOVSKIY, A., WANG, C.-Y. & LIAO, H.-Y. M. J. A. P. A. 2020. Yolov4: Optimal speed and accuracy of object detection.

CHEN, A. Y., HUANG, Y.-N., HAN, J.-Y. & KANG, S.-C. J. J. S. S. S. 2014. A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering. 13**,** 1065-1094.

CHITALE, P. A., KEKRE, K. Y., SHENAI, H. R., KARANI, R. & GALA, J. P. Pothole detection and dimension estimation system using deep learning (yolo) and image processing. 2020 35th International Conference on Image and Vision Computing New Zealand (IVCNZ), 2020. IEEE, 1-6.

CHOI, J., ZHU, L., KUROSU, H. J. I. A. O. T. P., REMOTE SENSING & SCIENCES, S. I. 2016. Detection of cracks in paved road surface using laser scan image data. 41.

DU, Y., PAN, N., XU, Z., DENG, F., SHEN, Y. & KANG, H. J. I. J. O. P. E. 2021. Pavement distress detection and classification based on YOLO network. 22**,** 1659-1672.

HUANG, Z., WANG, J., FU, X., YU, T., GUO, Y. & WANG, R. J. I. S. 2020. DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection. 522**,** 241-258.

KIM, T., RYU, S.-K. J. J. O. E. T. I. C. & SCIENCES, I. 2014. Review and analysis of pothole detection methods. 5**,** 603-608.

KOCH, C., GEORGIEVA, K., KASIREDDY, V., AKINCI, B. & FIEGUTH, P. J. A. E. I. 2015. A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. 29**,** 196-210.

LAURENT, J., HÉBERT, J. F., LEFEBVRE, D. & SAVARD, Y. Using 3D laser profiling sensors for the automated measurement of road surface conditions. 7th RILEM international conference on cracking in pavements, 2012. Springer, 157-167.

PAN, Y., ZHANG, X., CERVONE, G., YANG, L. J. I. J. O. S. T. I. A. E. O. & SENSING, R. 2018. Detection of asphalt pavement potholes and cracks based on the unmanned aerial vehicle multispectral imagery. 11**,** 3701-3712.

PATRA, S., MIDDYA, A. I., ROY, S. J. M. T. & APPLICATIONS 2021. PotSpot: Participatory sensing based monitoring system for pothole detection using deep learning. 80**,** 25171-25195.

REDMON, J., DIVVALA, S., GIRSHICK, R. & FARHADI, A. J. A. P. A. 2015. You look only once: unified real-time object detection.

SALINI, R., XU, B. & PAPLAUSKAS, P. J. S. O.-T. C. E. J. 2017. Pavement distress detection with picucha methodology for area-scan cameras and dark images. 34-45.

SILVA, L. A., SANCHEZ SAN BLAS, H., PERAL GARCÍA, D., SALES MENDES, A. & VILLARUBIA GONZÁLEZ, G. J. S. 2020. An architectural multi-agent system for a pavement monitoring system with pothole recognition in UAV images. 20**,** 6205.

TAN, Y. & LI, Y. J. I. I. J. O. G.-I. 2019. UAV photogrammetry-based 3D road distress detection. 8**,** 409.

TEDESCHI, A. & BENEDETTO, F. J. A. E. I. 2017. A real-time automatic pavement crack and pothole recognition system for mobile Android-based devices. 32**,** 11-25.

WU, C. Towards linear-time incremental structure from motion. 2013 International Conference on 3D Vision-3DV 2013, 2013. IEEE, 127-134.

ZEYBEK, M., BIÇICI, S. J. T. J. O. R. S. & GIS 2020. Road distress measurements using UAV. 1**,** 13-23.

ZHAO, L. & LI, S. J. E. 2020. Object detection algorithm based on improved YOLOv3. 9**,** 537.

https://dl.djicdn.com/downloads/Mavic_2/Mavic+2+Pro+Zoom+User+Manual+V1.4.pdf