Low Latency and Non-Intrusive Accurate Object Detection in Forests

Ambreen Hussain*, Bidushi Barua, Ahmed Osman, Raouf Abozariba, and A. Taufiq Asyhari*
School of Computing and Digital Technology, Birmingham City University, UK
E-mail: {ambreen.hussain, bidushi.barua, ahmed.osman, raouf.abozariba, taufiq.asyhari}@bcu.ac.uk

***Abstract*—A resilient and healthy forest requires accurate and timely monitoring, observing key forest health indicators (FHI). Forest managers and rangers usually perform tedious manual data collection using citizen science for biodiversity conservation and ecological research. With the advent of faster radio network technologies such as 4G, it is advantageous to leverage these networks' high speed and low latency for real-time monitoring. We present a novel approach to stream high definition videos over cellular networks to provide real-time ($<$ 0.5 seconds) data transmission to the YOLOv5 machine learning algorithm, hosted in the cloud. The system provides non-intrusive precise tree class detection, matching existing models such as Fast R-CNN and SSD. Our investigation also reveals that the same accuracy can be achieved with 99% fewer iterations, minimizing computational time and cost.**

***Index Terms*—YOLOv5, 4G/5G, WebRTC, tree species detection, low-latency**

I. Introduction

A healthy forest provides a variety of services and products ranging from energy, building materials, and hosting biodiversity to regulate climate. The forest health (FH) around the world is declining due to air pollution, climate change, and increased human activities. However, the resilience and efficiency of the forest health (FH) is directly linked to the richness of tree species [1]. To support forest health monitoring and biodiversity conservation, forest ecological research and forest management observe many key forest health indicators (FHI) such as tree species identification, tree girth and height measurement, deadwood, wildlife signs, soil moisture, precipitation, temperature and fire risk [2]. FHI monitoring is often performed by forest managers, forest administrators and other public entities at local and national scales. Numerous on-site monitoring programs at local and regional levels utilize citizen science to record standard FHI. These programs perform on-site assessments, where a collection of FHIs are directly observed following a user guide [3]. For domain experts, manual monitoring may be a resource-intensive task, and for non-experts, it is error-prone and time-consuming [4]. Moreover, the results from manual observations are less accurate due to the unavailability of expert knowledge in many cases. Accelerating the monitoring tasks and making it accessible to non-experts reduces cost and effort [5]. A possible solution to speed up forest monitoring is to leverage computer vision and machine learning techniques such as image classification, object detection and image segmentation. Various machine learning algorithms have been explored in the research studies. Most recently, deep learning has been utilized in processing high spatial resolution imagery [6]. The most effective and popularly used methods are convolutional neural networks (CNN). Object detection is an important research direction for complex computer vision tasks such as target detection, target tracking, and event detection [7].

Capturing information from a distance is an excellent technique for non-intrusive forest monitoring to prevent tree bark damage and preserve wildlife. Remarkable progress in remote sensing applications is facilitated through the advancement of mobile phones, Unmanned Aerial Vehicles (UAVs), sensing devices, Geographical Information Systems (GIS), Global Navigation Systems (GNS), and evolving fifth-generation (5G) networks [8]. According to GSMA, 5G connections are expected to grow to 1.8 billion by 2025 [9]. Contemporary cellular networks are aimed at providing high bandwidth up to 1 GHz for faster communication at the speed of up to 10 GB/s with low latency of 1 ms [10]. HTTP adaptive streaming introduces high latency, and are not suitable for real-time monitoring applications. Web Real-Time Communication (WebRTC) [11] is an open-source project started by Google in 2011 that provides real-time communication for browser-based applications [12]. Considering the lack of 5G deployments in forests 4G can be used with WebRTC to provide video communication with relatively ultra-low latency. In our previous work [13], we performed an offline real-time handheld device-based tree species identification for the rural areas where internet infrastructure is sparse. We used CNN based MobileNetV3 model, trained on 184 tree species and used this model in android based application for real-time offline tree species identification. MobileNet models are lightweight and mostly aimed at solving classification problems. For multi-objects detecting MobileNet models are augmented with SSD or Fast R-CNN as a backbone [14], [15], requiring high computational costs, not available on mobile devices. A more native architecture used for object detection is YOLOv5, which can also be used to detect smaller objects with consistent speed and accuracy.

We use wireless networking technologies with WebRTC, a protocol suite intended for use with real-time applications, to stream video to the cloud where YOLOv5 is hosted. The outrput of the machine learning model is then displayed on web-based application, accessible for end-users (forest

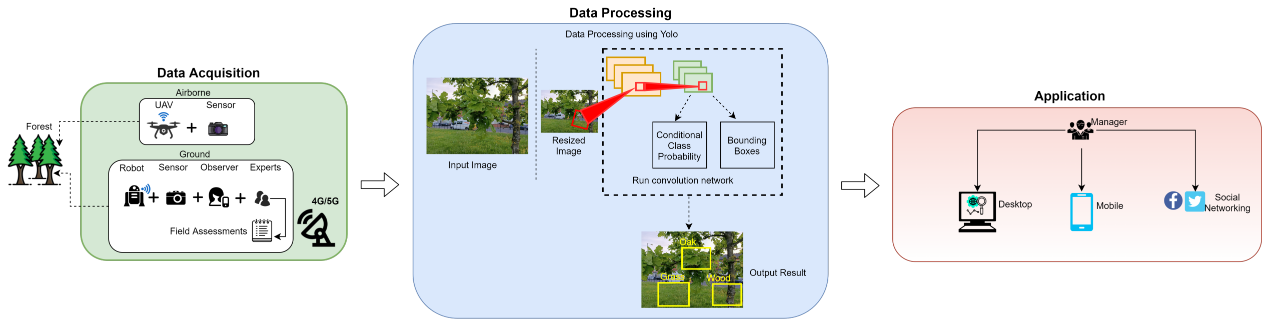

Fig. 1. System Architecture

managers/ranger). We envisage to use UAVs equipped with sensing devices above the canopy, however, for this paper we used mobile devices to capture tree images from the ground. This approach speeds up and automate the process of forest monitoring and help forest managers to observe live FHI remotely. To the best of our knowledge, the combination of YOLOv5, WebRTC, and 4G/WiFi networks has not been investigated for live FHI detection. Our main contributions are (1) design of novel approach to stream high definition videos to the YOLOv5 machine learning algorithm to perform live monitoring of health of the forest (2) a non-intrusive precise tree class detection, comparable to the precision of state-of-the-art models while significantly requiring less number of iterations, making it more suitable for remote sensing (3) training YOLOv5 model on custom data for accurate tree species detection.

II. Literature Review

A wide range of object detection algorithms have been studied in the field of computer vision and some of the most popular algorithms are convolutional neural networks (CNN) and GPU-accelerated deep learning frameworks. CNN-based object detectors have the flexibility of being trained with classes that can range up to a few thousands. Some of the popular CNN object detection algorithms are Regions Based CNN (R-CNN) [16], Fast R-CNN [17], Faster R-CNN [18], Region Based Fully CNN (R-FCN) [19], Single Shot Detector (SSD) [20] and the You Only Look Once (YOLO) series [21]. Depending on the application and the evaluation metric to be prioritized, different models can be deployed.

Deep CNNs object detection frameworks can be categorized into two kinds: the two-stage and single-stage. A few examples of two-stage object detection algorithms are R-CNN, Spatial Pyramid Pooling (SPP-Net) [22], Fast R-CNN, Faster R-CNN. R-CNN is the first real target detection model based on CNN that achieves a mean average precision (mAP) of 66% [16]. Although the accuracy as compared to traditional methods of detection is improved, the calculation efficiency is observed to be too low [7]. Moreover, directly scaling the region proposal to a fixed-length feature vector can cause object distortion. Unlike R-CNN, SPP-Net performs feature extraction on the entire image only once, avoiding repeated calculations. However, both algorithms share the same shortcomings, including the complexities of multi-step training and the requirement of training separate SVM classifiers and additional regressors [22]. Although Fast R-CNN and Faster-RCNN provide improvement in detection accuracy, they still can not achieve real-time detection [7].

One-stage detection algorithm examples include YOLOv1, YOLOv2, YOLOv3, SSD, YOLOv4, and YOLOv5. Unlike the two-stage detection models, YOLOv1 has a simple CNN network structure without the extraction process of region proposal. It is based on the idea of using the entire graph as input of the network that outputs the location and category of the bounding box. Though the detection speed of YOLOv1 can reach up to 45 frames per seconds (or fps), it produces background errors and poor recognition performances for objects in the form of groups [7]. YOLOv2 uses Darknet-19 for fully convolutional feature extraction and anchor box mechanism, k-means clustering, multi-scale training, which improves the recall and accuracy although, detecting targets with high overlap or small size is still challenging. YOLOv3 adopts Darknet-53 for a deeper feature extraction network, multiple scales for prediction, up sampling fusion method, and finally merging of three scales, which largely improves the effect of small objects and detection speed [21]. However, the detection accuracy is still not improved, especially when the intersection over union (IoU) > 0.5. SSD combines the concept of regression in the YOLO algorithm and anchor box in Faster R-CNN. SSD face poor classification for small objects, and a single object is simultaneously detected by boxes of different sizes [20]. YOLOv4 uses CSPDarknet53 as the backbone network and adds Cross Stage Partial Connection, Weighted Residual Connection, Self adversarial training, Cross mini Batch Normalization, Mosaic data augmentation, DropBlock, Mish activation, and CIoU to the first YOLO framework. These modifications help in increasing the speed and accuracy by 20% and 10%, respectively [7]. YOLOv5 is the first YOLO algorithm where the backbone network does not consist of Darknet but PyTorch. It is a lightweight algorithm that is easier to use, train and it infers more quickly and performs better than previous versions. Therefore, for time-critical applications, YOLOv5 has the potential to be most effective for object detection operations.

Various versions of YOLO have been used for different applications in forest settings to measure numerous FHI. For example, [23] uses YOLOv2 to detect trees for tree girth and height evaluation. Binary classification of the tree and non-tree objects was performed with accuracy between 87% to 93% on the test dataset. The authors of [24] used YOLOv3 for fire detection with 98% accuracy. It was not clear how many iterations and minutes took to achieve this that accuracy. Similarly, for forest fire detection, a comparison of Fast R-CNN, different YOLO versions (tiny-yolo-voc, tiny-yolo-voc1, yolo-vo.2.0, YOLOv3), and SSD was performed by [25]. They trained each model in 120,000 iterations to get the best accuracy of 92.29% with YOLOv3, 99.7% with Fast R-CNN, and 99.88% with SSD. However, the time to train these models is not recorded. We compare our research findings with these results in terms of the number of iterations, time to train the model and the accuracy achieved given in Table I in Section IV.

III. Methods and Materials

A. YOLOv5

YOLOv5's network structure contains the same set of components as any other YOLO series which includes Input, Backbone and Neck. In addition, YOLOv5 input uses adaptive anchor frame calculation that adaptively gives the optimal anchor frame in different training sets. The YOLOv5 Backbone includes the Focus structure to realize the slicing operation and the Neck uses a new FPN structure, that enhances the propagation of low-level features [26]. As a result, YOLOv5 achieves a reduction in computation complexity at least by a factor of four [27]. YOLOv5 is pre-trained on Common Objects in Context (COCO) [28] dataset, an immense repository of images used for object detection, segmentation and captioning. The dataset contains a total of 330K images, out of which over 200K are labelled images with 80 different object classes and 1.5 million object instances. In contract to these methods, we trained the YOLOv5 model on our custom dataset. The system architecture is shown in Fig. 1.

B. WebRTC

Streaming protocols define how videos and audios can be transmitted across the Internet from one device or system to another [29]. An encoder generally starts the transmission, which is then ingested by a media server. Real-Time Streaming Protocol (RTSP) [30] and Real-Time Messaging Protocol (RTMP) [31] are frequently used on the encoder side. For browser-based applications, WebRTC is more prominent than RTMP or RTSP and provides higher Quality of Service (QoS) and Quality of Experience (QoE) to the end user [32]. As such, in our system we adopt WebRTC for delivering images/video to the cloud.

C. Dataset and Training

For the plant detection use case, we created our own dataset using a 4K quality RGB camera of a mobile phone. The images were resized to 640x640 pixels and labelled using Makesense.ai [33]. The images were annotated using bounding boxes for three classes: oak, wood, and grass. The Yolov5s model was trained using Google colab [34] on our custom dataset using pre-trained COCO weights. The hardware to train the model included a Lenovo laptop equipped with an 8265U CPU at 1.80 GHz of Intel Core i5, 8 GB of RAM running on a Windows 10 64-bit system. The dataset was divided in a ratio of 75:25 to get a training set and a validation set, respectively. We trained the model in 1000 epochs which took approximately 24 minutes.

IV. Results and Discussion

A. Accurate Object Detection in Forest

TABLE I
Comparison of our model with other models in terms of number of Iterations and Accuracy

Model	Iterations	Accuracy (%)
Fast R-CNN [21]	120,000	99.7
YOLOv3 [21]	120,000	92.29
SSD [21]	120,000	99.88
YOLOv5 [Ours]	**1000**	99.8

The results of training and validation sets in Fig. 2 shows three types of loss: objectness loss, box loss and classification loss. Objectness loss measures the probability that an object exists in a proposed region of interest. If the objectness is high, this means that the image window is likely to contain an object. The box loss points to how well the model can locate the centre of an object and how well the predicted bounding box covers an object. Classification loss indicates how well the algorithm can predict the correct class of an object. The model improved significantly in terms of precision, recall and mAP after 250 epochs and becomes stable after 500 epochs, which means stopping model early would give almost same results in 50% less time.

After the model was trained, for the model inference, we fed unseen pictures and a video into the model with confidence threshold of 0.25. Fig. 4 shows that the algorithm can detect oak, wood and grass with prediction value of more than 90% in almost all instances. The mAP results for each class at IoU 0.5 and from 0.5 to 0.95 are shown in Table II.

TABLE II
Accuracy of YOLOv5 Model for the classes oak, wood, grass and overall

Classes	mAP@0.5	mAP@0.5:0.95	Precision	Recall
oak	0.996	0.87	0.997	1
wood	0.995	0.661	1	1
grass	0.995	0.914	0.996	1
all	0.995	0.815	0.998	1

B. Low Latency

For live object detection, we created a web application consists of HTML, Javascript and CSS files running on the

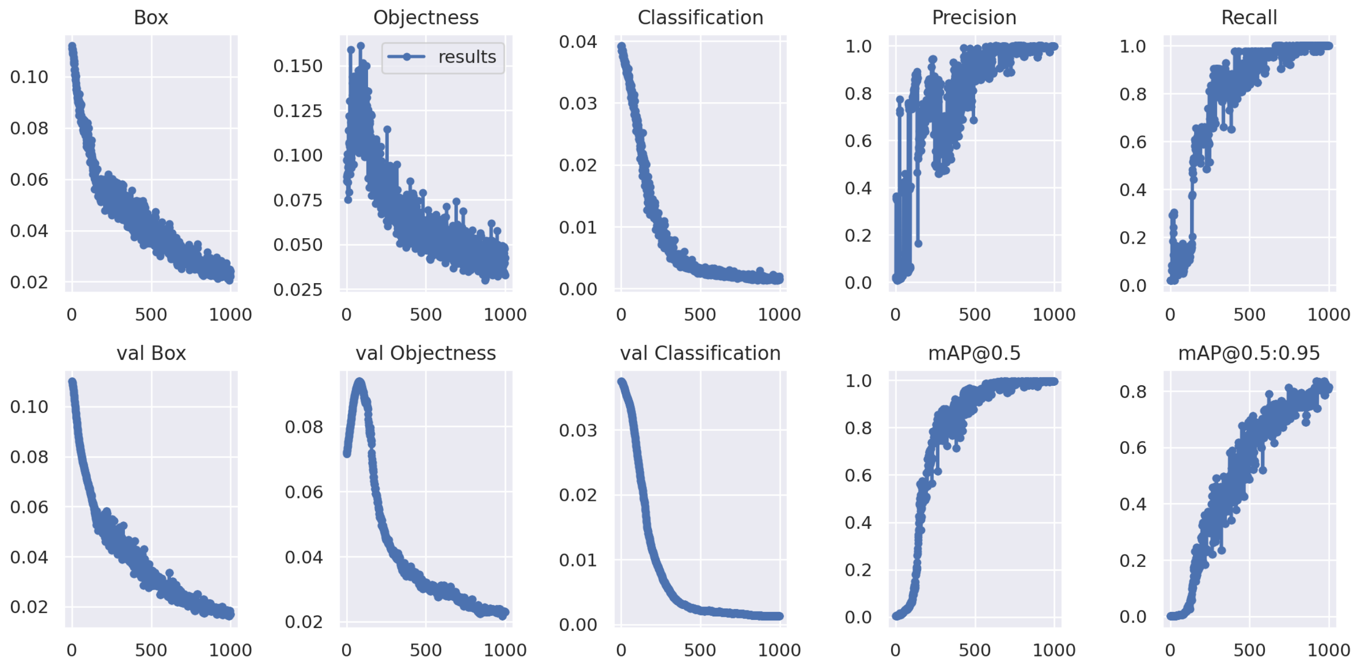

Fig. 2. The graphs of box loss, objectness loss, classification loss, precision, recall and mean average precision (mAP) over the training epochs over the training and validation sets

cloud. The application utilized Yolov5 models with trained weights to perform object detection on live stream from multiple cameras. WebRTC is used for live streaming from these cameras e.g. mobile device and webcam.

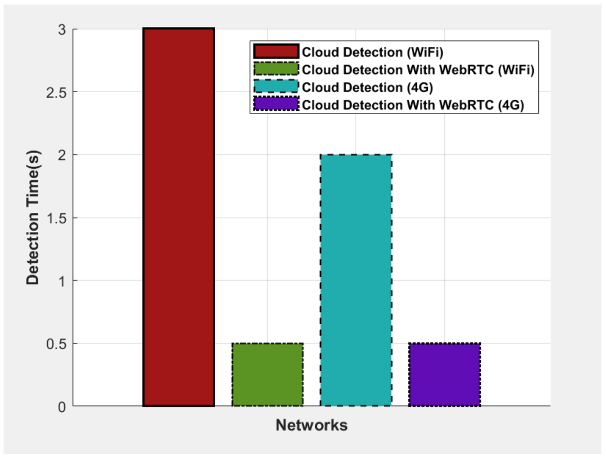

Fig. 3. Network Latency using WiFi and 4G

We observed the latency via both WiFi (IEEE 802.11ac or WiFi6) and 4G networks with and without WebRTC. Our results show in Fig. 3, that it took upto 2 seconds when video was streamed using 5G without WebRTC. On the other hand, it took on average upto 0.5 seconds when streamed using 4G with WebRTC. Similarly using WiFi it took 3 seconds without WebRTC and upto 0.5 seconds with WebRTC. We also noted object detection time in each frame of the video. Each frame took 40ms to be processed, i.e. 25 FPS to detect objects of 3 classes with average accuracy of 0.99, near real-time object detection.

V. Conclusions

Real-time forest monitoring using the latest technologies is essential to observe forest health remotely for forest managers and ecologists. For this reason, this paper analysed the YOLOv5 algorithm for object detection and compared it with its previous versions and other machine learning algorithms such as Fast R-CNN, SSD and found that YOLOv5 is lightweight, easy to use and fast to train and detect objects. We trained YOLOv5 on custom imagery data taken from a forest to detect different objects, mainly: tree species, wood and grass. For non-intrusive monitoring, we used cameras of mobile devices in this paper and in the future, we are planning to use cameras mounted on the drone. For real-time imagery data transmission, we used 4G networks with integrated WebRTC for ultra-low latency. Our experiments showed that compared to other algorithms with YOLOv5, accuracy of 99.9% could be achieved within significantly fewer iterations and much less time.

VI. Future Work

In this preliminary investigation, we detected only one tree species, i.e. oak and distinguished it with grass and wood. We can extend the training of the algorithm with multiple tree species detection in single video frame or different frames. For this, we need to expand our dataset with capturing images of different tree species. We can detect other FHI including

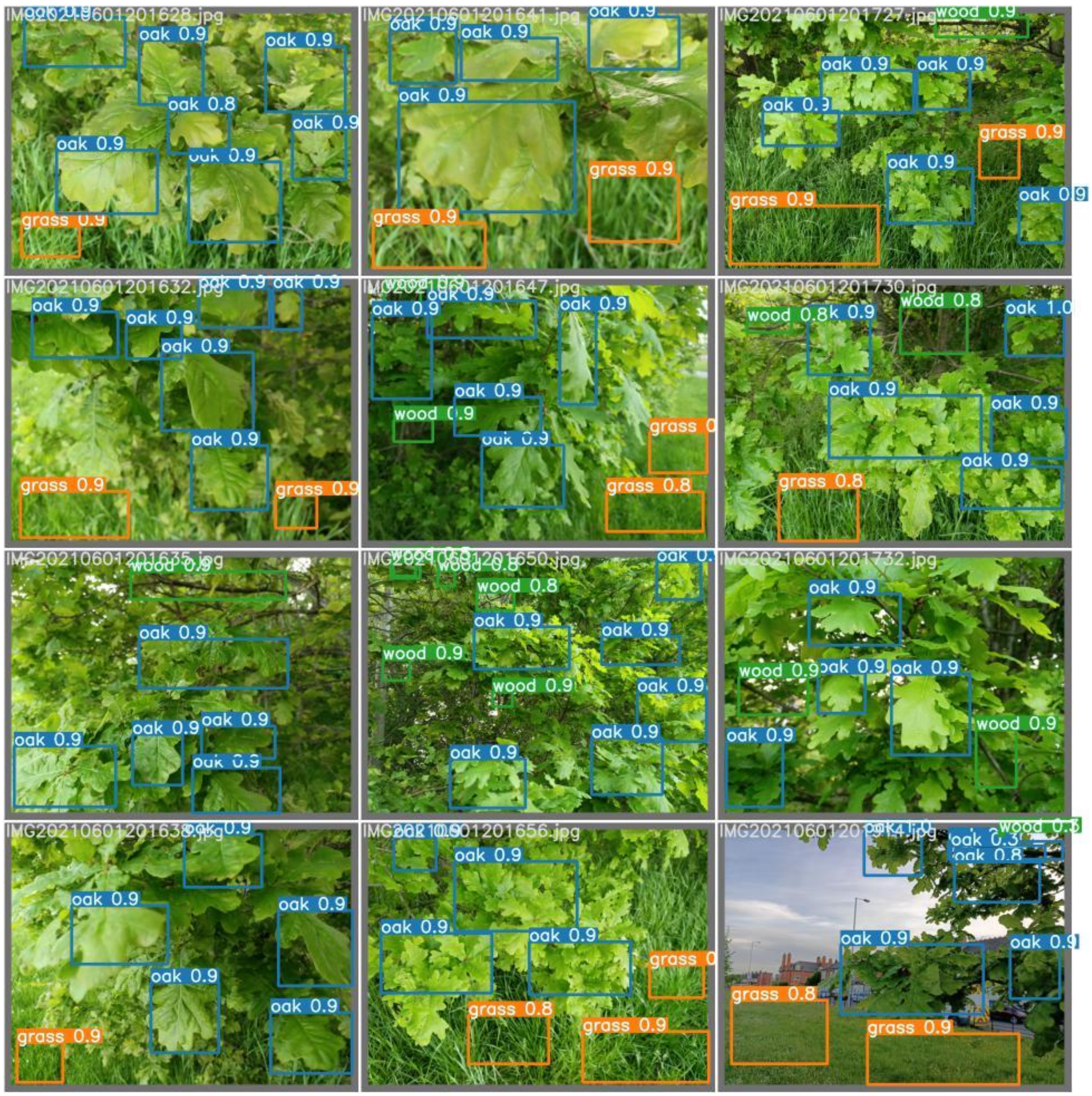

Fig. 4. The percentage of Accuracy of object detection for classes: oak, wood, and grass

deadwood, wildlife signs (presence of squirrels, rabbits, and birds). We are working towards attaching mobile phone and other sensing devices with UAV and capture 4K imagery above the canopy to train the algorithm and process data. The outcome of this work can be used for time-critical applications such as forest fire detection and search/rescue.

Acknowledgment

This work was funded by the Department for Digital, Culture, Media & Sport (DCMS) as part of the 5G Connected Forest (5GCF) project under its 5G Testbeds and Trials Program. *The corresponding authors for this work are Ambreen Hussain and A. Taufiq Asyhari.

References

[1] L. Gamfeldt, T. Snäll, R. Bagchi, M. Jonsson, L. Gustafsson, P. Kjellander, M. C. Ruiz-Jaen, M. Fröberg, J. Stendahl, C. D. Philipson *et al.*, "Higher levels of multiple ecosystem services are found in forests with more tree species," *Nature communications*, vol. 4, no. 1, pp. 1–8, 2013.

[2] "About forest health index – forest health," https://foresthealthindex.org/about/ , accessed: 06.16.2021.

[3] D. D. Slawson and A. J. Moffat, “How effective are citizen scientists at contributing to government tree health public engagement and surveillance needs—an analysis of the uk open air laboratories (opal) survey model,” *Insects*, vol. 11, no. 9, p. 550, 2020.

[4] G. E. Austen, M. Bindemann, R. A. Griffiths, and D. L. Roberts, “Species identification by experts and non-experts: comparing images from field guides,” *Scientific Reports*, vol. 6, no. 1, pp. 1–7, 2016.

[5] N. MacLeod, M. Benfield, and P. Culverhouse, “Time to automate identification,” *Nature*, vol. 467, no. 7312, pp. 154–155, 2010.

[6] D. Han, Q. Liu, and W. Fan, “A new image classification method using cnn transfer learning and web data augmentation,” *Expert Systems with Applications*, vol. 95, pp. 43–56, 2018.

[7] J. Deng, X. Xuan, W. Wang, Z. Li, H. Yao, and Z. Wang, “A review of research on object detection based on deep learning,” in *Journal of Physics: Conference Series*, vol. 1684, no. 1. IOP Publishing, 2020, p. 012028.

[8] N. Guimarães, L. Pádua, P. Marques, N. Silva, E. Peres, and J. J. Sousa, “Forestry remote sensing from unmanned aerial vehicles: A review focusing on the data, processing and potentialities,” *Remote Sensing*, vol. 12, no. 6, p. 1046, 2020.

[9] “The mobile economy,” https://www.gsma.com/mobileeconomy/, accessed: 06.16.2021.

[10] S. Li, L. Da Xu, and S. Zhao, “5g internet of things: A survey,” *Journal of Industrial Information Integration*, vol. 10, pp. 1–9, 2018.

[11] “Webrtc,” https://webrtc.org/ , accessed: 06.13.2021.

[12] S. Loreto and S. P. Romano, “How far are we from webrtc-1.0? an update on standards and a look at what’s next,” *IEEE Communications Magazine*, vol. 55, no. 7, pp. 200–207, 2017.

[13] A. Hussain, “Performance of mobilenetv3 transfer learning on handheld device-based real-time tree species identification,” (in press).

[14] R. Guo and X. Xie, “Object detection method of autonomous vehicle based on lightweight deep learning,” SAE Technical Paper, Tech. Rep., 2021.

[15] Z. Chang, L. Hao, H. Tan, and W. Li, “Design of mobile garbage collection robot based on visual recognition,” in *2020 IEEE 3rd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*. IEEE, 2020, pp. 448–451.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[17] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[18] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[19] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn: Object detection via region-based fully convolutional networks,” in *Advances in neural information processing systems*, 2016, pp. 379–387.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[22] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[23] K. Itakura and F. Hosoi, “Automatic tree detection from three-dimensional images reconstructed from 360 spherical camera using yolo v2,” *Remote Sensing*, vol. 12, no. 6, p. 988, 2020.

[24] Y.-Y. Qin, J.-T. Cao, and X.-F. Ji, “Fire detection method based on depthwise separable convolution and yolov3,” *International Journal of Automation and Computing*, vol. 18, no. 2, pp. 300–310, 2021.

[25] S. Wu and L. Zhang, “Using popular object detection methods for real time forest fire detection,” in *2018 11th International symposium on computational intelligence and design (ISCID)*, vol. 1. IEEE, 2018, pp. 280–284.

[26] J. Yao, J. Qi, J. Zhang, H. Shao, J. Yang, and X. Li, “A real-time detection algorithm for kiwifruit defects based on yolov5,” *Electronics*, vol. 10, no. 14, p. 1711, 2021.

[27] “Gcp automl vs. yolov5 for training a custom object detection model,” https://medium.com/slalom-data-analytics/gcp-automl-vs-yolov5-for-training-a-custom-object-detection-model-c1481b8a5c58 , accessed: 06.17.2021.

[28] “Coco - common objects in context,” https://cocodataset.org/home , accessed: 06.13.2021.

[29] J. G. Apostolopoulos, W.-t. Tan, and S. J. Wee, “Video streaming: Concepts, algorithms, and systems,” *HP Laboratories, report HPL-2002-260*, 2002.

[30] Y. Liu, B. Du, S. Wang, H. Yang, and X. Wang, “Design and implementation of performance testing utility for rtsp streaming media server,” in *2010 First International Conference on Pervasive Computing, Signal Processing and Applications*. IEEE, 2010, pp. 193–196.

[31] X. Lei, X. Jiang, and C. Wang, “Design and implementation of streaming media processing software based on rtmp,” in *2012 5th International Congress on Image and Signal Processing*. IEEE, 2012, pp. 192–196.

[32] I. Santos-González, A. Rivero-García, J. Molina-Gil, and P. Caballero-Gil, “Implementation and analysis of real-time streaming protocols,” *Sensors*, vol. 17, no. 4, p. 846, 2017.

[33] “Make sense,” https://www.makesense.ai/ , accessed: 06.13.2021.

[34] “Yolov5 tutorial - colaboratory,” https://colab.research.google.com/github/ultralytics/yolo5/, accessed: 06.13.2021.