Vegetation classification using DeepLabv3+ and YOLOv5

Paulo A. S. Mendes[1], A. Paulo Coimbra[2] and Aníbal T. de Almeida[3]

Abstract— **Semantic segmentation and object detection are challenging tasks in computer vision. In recent years the performance of semantic segmentation and object detection has been greatly improved by using deep learning techniques. A large number of novel methods have been proposed to achieve relevant results in different applications, namely autonomous vehicles, mobile robotics and agriculture. A key challenge for autonomous navigation in cluttered outdoor environments is the reliable discrimination between obstacles that must be avoided at all costs, and obstacles/objects that need to be identified to pursue the intended action of the robot. In this paper are presented and compared DeepLabv3+ semantic segmentation and YOLOv5 object detection of vegetation, to be used by an Unmanned Ground Vehicle (UGV) to clean fuel in forest environments, to prevent fires. The results were obtained with a dataset taken from a local forest.**

I. INTRODUCTION

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks like recognize and locate objects of interest within a matter of moments. Fast, accurate algorithms for object detection would allow computers to drive cars with simple sensors such as cameras, enable assistive devices to output real-time scene information to human users and robotic systems. Object detection is a computer vision technique that allows a computer to identify and locate objects in images. Object detection allows identification and localization of objects and can be used to count objects in scenes, pedestrian detection, face detection, text detection, pose detection, number-plate recognition, determining and tracking their precise locations, all while accurately labeling them. Most of the early object detection algorithms were built based on handcrafted features.

The main goal of this work is to detect and classify vegetation to be cleaned by an Unmanned Ground Vehicle (UGV) in forests, to reduce fire risk. To achieve better cleaning results and help autonomous navigation, a deep learning model is used to localize vegetation in images, in real time. The main contribution of this work is the comparison of two models to classify vegetation to be cleaned by an UGV in forest complex environments.

This paper is organized as follows: a brief state of the art in object detection and in semantic segmentation is presented in section II, in section III is presented the DeepLab network, in section IV is presented the YOLO deep learning network, in section V is presented the methodology and results, followed by the conclusions.

[1]Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal email: 33paulomendes@gmail.com

[2]Institute of Systems and Robotics and Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal email: acoimbra@isr.uc.pt

[3]Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal email: adealmeida@isr.uc.pt

II. RELATED WORK

P. Viola and M. Jones achieved real-time detection of human faces for the first time without any constraints [1][2]. The Viola and Jones detector go through all possible locations and scales in an image to see if any window contains a human face, using sliding windows. The Viola and Jones detector has dramatically improved its detection speed by incorporating three important techniques: integral image, feature selection, and detection cascades.

Histogram of Oriented Gradients (HOG) feature descriptor was originally proposed in 2005 by N. Dalal and B. Triggs [3]. HOG can be considered as an important improvement of the scale-invariant feature transform [4] and shape contexts [5] of its time. To balance the nonlinearity and feature invariance such as translation, scale or illumination, the HOG descriptor is designed to be computed on a dense grid of uniformly spaced cells and use overlapping local contrast normalization for improving accuracy. HOG can be used to detect a variety of object classes but it was motivated, primarily, by the problem of pedestrian detection. To detect objects of different sizes, the HOG detector rescales the input image for multiple times while keeping the size of a detection window unchanged.

Systems like Deformable Parts Models (DPM) use a sliding window approach where the classifier is run at evenly spaced locations over the entire image [6]. A typical DPM detector is composed of a root-filter and a number of part-filters. Instead of manually specifying the configurations of the part filters (e.g., size and location), a weakly supervised learning method is developed in DPM where the configurations of the part filters can be learned automatically as latent variables. Girshick *et al.* concluded this process as a special case of multi-instance learning [7]. Other important techniques such as "hard negative mining", "bounding box regression", and "context priming" are also applied for improving detection accuracy. To speed up the detection, Girshick *et al.* developed a technique for compiling detection models into a faster one that implements a cascade architecture, which has achieved over 10 times acceleration without sacrificing any accuracy [8][9].

In the deep learning era, object detection can be grouped into two genres, and these are "two-stage detection" and "one-stage detection". Two-stage detection frames the detection as a coarse-to-fine process while one-stage detection

frames it as complete in one step.

Recent approaches like R-CNN (two-stage detector) use region proposal methods to first generate potential bounding boxes in an image and then run a classifier on these proposed boxes. After classification, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and rescore the boxes based on other objects in the scene [10]. These complex pipelines are hard to optimize because each individual component must be trained separately resulting in a slow processing. Recently, deep ConvNets [11] have significantly improved image classification and object detection accuracy [12]. Compared to image classification, object detection is a challenging task that requires complex methods to solve the problem. Due to this complexity, current approaches (e.g., [12], [13], [14], [15]) train models in multi-stage pipelines that are slow and inelegant.

In 2014, K. He *et al.* proposed Spatial Pyramid Pooling Networks (SPPNet) [16]. The main contribution of SPPNet (two-stage detector) is the introduction of a Spatial Pyramid Pooling (SPP) layer, which enables a Convolutional Neural Network (CNN) to generate a fixed-length representation regardless of the size of the image/region of interest without rescaling it. When using SPPNet for object detection, the feature maps are computed from the entire image only once, then, fixed length representations of arbitrary regions can be generated to train the detectors, which avoids repeatedly calculating the convolutional features. SPPNet is more than 20 times faster than R-CNN without sacrificing any detection accuracy.

In 2015, Girshick *et al.* proposed the Fast RCNN [17] a two-stage detector. Fast R-CNN enables simultaneously training a bounding box regressor and a detector under the same network configurations. Although Fast-RCNN successfully integrates the advantages of SPPNet and R-CNN, the detection speed is limited by the proposal detection.

In 2015, Ren *et al.* proposed Faster R-CNN detector [18] shortly after the Fast R-CNN. Faster RCNN (two-stage detector) is the first end-to-end, and the first near real time deep learning detector. The contribution of Faster-RCNN is the introduction of the Region Proposal Network (RPN) that enables nearly cost-free region proposals. From R-CNN to Faster RCNN, most individual blocks of an object detection system, e.g., bounding box regression, feature extraction, proposal detection, etc..., have been gradually integrated into a unified, end-to-end learning framework.

In 2017, Lin *et al.* proposed Feature Pyramid Networks (FPN) [19] a two-stage detector on basis of Faster R-CNN. CNN naturally forms a feature pyramid through its forward propagation. FPN shows great advances to detect objects with a wide variety of scales. Using FPN in a basic Faster R-CNN system, it achieves state-of-the-art single model detection results.

"You Only Look Once" (YOLO) proposed by Redmon *et al.* [20] in 2015 was the first one-stage detector in the deep learning era. This network divides the image into regions and predicts bounding boxes and probabilities for each region simultaneously. Later, Redmon *et al.* has made a series of improvements on basis of YOLO and has proposed its second and third versions [21][22], which further improve the detection accuracy while keeping a very high detection speed.

Single Shot MultiBox Detector (SSD) [23] was proposed by Liu *et al.* in 2015. It was the second one-stage detector in the deep learning era. The main contribution of SSD is the presentation of the multi-reference and multi-resolution detection techniques. Multi-reference and multi-resolution significantly improves the detection accuracy of a one-stage detector, especially for small objects.

In 2017, RetinaNet has been proposed by Lin *et al.* [24]. A new loss function named "focal loss" has been introduced in RetinaNet by reshaping the standard cross entropy loss so that the detector will put more focus on hard, misclassified examples during training. Focal Loss makes possible for the one-stage detectors to achieve comparable accuracy of two-stage detectors while maintaining very high detection speed.

Nowadays, semantic segmentation is one of the key problems in the field of computer vision. Semantic segmentation is the process of dividing an image into multiple segments, all objects of the same type are marked using one class label. As semantic segmentation is able to provide the class information at the pixel level, many real-world applications benefit from this task, such as defect detection [25], computer aided diagnosis [26], therapy planning [27], self-driving vehicles [28] and pedestrian detection [29]. Fine-grained inference is achieved by semantic segmentation by making dense predictions inferring labels for every pixel, so that each pixel is labeled with the class of its enclosing object. Certain deep networks have made such significant contributions to the field that they have become widely known standards, it is the case of GoogLeNet, DeepLab, AlexNet, ResNet, VGG-16, U-Net, Fast Fully Convolutional Network and Gated-SCNN.

AlexNet was the pioneering Deep Convolutional Neural Network (DCNN) that won the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC- 2012) with a TOP-5 test accuracy of 84.6%. This architecture was presented by Krizhevsky *et al.* [30]. It consists of Rectified Linear Units (ReLUs) as non-linearities, five convolutional layers, max-pooling ones, three fully-connected layers, and dropout.

Visual Geometry Group (VGG) from the University of Oxford introduced the Visual Geometry Group (VGG) CNN model [31]. They proposed various models and configurations of deep CNN's [31], one of them was submitted to the ILSVRC-2013. That model, also known as VGG-16 because it is composed by 16 weight layers, became popular thanks to its achievement of 92.7% TOP-5 test accuracy in the ILSVRC-2013. The main difference between VGG-16 and CNN models is the use of a stack of convolution layers with small receptive fields in the first layers instead of few layers with big receptive fields. This leads to less parameters and more non-linearities, thus making the model easier to train and the decision function more discriminative.

Szegedy *et al.* introduced GoogLeNet [32] which won the ILSVRC-2014 challenge with a TOP-5 test accuracy of

93.3%. This CNN architecture is composed by 22 layers and a newly introduced building block called inception module. This approach proved that CNN layers could be stacked in more ways than a typical sequential manner. In fact, those modules consist of a Network in Network (NiN) layer, small-sized convolution layer, a large-sized convolution layer and a pooling operation. All of them are computed in parallel and followed by 1×1 convolu- tion operations to reduce dimensionality. Thanks to those modules, this network puts special consideration on memory and computational cost by significantly reducing the number of parameters and operations.

Microsoft's ResNet [33] is known thanks to winning ILSVRC-2016 with 96.4% accuracy. The network is well-known due to its depth (152 layers) and the introduction of residual blocks. The residual blocks address the problem of training a really deep architecture by introducing identity skip connections so that the layers can copy their inputs to the next layer. The intuitive idea behind this approach is that it ensures that the next layer learns something new and different from what the input has already encoded (since it is provided with both the output of the previous layer and its unchanged input). In addition, this kind of connections help overcoming the vanishing gradient problem.

In order to extend Recurrent Neural Networks (RNNs) architectures to multi-dimensional tasks, Graves *et al.* [34] proposed a Multi-dimensional Recurrent Neural Network (MDRNN) architecture which replaces each single recurrent connection from standard RNNs with d connections, where d is the number of spatio-temporal data dimensions. Based on this initial approach, Visin *el al.* proposed ReNet [35] architecture in which instead of multidimensional RNNs, they have been using usual sequence RNNs. In this way, the number of RNNs is scaled linearly at each layer regarding to the number of dimensions d of the input image ($2D$). In this approach, each convolutional layer (convolution + pooling) is replaced with four RNNs sweeping the image vertically and horizontally in both directions.

U-Net [36] is a CNN originally developed for segmenting biomedical images. Its architecture looks like the letter U when visualized and hence the name U-Net. Its architecture is made up of two parts, the left part the "contracting path" and the right part the "expansive path". The purpose of the contracting path is to capture context while the role of the expansive path is to aid in precise localization. The contracting path is made up of two 3×3 convolutions. The convolutions are followed by a rectified linear unit and a 2×2 max-pooling computation for downsampling.

In Fast Fully Convolutional Network (FFCN) [37] architecture, a Joint Pyramid Upsampling (JPU) module is used to replace dilated convolutions since they consume a lot of memory and time. It uses a fully-connected network at its core while applying JPU for upsampling. JPU upsamples the low-resolution feature maps to high-resolution feature maps.

Gated shape CNN's (Gated-SCNN) [38] architecture consists of a two-stream CNN architecture. In this model, a separate branch is used to process image shape information. The shape stream is used to process boundary information.

Indoor RGBD pixel-wise semantic segmentation has also gained popularity since the release of the NYU dataset [39]. This dataset shows the usefulness of depth data to improve segmentation. Their approach used features such as RGB-SIFT [40], depth-SIFT [41] and pixel location as input to a neural network classifier to predict pixel unaries. The noisy unaries are then smoothed using a Conditional Random Fields (CRF). In a more recent work [39], both class segmentation and support relationships are inferred together using a combination of RGB and depth based cues. Another approach focuses on real-time joint reconstruction and semantic segmentation, where Random Forests (RF) are used as the classifier [42].

In Mask R-CNN [43] objects are classified and localized using a bounding box and semantic segmentation that classifies each pixel into a set of categories. Every region of interest gets a segmentation mask. Class labels and bounding boxes are produced as the final output. The architecture is an extension of the Faster R-CNN. The Faster R-CNN is made up of a Deep Convolutional Neural Network (DCNN) that proposes the regions and a detector that utilizes the regions.

DeepLab architecture [44], convolutions with upsampled filters are used for tasks that involve dense prediction. Segmentation of objects at multiple scales is done via atrous spatial pyramid pooling. Finally, DCNN's are used to improve the localization of object boundaries. Atrous convolution is achieved by upsampling the filters through the insertion of zeros or sparse sampling of input feature maps. DeepLabv3+ [45] employs the encoder-decoder structure where DeepLabv3 is used to encode the rich contextual information and a simple yet effective decoder module is adopted to recover the object boundaries. It is also explored the Xception model and atrous separable convolution to make the proposed model faster and stronger.

Finally, experimental results show that DeepLabv3+ semantic segmentation sets a new state-of-the-art performance on PASCAL VOC 2012 [46] and Cityscapes [47] datasets. YOLO is the fastest general-purpose object detector in the literature and YOLO pushes the state of the art in real-time object detection. YOLO also generalizes well to new domains making it ideal for applications that rely on fast, robust object detection.

III. DeepLabv3+

The Deeplab series network was proposed by Chen *et al.* [48]. It is a model specifically used to deal with semantic segmentation. Currently, four versions have been launched.

Deeplabv1 [48] was developed based on the VGG16 network, first removing the last fully connected layer to achieve end-to-end output. Because convolution itsef has translation invariance and pooling can enhance this feature of the network, the last two pooling layers are removed. In order to solve the ability for multiscale segmentation of objects, since DeepLabv1 have poor ability for multiscale segmentation, in Deeplabv2 [49], Chen *et al.* concluded that VGG16 have limited expressive power, and replaced it

with the ResNet-101 backbone, which is more complex and expressive. Also an Atrous Spatial Pyramid Pooling (ASPP) structure is proposed. Deeplabv3 [50] improves ASPP, uses hole convolution to deepen the network and discards the Conditional Random Field (CRF). CRF is no longer needed because the accuracy of the classification results has been improved. DeepLabv3+ is a state of the art deep learning model for semantic image segmentation, where the goal is to assign semantic labels (such as a road, a dog, a rider or a person) for every pixel in the input image. Open sourced by Google in 2016, multiple improvements have been made to the model with the latest being DeepLabv3+ [51].

Deeplabv3+ boost Deeplabv3 by adding a simple yet effective decoder module to refine the segmentation results, especially along the object boundaries. It includes the encoder and the decoder parts. The encoder is mainly used for reducing the dimensionality of the feature map and for extracting features. The decoder is mainly used to restore resolution of the feature map and the edge information to obtain the final semantic segmentation results. To maintain the resolution of the feature map and to increase the receptive field, the convolution operation of the last few convolutional layers of the encoder is replaced with hole convolution. To obtain multi-scale semantic contextual information the atrous spatial pyramid pooling (ASPP) module introduced in Deeplabv3+ uses dilation convolution at various rates. By using these novel structures, Deeplabv3+ produces accurate semantic segmentation results among different datasets.

IV. YOLOv5

Object detection is a computer vision technique that allows us to locate and identify objects in an image. With this kind of identification and localization, object detection can be used to detect various types of objects in a scene and determine and track their precise locations, all while accurately labeling them. Image classification involves the designation of a class label to an image, whereas object localization comprise drawing a bounding box around one or more objects in an image. Object detection is challenging and combines the two tasks and draws a bounding box around each object of interest in the image and assigns them a class label.

The YOLO model was first described by Joseph Redmon *et al.* [20]. The method involves a single neural network trained end to end with an image input and predicts bounding boxes and class labels. The technique offers lower predictive accuracy (e.g. more localization errors), although it operates at 45 frames per second and can operate up to 155 frames per second for a speed-optimized version of the model. The model works by splitting the input image into a grid of cells, each cell is responsible for predicting a bounding box if the center of a bounding box falls within the cell. Each grid cell predicts a bounding box involving the x, y coordinates, the width and height and the confidence. A class prediction is based on each cell. The class probability map and the bounding boxes with confidences are after combined into a final set of bounding boxes and class labels.

The model was updated by Joseph Redmon and Ali Farhadi in an effort to further improve the model performance [21]. Various training and architectural changes were made to the model, such as the use of high-resolution input images and batch normalization. YOLOv2 model makes use of anchor boxes like Faster R-CNN, pre-defined bounding boxes with useful shapes and sizes that are tailored during training. The choice of bounding boxes for the image are preprocessed using a k-means analysis on the training dataset. Importantly, the predicted representation of the bounding boxes is changed to allow small changes to have a less impressive effect on the predictions, resulting in a stronger model. Rather than predicting size and position directly, for moving and reshaping the pre-defined anchor boxes offsets are predicted relative to a grid cell and dampened by a logistic function.

Further improvements to the model were proposed by Joseph Redmon and Ali Farhadi in YOLOv3 [22]. The improvements were reasonably minor, including minor representational changes and a deeper feature detector network.

Bochkovskiy *et al.* have propelled the YOLOv4 [52] model forward by efficiently scaling the network design and scale, surpassing the previous state of the art EfficientDet [53]. Bochkovskiy *et al.* scale the YOLO model up and down, beating prior benchmarks from previous small and large object detection models on both ends of the speed versus accuracy frontier. In general, the authors of the Scaled-YOLOv4 are holding a few scaling concepts in balance as they are working on the construction of their models - number of layers, number of channels and image size, while optimizing for inference speed and model performance. The YOLOv4-tiny model had different applications than the Scaled-YOLOv4 model because different constraints are used, memory bandwidth and memory access are two of them. To detect large objects in large images, the authors concluded that it is important to increase the depth and number of stages in the Convolutional Neural Network (CNN) backbone and neck. This allows them to first scale up the input size and number of stages, and dynamically adjust width and depth according to real time inference speed requirements. In addition to these scaling factors, the authors also adjust the configuration of the model architecture in the paper.

YOLOv5[1] was released very shortly after YOLOv4. Despite its name, the authors are not related neither from the same instituition, and there have been controversy on whether it is fair to call YOLOv5 a successor of YOLOv4. This implementation shares the same design and provides similar performance to YOLOv4. The main point of attention is the fact that it is fully written in the PyTorch framework, as opposed to using any form of the Darknet framework, and it is significantly smaller, faster to train and more acessible to use in a wider range of development environments. Additionally, the models in YOLOv5 prove to be significantly

[1]Ultralytics YOLOv5 github repository (last seen 21/02/2022): `https://github.com/ultralytics/yolov5`

smaller, faster to train and more accessible to be used in real-world applications.

V. Methodology and Results

A. Vegetation semantic segmentation with DeepLabv3+

In this section are presented the results of semantic vegetation segmentation using DeepLabv3+s.

It was used an already trained DeepLabv3+ model, trained with the CityScapes dataset with pre-trained MobileNet backbone, for vegetation segmentation in wildland-urban interfaces areas. Figure 1 show the source images (on the left), the resulting vegetation segmentation of those source images using DeepLabv3+ (on the right) and also the color used in the output of the DeepLabv3+ semantic segmentation method (bottom). Figure 1 RGB source images were obtained in a forest near the Department of Electrical and Computer Engineering of the University of Coimbra.

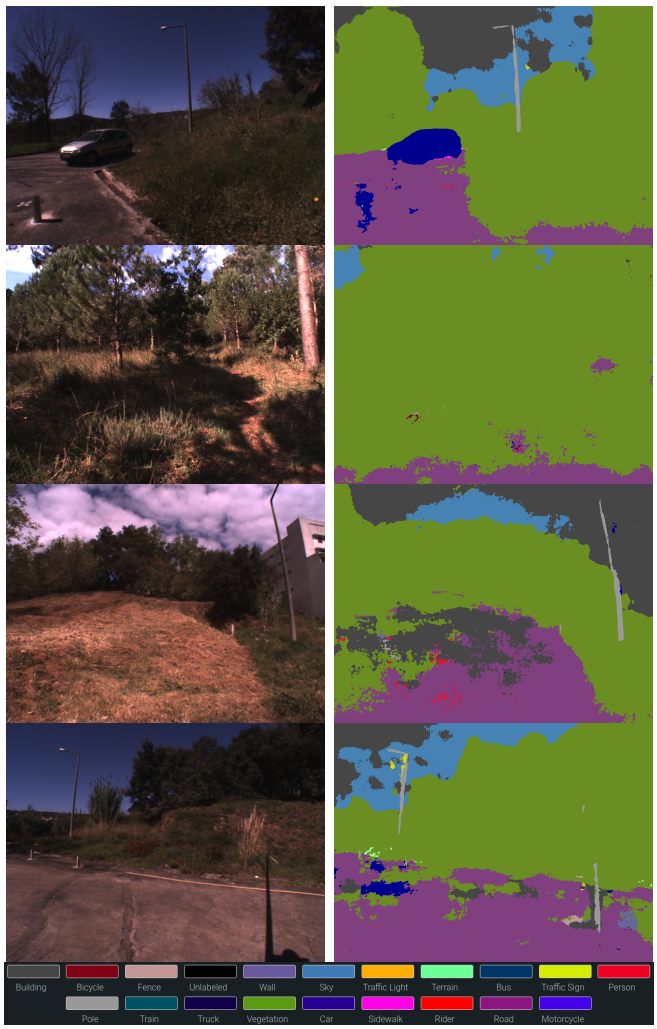

Fig. 1. Segmentation using DeepLabv3+. Source RGB images at left and DeepLabv3+ segmentation results at right. At the bottom is presented the color used for each class.

B. Vegetation detection with YOLOv5

In this section are presented the results of the vegetation detection using YOLOv5.

A YOLOv5 network was trained for vegetation detection in a forest environment for a thousand epochs using Google Colab[2]. It were used two hundred images for training (one hundred and fifty for training, thirty for validation and twenty for testing) and this training images were labeled with five classes (using RoboFlow label tool[3]): "Live Vegetation", "Grass", "Cutted-Vegetation", "Tree-trunk" and "Dead Vegetation". The training dataset was acquired in the same forest near the Department of Electrical and Computer Engineering of the University of Coimbra.

In Figures 2, 3, 4 and 5 it is presented the mean average precision at 0.5 and 0.95 Intersection Over Union (IOU), precision, recall, bounding box loss, class loss and object loss that resulted from the YOLOv5 training for a thousand epochs. These metrics are obtained using the validation set from the training dataset.

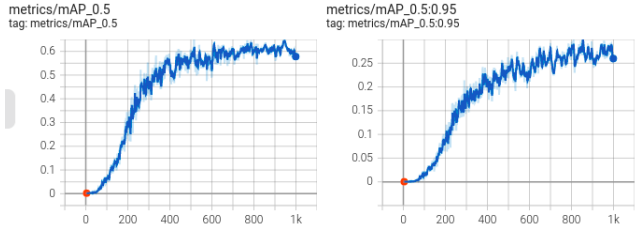

Fig. 2. YOLOv5 model training metrics mean average precision at 0.5 IOU (left) and at 0.95 IOU (right) during the 1000 epochs.

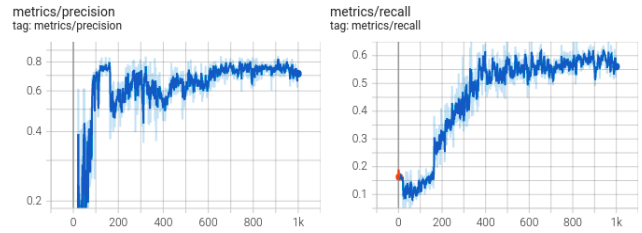

Fig. 3. YOLOv5 model training metrics precision (left) and recall (right) during the 1000 epochs.

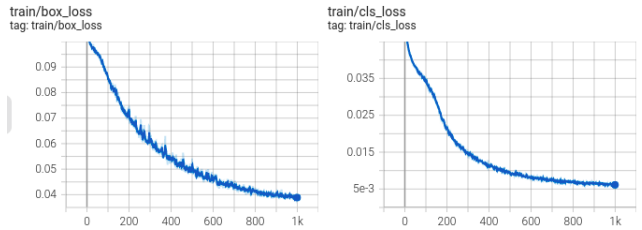

Fig. 4. YOLOv5 model training metrics bounding box loss (left) and the class loss (right) during the 1000 epochs.

Figures 6 show the RGB test images (at left) and the resulting object detection using the trained YOLOv5 model

[2]Google Colab website (last seen 22/02/2022): https://colab.research.google.com/

[3]RoboFlow website (last seen 22/02/2022): https://roboflow.com/

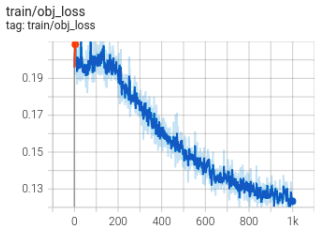

Fig. 5. YOLOv5 metric object loss evolution during the 1000 epochs training.

(at right). In the first row of Figure 6 the maximum confidence of the detection of live vegetation is 95 percent, the maximum confidence of the detection of grass is 92 percent and the maximum confidence of the detection of tree trunk is 88 percent. In the third row of Figure 6 the maximum confidence of the detection of cut vegetation is 40 percent, the maximum confidence of the detection of live vegetation is 92 percent and the maximum confidence of the detection of grass is 89 percent.

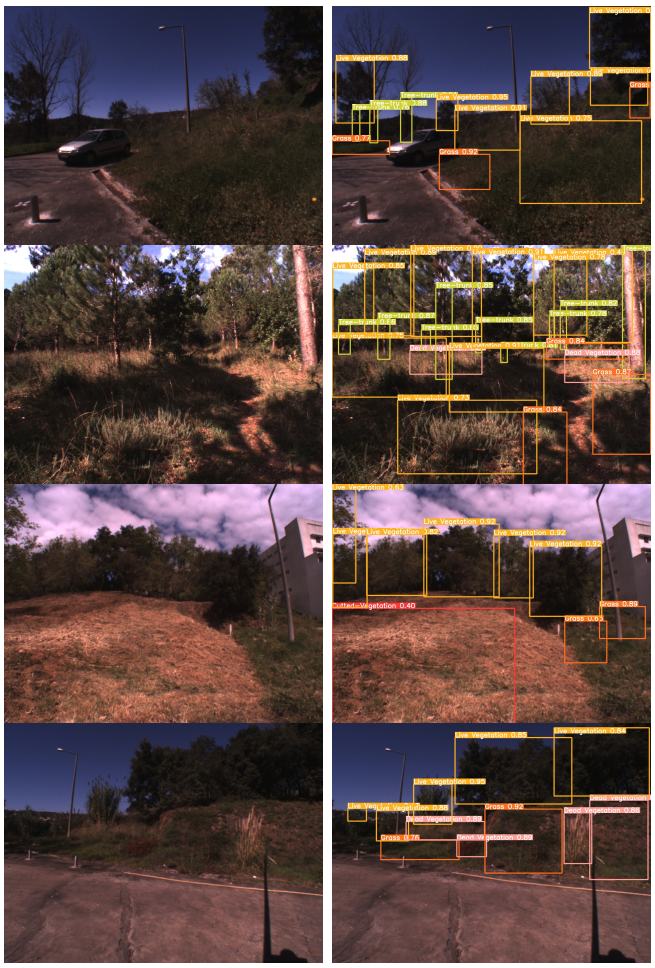

Fig. 6. Segmentation using the trained YOLOv5 model. Source RGB images at left and obtained results at right.

TABLE I

PROCESSING TIME OF DEEPLABV3+ AND YOLOV5 MODELS FOR IMAGES WITH 1440×1080 PIXELS.

	image 1	image 2	image 3	image 4
DLV3+ (s)	4.860	4.720	4.730	4.710
YOLOv5 (s)	0.215	0.230	0.225	0.251

C. Computation times

In table I is presented the computation times of the YOLOv5 and DeepLabv3+ models for vegetation detection. The four source images shown, in Figures 1 and 6, have a 1440×1080 pixels size. The computation times were obtained with a laptop with a Intel Core i7 6th Gen 6500U processor and 8 GB of RAM. The processing was made using the CPU.

VI. CONCLUSIONS

Semantic segmentation and object detection was developed for the end of vegetation detection and classification for an Unmanned Ground Vehicle (UGV) to clean forest fuel to prevent fires. DeepLabv3+ pre-trained with the dataset CityScapes was used for semantic segmentation of vegetation in RGB images acquired in a forest environment. It results in effective detection of green vegetation, however, the dead/dry vegetation segmentation results in misclassifications. A YOLOv5 model for object detection was also used for vegetation detection and classification in forest images and it results in an effective detection and classification of vegetation. Unlike the deepLabv3+ pre-trained model, the YOLOv5 model was effective in detecting live and dead/dry vegetation. Since the detection of dead/dry vegetation is effective using the YOLOv5 method, this deep learning approach will be used by the UGV to perform the detection, classification and localization of vegetation in forest environments. YOLOv5 also has clear advantages in terms of training speed and the size of the weight file. These advantages made YOLOv5 more suitable for the detection of vegetation. The resulting processing times show real-time processing capabilities for YOLOv5 object detection, what is a step forward to the task of forest fuel cleansing for cluttered forest environments using autonomous robots.

ACKNOWLEDGMENT

This research is supported by the Programa Operacional Regional do Centro, Portugal 2020, European Union through the FEDER Fund, through the project with the Operation Code: CENTRO-01-0247-FEDER-045931.

REFERENCES

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–I.

[2] ——, "Robust real-time face detection," in *International Journal of Computer Vision 57*, vol. 1, 2004, pp. 137–154.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 886–893 vol. 1.

[4] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157 vol.2.

[5] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[6] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *2008 IEEE conference on computer vision and pattern recognition*. Ieee, 2008, pp. 1–8.

[7] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," *Advances in neural information processing systems*, vol. 15, 2002.

[8] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *2010 IEEE Computer society conference on computer vision and pattern recognition*. Ieee, 2010, pp. 2241–2248.

[9] R. B. Girshick, *From rigid templates to grammars: Object detection with structured models*. The University of Chicago, 2012.

[10] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *2011 international conference on computer vision*. IEEE, 2011, pp. 1879–1886.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[12] 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.

[13] 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.

[14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[15] Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler, "segdeepm: Exploiting segmentation and context in deep neural networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4703–4711.

[16] 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.

[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, 2015.

[19] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[20] 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.

[21] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[22] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[23] 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.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[25] X. Tao, D. Zhang, W. Ma, X. Liu, and D. Xu, "Automatic metallic surface defect detection and recognition with convolutional neural networks," *Applied Sciences*, vol. 8, no. 9, 2018. [Online]. Available: https://www.mdpi.com/2076-3417/8/9/1575

[26] X. Zhu, H.-I. Suk, S.-W. Lee, and D. Shen, "Subspace regularized sparse multitask learning for multiclass neurodegenerative disease identification," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 607–618, 2016.

[27] Z. Wang, L. Wei, L. Wang, Y. Gao, W. Chen, and D. Shen, "Hierarchical vertex regression-based segmentation of head and neck ct images for radiotherapy planning," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 923–937, 2018.

[28] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, and A. M. Lopez, "Vision-based offline-online perception paradigm for autonomous driving," in *2015 IEEE Winter Conference on Applications of Computer Vision*, 2015, pp. 231–238.

[29] G. Brazil, X. Yin, and X. Liu, "Illuminating pedestrians via simultaneous detection and segmentation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*, vol. 25, 2012.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[34] A. Graves, S. Fernández, and J. Schmidhuber, "Multi-dimensional recurrent neural networks," in *Artificial Neural Networks – ICANN 2007*, J. M. de Sá, L. A. Alexandre, W. Duch, and D. Mandic, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 549–558.

[35] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, and Y. Bengio, "Renet: A recurrent neural network based alternative to convolutional networks," July 2015.

[36] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11.

[37] Q. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[38] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-scnn: Gated shape cnns for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[39] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European conference on computer vision*. Springer, 2012, pp. 746–760.

[40] M. Ouloul, Z. Moutakki, K. Afdel, and A. Amghar, "An efficient face recognition using sift descriptor in rgb-d images," *International Journal of Electrical and Computer Engineering*, vol. 5, no. 6, 2015.

[41] P. Sykora, P. Kamencay, and R. Hudec, "Comparison of sift and surf methods for use on hand gesture recognition based on depth map," *Aasri Procedia*, vol. 9, pp. 19–24, 2014.

[42] A. Hermans, G. Floros, and B. Leibe, "Dense 3d semantic mapping of indoor scenes from rgb-d images," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2631–2638.

[43] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[44] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[45] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[46] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[47] M. Cordts, M. Omran, S. Ramos, T. Scharwächter, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes

dataset," in *CVPR Workshop on The Future of Datasets in Vision*, 2015.

[48] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[49] ——, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[50] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[51] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[52] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[53] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.