Detection of Norway Spruce Trees (Picea Abies) Infested by Bark Beetle in UAV Images using YOLOs Architectures

ANASTASIIA SAFONOVA1,2, YOUSIF HAMAD2,3, ANNA ALEKHINA2, and DMITRY KAPLUN3

1Department of Automation and Control Processes, Saint Petersburg Electrotechnical University “LETI”, 197376, Saint Petersburg, Russia
2Department of Artificial Intelligence Systems, Siberian Federal University, 660074, Krasnoyarsk, Russia
3College of Information Technology, Imam Ja’afar Al-Sadiq University, Kirkuk, 661001, Iraq

Corresponding author: Anastasiia Safonova (e-mail: safonova.nastyal@gmail.com).

This work was supported by the Russian Foundation for Basic Research (RFBR) with projects No. 19-01-00215 and No. 20-07-00370, and by the Ministry of Science and Higher Education of the Russian Federation under assignment No. FSEE-2020-0002.

ABSTRACT In recent years, massive outbreaks of the European spruce bark beetle (Ips typographus, (L.)) have caused colossal harm to coniferous forests. The main solution for this problem is the timely prevention of the bark beetle spread, for which it is necessary to identify damaged trees in their early stages of infestation. Fortunately, high-resolution unmanned aerial vehicle (UAV) imagery together with modern detection models provide a high potential for addressing such issues. In this work, we evaluate and compare three You Only Look Once (YOLO) deep neural network architectures, namely YOLOv2, YOLOv3, and YOLOv4, in the task of detecting infested trees in UAV images. We built a new dataset for training and testing these models and used a pre-processing balance contrast enhancement technique (BCET) that improves the generalization capacity of the models. Our experiments show that YOLOv4 achieves particularly good results when applying the BCET pre-processing. The best test result when comparing YOLO models was obtained for YOLOv4 with the mean average precision up to 95%. As a result of applying artificial data augmentation, the improvement for models YOLOv2, YOLOv3, and YOLOv4 was 65%, 7.22%, and 3.19%, respectively.

INDEX TERMS Norway spruce, bark beetle, unmanned aerial vehicle (UAV), you only look once (YOLO), object detection.

I. INTRODUCTION

Preserving natural forests is essential for the environment as they play a very important role in the global ecosystem. Unfortunately, there are several factors that threaten the well-being of forests. One of them is different pests that can attack trees leading to their weakening or even death. In particular, the European spruce bark beetle (Ips typographus, (L.)) [1–3] is widespread in the coniferous, mainly spruce forests of Eurasia (Sweden, Finland, Denmark, Germany, Bulgaria and other). This beetle belongs to the class of especially dangerous forest pests. Usually, a small population of bark beetles attack predominantly weakened trees. The flight of beetles begins in the spring when the average temperature is about 18 °C. Infestation of Norway spruce trees is accompanied by the mass reproduction of bark beetles that can last 4-5 years, in rare cases up to 9-12 years [4], [5]. There has been an increase in mortality in European forests from beetle attacks of more than 8% from 1850 to the present, mainly due to increased temperatures and the increased frequency of droughts. In addition, during the outbreaks bark beetles attack even healthy trees, leading to mass extinction of the forest [6], [7]. Since the bark beetles cause significant damage to forestry and forest parks, it is important to develop a program for the reduction of their quantity. The first step of this program should be forest monitoring. One of the most effective ways of forest conservation monitoring is the use of remote sensing (RS) data.

The choice of methods for conducting timely analysis of forest monitoring data is also important. Usually, monitoring of the infested forest is carried out during field research. However, due to the high level of outbreaks, it is impossible to carry out timely monitoring of vast areas of the forest. The large size of forests and hard-to-reach places under study do
not allow the use of pure manual analysis, such as manual counting and individual detection of trees. Thus, to solve this problem, it is advisable to use Earth remote sensing tools for timely monitoring of the forest at different stages of its infestation. Since the advent of unmanned aerial vehicles (UAVs) in mass production, it has become possible for scientists to collect a large amount of local area imagery with very high spatial resolution, but the problem of processing large arrays of information has remained. In particular, the resource-intensive processing of a large amount of unsorted RS data obtained from drones can include an expert opinion on the type of tree and manual counting of specimens with a particular state of crown integrity. However, recently it has been shown that such problems can be effectively solved with the use of deep neural networks (DNNs) [8]–[10] (see Section II).

The goal of this work is the detection of infested trees in images obtained from UAVs, by using YOLO architectures that employ convolutional neural networks (CNN) [11]–[15]. First, we prepared a dataset for training and testing YOLO architectures from the orthophoto images obtained with UAV. In the next step, we applied a pre-processing procedure to the dataset. This process consists in increasing the contrast of the input image, which makes it possible to increase the accuracy of detecting individual tree crowns. Next, we trained and tested the YOLO architectures, using versions from 2 to 4. We then presented the results of comparing these architectures and determined the best YOLO architecture for the task of detecting infested trees.

The main contributions of this paper can be listed as follows:

- We have built a new dataset for the detection of bark beetle infestation in Norway spruce trees to detect four categories of Siberian fir trees (Abies sibirica, (L.)). The dataset is organized into 400 images of Red-Green-Blue (RGB) color images having 3.75 cm/pixel spatial resolution.
- We have created a new annotator to quickly annotate our dataset called Visual Object Labeller 1.3.
- We have applied a pre-processing balance contrast enhancement technique (BCET) for improving the quality of the images and hence for increasing the generalization capacity of the detection model.
- We have evaluated the YOLO object detection architectures with versions 2, 3 and 4 for the task of infested tree detection in UAV images.

The paper is organized as follows: The related works are given in Section II. The materials and methods are presented in Section III, where the study area is shown in Subsection III.A, the pre-processing methodology is provided in Subsection III.B, the annotator of images is described in Subsection III.C, the YOLO models are evaluated and compared in Subsection III.D, the experimental setup is described in Subsection III.E, and the evaluation metric is in Subsection III.F. The experimental Results are presented in Section IV. The paper concludes with Section V. This article is based on Chapter 3 of the Anastasia Safonova’s Ph.D. thesis [16].

II. RELATED WORKS

In this section, we consider related works devoted to the problems of tree detection on the Earth’s surface data using deep learning (DL). Object detection is the task of localizing all specified objects of a class and building a bounding box for each of them. Intelligent systems based on NNs can successfully solve problems of plant recognition [17]. There are other, more traditional approaches to solving these problems, but they do not have the required flexibility outside of limited conditions [18]. NNs provide promising alternative solutions, and many applications benefit from their use.

Below are the works on solving tasks of object detection in the forestry industry on RS data by DNN, which are the closest to our problem. In the work [19], using a pre-trained YOLO architecture, authors were able to obtain an average accuracy of up to 91.82% in the detection of affected pine trees on UAV-obtained very high-resolution images. Thanks to this solution, it became possible to localize the affected tree on various aerial images. In another work [20], the authors considered a similar problem of detecting a dead pine tree on UAV data using CNNs of AlexNet and GoogLeNet with a maximum accuracy of up to 97.38%. Tao et al. in [21] performed the insect-damaged tree detection (dead fir, sick fir, healthy fir, deciduous trees, grass, and uncovered) with DJI Mavic 2 Pro quadcopter data and DL technique based on CNNs.

In our previous work [10], we were able to detect four categories of Siberian fir trees (Abies sibirica, (L.)) damaged by the Polygraphus Proximus Blandford bark beetle in UAV images using DL. We solved the problem of detecting individual trees on UAV images using our own algorithm. Then we classified the detected patches as belonging to one of the four categories of the tree using an approach by Krivets et al. [22]: healthy tree or recently attacked by beetles, tree colonized by beetles, recently died tree, and deadwood. In the last step, we proposed a new CNN architecture specially designed to solve that problem. We also presented a comparison of the developed architecture with such models as VGG, ResNet, Inception-V3, InceptionResNet-V2, Xception, and DenseNet. It is important to note that our model, trained with data augmentation, showed up to 98.77% accuracy for the next categories of fir trees: healthy tree, recently dead tree, and deadwood. Our next work was about individual tree crown delineation for the species classification and assessment of vital status of forest stands from UAV images. In this work, we proposed an approach to enhance algorithms for species

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
classification and assessment of the vital status of forest stands by using automated individual tree crowns delineation (ITCD). The performance of the ITCD algorithm was demonstrated for different test plots containing homogeneous and complex structured forest stands. The pixel-by-pixel classification is based on the ensemble supervised classification method error-correcting output codes with the Gaussian kernel support vector machine chosen as a binary learner.

We demonstrated that pixel-by-pixel species classification of multi-spectral images can be performed with a total error of about 1%, which is significantly less than that obtained by the processing of RGB images. For typical scenes, the crown contouring accuracy is about 95%. The advantage of the proposed approach lies in the combined processing of multispectral and RGB photo images from DJI quadcopters [23].

This experiment is complementary to our latest research [23]. In this paper, we analyze the detection of four infestation classes of Norway spruce trees attacked by the European bark beetle (green-, yellow-, red-, and gray-attack (see Section III.A)) on UAV images using YOLO architectures. Then we compare the quality of the trained architectures on the test dataset.

III. MATERIALS AND METHODS

A. STUDY AREA

The study area is the territory of the West Balkan Mountains (up to 1300–1500 m altitude), the south of the administrative center of Chuprene, Vidin Province, Bulgaria (Figure 1). We considered study area located in protected place of Chuprene Biosphere Reserve, included in the list of UNESCO [23]. The reserve has been created with the aim of preserving the environment of the unique natural complex. Regular and detailed remote monitoring is required to take time management actions aimed at solving problems associated with damage to forest stands by stem pests and changes in stand structure parameters.

We used RS data obtained from a DJI-Phantom 4 Pro UAV with an RGB (red, green, and blue channels) camera with a resolution of 3.75 cm/pixel. The drone flew several times on August 16, 2017, and September 25, 2017, at the maximum permitted altitude of 120 meters above the ground. The object of the study is natural forests damaged as a result of attacks by European spruce bark beetles (Ips typographus, (L.)) [24]. Forests mainly consist of Norway spruce (Picea abies, (L.) Karst.), European beech (Fagus sylvatica, (L.)), Scotch pine (Pinus sylvestris, (L.)), and Black pine (Pinus nigra, (L.)). However, in our experiments, the object of research predominantly is damaged Norway spruce trees (Picea abies, (L.) Karst.). Infested by bark beetles tree can be classified based on visible crown symptoms in four stages of attack. The initial attack is not visible to the human eye (green-attack), and a tree of this class cannot be visually (looking only on its crown) differentiated from a healthy one. With thousands of attacking beetles per one Norway spruce tree, needles first turn yellow (yellow-attack), then to reddish brown (red-attack), and finally to grey (dead tree) [25–27]. Verification of infested trees on images was done by the Bulgarian experts in the field of forest entomology and phytopathology prof. DSc Georgi Georgiev and his colleagues in accordance with the methodical manual of assessment of crown conditions of ICP “Forests” [28].

The image dataset has been prepared using QGIS software (Quantum GIS v. 2.14.21). RGB images are represented by unsigned integers from 0 to 255 in natural colors. We built a dataset made of 400 images, where 80% were for training, 20% for validation, and two plots A and B were for external testing (Table 1).
TABLE 1. Training and validating datasets, where a is green-attack, b is yellow-attack, c is red-attack, and d is grey-attack.

| # of training images | # of training trees | # of validating images | # of validating trees | Total of images | Total of trees |
|----------------------|---------------------|------------------------|-----------------------|-----------------|---------------|
|                      | a | b | c | d | a | b | c | d |                     |               |
| 320                  | 312 | 622 | 76 | 188 | 80 | 202 | 400 | 61 | 400                  | 1881          |

We pre-processed the original dataset using pixel contrast enhancement. During the training process, we applied dynamic data augmentation using standard functions: rotation, horizontal flip, vertical flip, resizing. It should also be noted that the number of damaged trees varies from class to class, as in the experiment we used data from the natural forest. Thus, the total number of trees obtained from the training and validating datasets per class is as follows: a—green-attack (594 trees), b—yellow-attack (1206 trees), c—red-attack (104 trees), and d—grey-attack (277 trees). The resulting dataset is unbalanced as these are trees from natural forest. The total number of trees in the test plot A is 38, where 12, 23, 1, and 2 correspond to the classes of a, b, c, and d, respectively. The total number of trees in the test plot B is 51, where 12, 34, 3, and 2 correspond to the classes of a, b, c, and d.

B. PRE-PROCESSING METHODOLOGY

UAVs are an affordable and widely used resource for local monitoring of plants. However, in practice, the quality of the images obtained from an UAV can be deteriorated due to the suspended particles in the atmosphere, weather conditions, and the quality of the equipment itself. This results in low contrast images [29]. In this work, to improve the quality of the images, we used a balance contrast enhancement technique (BCET) [30]. This technique was previously successfully applied by us in our previous work [23].

The contrast of the image can be stretched or compressed without changing the histogram pattern of the input image. For this, we can use a parabolic function in the general form whose coefficients depend on the input image:

\[ I_{\text{New}} = m \times (I_{\text{Old}} - n) + k, \]

where \( I_{\text{Old}} \) is the intensity of the input image (of one of the three-color channels), and \( I_{\text{New}} \) is the intensity of the output image. The coefficients \( m, n, \) and \( k \) are calculated using the minimum and maximum intensity values of the input (\( l \) and \( h \)) and output (\( L \) and \( H \)) image and the intensity mean value of the input (\( e \)) and output (\( E \)) image:

\[
m = \frac{H - L}{(h - l)(h + l - 2n)},
\]

\[
n = \frac{h^2 \cdot (E - L) - s \cdot (H - L) + l^2 \cdot (H - E)}{2 \left( h \cdot (E - L) - s \cdot (H - L) + l \cdot (H - E) \right)},
\]

\[
k = L - m \cdot (l - n)^2,
\]

where \( s \) denotes the intensity mean square sum of the input image (5):

\[
s = \frac{1}{N} \sum_{i=1}^{N} 2^{2} I_{\text{Old}}(i),
\]

where the summation is taken over all image pixels with the total number of \( N \). Note that the target values of parameters \( L, H, \) and \( E \) are set manually. The example of the quality improvement for one image is shown in Figure 2.

![Figure 2](image_url)  
**FIGURE 2.** Example of pre-processing, where (a) the original image, (b) result of balance contrast enhancement; histogram of input (c) and output (d) image.

A clearer visual separation of the individual tree crowns can be observed in the processed image. Figure 2 shows the histogram of the processed image. It indicates that the mean values of all the RGB channels are balanced from 0 to 255, while the mean values of the channels of the source image are from \(-90\) to \(-200\). This process enhances image quality allowing it to get better and more accurate results. A histogram value of the RGB channels after applying a contrast balance enhancement shows that the data has been spread over much more of the available dynamic range. Thus, as a result of the application of preprocessing, the crowns of damaged trees are more clearly distinguished in the image.
C. AN ANNOTATOR OF DATASET

For the convenience and acceleration of data annotation for this work, a new lightweight version of the software for annotation according to the YOLO standard was developed to label objects on images [31]. The proposed annotator does not require additional graphic libraries and has a convenient interface allowing for fast annotation of a large number of images in a few steps. The graphical interface of software called Visual Object Labeller 1.3 is shown in Figure 3.

FIGURE 3. Graphical user interface of the Visual Object Labeller version 1.3 annotator with main menu—1, work window—2, file location—3, name and file type—4, selected file—5, total number of files—6, navigation between files—7, assigned class—8, list of labelled points—9.

Our version does not require the OpenCV library installed and uses only the standard graphics functions of the JAVA programming language. The basic steps for using the Visual Object Labeller 1.3 to annotate images are presented in Figure 4.

FIGURE 4. Scheme of operation of the Visual Object Labeller 1.3 annotator.

The text labelling of the areas on the image has a form:

\[ < \text{class} > < X_{YOLO} > < Y_{YOLO} > < W_{YOLO} > < H_{YOLO} >, \]  \hspace{1cm} (6)

where \(<\text{class}>> is an integer class number, in the present work it corresponds to infested trees classes \(a, b, c,\) and \(d.\) \(<X_{YOLO}>, <Y_{YOLO}>, <W_{YOLO}>,\) and \(<H_{YOLO}>> are the reduced coordinates and width and height of a labelled area, calculated as:

\[ X_{YOLO} = \frac{X_{box} + W_{box} \times 0.5}{W_{image}}, \]  \hspace{1cm} (7)

\[ Y_{YOLO} = \frac{Y_{box} + H_{box} \times 0.5}{H_{image}}, \]  \hspace{1cm} (8)

\[ W_{YOLO} = \frac{W_{box}}{W_{image}}, \]  \hspace{1cm} (9)

\[ H_{YOLO} = \frac{H_{box}}{H_{image}}, \]  \hspace{1cm} (10)

where \(X_{box}\) and \(Y_{box}\) are the coordinates of the upper-left corner of the rectangle, \(W_{box}\) is width and \(H_{box}\) is height of the rectangle, and \(W_{image}\) is width and \(H_{image}\) is height of the image.

VOLUME XX, 2021

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
D. YOLO ARCHITECTURES

Object detection is a fundamental computer vision task in which the algorithm analyzes the input image and outputs a label together with a bounding box that delimits where the object-class is in the image [9]. Currently, YOLO models are considered one of the most accurate deep CNN-based methods, since these have optimal speed and work well independently on small datasets in real-time. In this work we used three YOLO architectures, namely YOLOv2 [12], YOLOv3 [13], and YOLOv4 [14], to compare them and analyze the impact of image preprocessing on the detection. It should be noted that the first version of the YOLO architecture was not included in the comparison, since the first version contains a number of errors and cannot detect small images which we used in the experiment [11].

Most object detection algorithms take in and process the image multiple times to be able to detect all the objects present in the images. But YOLO looks at the object only once. It applies a single forward pass to the whole image and predicts the bounding boxes and their class probabilities. The architecture consists of two major components: feature extractor and feature detector or multi-scale detector. The image is first given to the feature extractor which extracts feature embeddings and then is passed on to the feature detector part of the network that produces the processed image with bounding boxes around the detected classes.

If we compare the selected architectures, then YOLOv1 with the Darknet-19 NN results in more localization errors but is much less likely to predict false positives when searched objects do not present in the data. It outperforms other detection methods, including deformable part models (DPM) and Region-based CNN (R-CNN). However, despite the improvements in the YOLOv2 version on Darknet-30 networks, it has better results of detection, but also has a problem while detecting small objects due to down-sampling the input image and losing fine-grained features. The improved architecture of YOLOv3 with Darknet-53 and ResNet networks due to its complexity is a bit slower compared to YOLOv2, but at the same time, gives results with higher accuracy. The improved YOLOv4 architecture, in contrast to the previous version, works much faster without loss of definition quality.

E. EXPERIMENTAL SETUP

All models were trained and tested on an Ubuntu 16.04.6 LTS operating system with an NVIDIA GeForce GTX 1060 graphics processing unit (GPU) and CUDA 10.1 parallel computing platform. The input images were cropped to 416x416 pixels to fit the input layer of the training model. We used a global learning rate of 0.001, and four classes have 8000 iterations for maximum batches. Standardized augmentation techniques were used during training where decay is 0.0005, saturation is 1.5, exposure is 1.5.

The loss function [11] for each of the YOLO models was calculated as follows (11):

\[
\begin{align*}
\lambda_{\text{coord}} & = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \frac{\text{obj}_{ij}}{n_{ij}} \left[ \left( x_i - \hat{x}_i \right)^2 + \left( y_i - \hat{y}_i \right)^2 \right] + \\
\lambda_{\text{coord}}^{\text{noobj}} & = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \frac{\text{noobj}_{ij}}{n_{ij}} \left( \left( x_i - \hat{x}_i \right)^2 + \left( y_i - \hat{y}_i \right)^2 \right) + \\
\sum_{i=0}^{S^2} \sum_{j=0}^{B} \frac{\text{obj}_{ij}}{n_{ij}} \left( \left( \sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left( \sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right) + \\
\sum_{i=0}^{S^2} \sum_{j=0}^{B} \frac{\text{noobj}_{ij}}{n_{ij}} \left( \left( \sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left( \sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right) + \\
\sum_{j=0}^{B} \sum_{i=0}^{S^2} \frac{\text{obj}_{ij}}{n_{ij}} \left( \left( \hat{c}_i - c_i \right)^2 + \left( \hat{c}_j - c_j \right)^2 \right) + \\
\sum_{j=0}^{B} \sum_{i=0}^{S^2} \frac{\text{noobj}_{ij}}{n_{ij}} \left( \left( \hat{c}_i - c_i \right)^2 + \left( \hat{c}_j - c_j \right)^2 \right) + \\
\sum_{j=0}^{B} \sum_{i=0}^{S^2} \frac{\text{obj}_{ij}}{n_{ij}} \left( \left( \hat{p}_i - p_i \right)^2 + \left( \hat{p}_j - p_j \right)^2 \right)
\end{align*}
\]

where \( S^2 \) is the output feature map of all grid cells, \( B \) is the number of bounding box for each grid, \( i \) is the \( i \)-th grid, \( j \) is the \( j \)-th predicted box of this grid, \( x_i \) and \( y_i \) are the location of the centroid of the anchor box, \( w_i \) and \( h_i \) are the width and height of the anchor of the box, \( C_i \) is the Objectness which means a measure of the probability that an object exists in a proposed region of interest, \( p_i(c) \) is the probability of real box class, \( \hat{p}_i(c) \) is the probability of predict box class, \( \frac{\text{obj}_{ij}}{n_{ij}} \) denotes if object appears in cell \( i \), \( \frac{\text{noobj}_{ij}}{n_{ij}} \) defines whether the \( j \)-th box in the \( i \)-th grid is responsible for that prediction, and \( \lambda_{\text{coord}} \) and \( \lambda_{\text{noobj}} \) are weighting factors.

The training time for each model was: YOLOv2 - 3 hours, YOLOv3 - 12 hours, YOLOv4 - 6 hours on original images, and YOLOv2 - 4 hours, YOLOv3 - 15 hours, YOLOv4 - 5 hours on pre-processed images.

For detection of infested spruce trees, we trained models on two datasets: without and with data augmentation. The trained models were independently tested on two test plots A and B.

F. EVALUATION METRICS

To evaluate the performance of the trained YOLO architectures in the task of detection of infested spruce trees, we used the mean average precision (mAP) and intersection over union (IoU) metrics. These are popular metrics in measuring the accuracy of object detectors. IoU is a simple scoring metric that requires the following data to apply as:

- The ground-truth labelled bounding boxes, which are created manually.
- The predicted bounding boxes from the trained model:

\[
\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}
\]

where overlap and union operations are applied to the ground-truth and corresponding predicted areas.

IoU is used to measure the overlap of the boundaries between ground-truth labels and predicted labels from the
trained model. If the detection is absolutely correct, then the indicator is equal to 1. The lower the value of the IoU metric, the worse the prediction result. Usually, the threshold of this indicator is set equal to 0.5, meaning that if the IoU > 0.5, then the prediction is considered as correct (true) and false otherwise. Based on this, calculations are made for indicators such as Precision (13) and Recall (14):

\[
\text{Precision} = \frac{TP}{TP + FP}, \quad (13)
\]

\[
\text{Recall} = \frac{TP}{TP + FN}, \quad (14)
\]

where \(TP\) is a true positive prediction, \(FP\) is a false positive prediction, and \(FN\) is a false negative prediction.

Precision determines the percentage of correctly recognized labels and Recall shows how good true positives were predicted. On these criteria, \(F1\)-score and \(m\text{AP}\) are calculated to evaluate the performance of a model. \(F1\)-score is calculated from the precision and the recall, and \(m\text{AP}\) is the average precision (the area under the Precision-Recall curve) averaged over all classes [14]:

\[
F1\_\text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (15)
\]

\[
m\text{AP} = \sum_{i=1}^{N} \frac{\text{AP}_i}{N} = \frac{1}{N} \sum_{i} \text{Precision} \left( \text{Recall}_i \right), \quad (16)
\]

where \(N\) is the number of classes.

IV. EXPERIMENTAL RESULTS

In this section, we present the results of our experiments. The purpose of this experiment was to test and compare YOLOv2 [12], YOLOv3 [13], and YOLOv4 [14] architectures on images captured from UAV. The main task is detecting infested trees belonging to one of four classes: a—green-attack, b—yellow-attack, c—red-attack, and d—grey-attack. The models were trained on the original images and pre-processed images, with data-augmentation, and tested on two test plots, A and B.

The results of training all YOLO architectures are presented in Table 2.

We used two pre-processed test plots A and B to test the trained YOLO architectures. The total number of trees in test plot A was 38, where 12, 23, 1, and 2 correspond to classes \(a, b, c, \) and \(d\). The total number of trees in test plot B was 51, where 12, 34, 3, and 2 correspond to classes \(a, b, c, \) and \(d\).

Table 3 shows the results of the predictions of YOLO models for each tree infestation class, presented next to the samples labelled by an expert (ground truth).

It can be noted that classes \(a \) and \(b \) are best detected by the YOLOv4 architecture on both datasets. The best detection of class \(c \) on data with and without pre-processing is provided by the YOLOv2 architecture. The class \(d \) is better detected on the original data by the YOLOv3 architecture, and on data with pre-processing it is better detected by the YOLOv3 and YOLOv4 architectures. In general, YOLOv4 showed the best result in detecting infested trees, as it showed fewer false positives and more correct answers. A graphical presentation of the results of Table 3 is shown in Figure 5. Table 4 presents the performance results of testing YOLO architectures.
The results presented in Table 4 show that the use of preprocessed images for training can increase the accuracy for each of the architectures with the degree of the improvement of the results depending on the tree class. Contrast enhancement results in the improvement for YOLOv2 and YOLOv3 in the detection of red-attack (c) and grey-attack (d) classes, and for YOLOv4 of yellow- (b), red- (c), and grey-attacks (d). In general, after applying contrast enhancement the accuracy of classification increased significantly for all considered architectures and there were fewer false positive and false negative.

It should be noted that testing the test areas took a different amount of time for each version of the YOLO. For example, the processing of images with a size of 1280×720 pixels using the CUDA 10.1 and NVIDIA GeForce GTX 1060 GPU on the trained YOLOv2 architecture took about 22.22 microseconds, and for the trained YOLOv3 and YOLOv4 architectures with the same image it was 31.25 milliseconds and 28.57 milliseconds, respectively.

### TABLE 4. Average results of the three YOLO architectures performance on the external A and B plots.

| Performance | YOLOv2 Original images | Pre-processed images | YOLOv3 Original images | Pre-processed images | YOLOv4 Original images | Pre-processed images |
|-------------|------------------------|---------------------|------------------------|---------------------|------------------------|---------------------|
| IoU         | 0.40                   | 0.76                | 0.68                   | 0.74                | 0.71                   | 0.73                |
| Precision   | 0.70                   | 0.90                | 0.93                   | 0.96                | 0.94                   | 0.95                |
| Recall      | 0.56                   | 0.96                | 0.48                   | 0.91                | 0.72                   | 0.76                |
| mAP         | 0.32                   | 0.92                | 0.90                   | 0.97                | 0.91                   | 0.94                |
| Loss        | 0.62                   | 0.93                | 0.63                   | 0.93                | 0.81                   | 0.84                |
V. CONCLUSIONS

According to the results of the experiments, it can be concluded that for a better result in the task of detecting infested Norway spruce trees (Picea abies (L.)) damaged by the bark beetle and counting the number of detected specimens in images obtained from UAV cameras, it is preferable to use the YOLOv4 architecture, trained on a dataset pre-processed by increasing the pixel contrast. It should be noted that the dataset composed of the pre-processed images showed a very good mAP metric for all trained models. The test results on the preprocessed images in comparison to the original images improved by 65% for YOLOv2, by 7.22% for YOLOv3, and by 3.19% for YOLOv4. Also, the results of image processing speed show that the architecture of the YOLOv4 version can be used as an expert agent for analyzing local areas using UAVs on medium resources (NVIDIA GeForce GTX 1060 GPU) in real-time.

We also note that the limitation of this study is the relatively small dataset. However, in this regard, different versions of YOLO architectures were used for the study. In addition, the considered models are intended to be used in real-time, which is our future research on the implementation of trained models at the UAV station.

ABBREVIATIONS

The following list of abbreviations was used in the manuscript:

| Abbreviation | Explanation of the abbreviation |
|--------------|----------------------------------|
| YOLO         | You Only Look Once               |
| RGB          | Red-Green-Blue                   |
| RS           | Remote Sensing                    |
| DL           | Deep Learning                     |
| CNN          | Convolutional Neural Network      |
| UAV          | Unmanned Aerial Vehicle           |
| DPM          | Deformable Part Models            |
| mAP          | Mean Average Precision            |
| IoU          | Intersection Over Union           |
| BCET         | Balance Contrast Enhancement Technique |

ACKNOWLEDGMENT

We are very thankful to Georgi Georgiev, Vladislav Trenkin, Margarita Georgieva, Stelian Dimitrov, and Martin Iliev from the Department of Forest Entomology and Phytopathology, Forest Research Institute – Bulgarian Academy of Sciences for providing the datasets and ground truth for the experiment.

REFERENCES

[1] H. Abdullah, R. Darvishzadeh, A. K. Skidmore, T. A. Groen, and M. Heurich, “European spruce bark beetle (Ips typographus, L.) green attack affects foliar reflectance and biochemical properties,”

[2] B. Wermelinger, “Ecology and management of the spruce bark beetle Ips typographus—a review of recent research,” Forest Ecology and Management, vol. 202, no. 1, pp. 67–82, Dec. 2004, doi: 10.1016/j.foreco.2004.07.018.

[3] T. Hlášny et al., “Bark Beetle Outbreaks in Europe: State of Knowledge and Ways Forward for Management,” Curr Forestry Rep, vol. 7, no. 3, pp. 138–165, Sep. 2021, doi: 10.1007/s40725-021-00142-x.

[4] E. Christiansen, R. H. Waring, and A. A. Berryman, “Resistance of conifers to bark beetle attack: Searching for general relationships,” Forest Ecology and Management, vol. 22, no. 1, pp. 89–106, Dec. 1987, doi: 10.1016/0378-1127(87)90098-3.

[5] G. Lobinger, “Die Lufttemperatur als limitierender Faktor für die Schwärmaktivität zweier rindenbrütender Fichtenkorkenfächerarten, Ips typographus L. und Pityogenes chlorographus L. (Col., Scolytidae),” Anz. Schadlingskd., Pflanzenschutz, Umweltschutz, vol. 67, no. 1, pp. 14–17, Jan. 1994, doi: 10.1007/BF01906563.

[6] C. Senf et al., “Canopy mortality has doubled in Europe’s temperate forests over the last three decades,” Nat Commun, vol. 9, no. 1, p. 4978, Nov. 2018, doi: 10.1038/s41467-018-07539-6.

[7] L. Marini et al., “Climate drivers of bark beetle outbreak dynamics in Norway spruce forests,” Ecography, vol. 40, no. 12, pp. 1426–1435, 2017, doi: 10.1111/ecog.02769.

[8] M. Onishi and T. Ise, “Explainable identification and mapping of trees using UAV RGB image and deep learning,” Scientific Reports, vol. 11, no. 1, Art. no. 1, Jan. 2021, doi: 10.1038/s41598-020-79653-9.

[9] A. Safonova, E. Guirado, Y. Maglinets, D. Alcaraz-Segura, and S. Tabik, “Olive Tree Biovolume from UAV Multi-Resolution Image Segmentation with Mask R-CNN,” Sensors, vol. 21, no. 5, Art. no. 5, Jan. 2021, doi: 10.3390/s21051617.

[10] A. Safonova, S. Tabik, D. Alcaraz-Segura, A. Rubtsov, Y. Maglinets, and F. Herrera, “Detection of Fir Trees (Abies sibirica) Damaged by the Bark Beetle in Unmanned Aerial Vehicle Images with Deep Learning,” Remote Sensing, vol. 11, no. 6, p. 643, Jan. 2019, doi: 10.3390/rs11060643.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” arXiv:1506.02640 [cs], vol. 56, May 2016, Accessed: Oct. 02, 2020. [Online]. Available: http://arxiv.org/abs/1506.02640

[12] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” arXiv:1612.08242 [cs], Dec. 2016, Accessed: Mar. 29, 2021. [Online]. Available: http://arxiv.org/abs/1612.08242
[13] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” arXiv:1804.02767 [cs], Apr. 2018. Accessed: Mar. 31, 2020. [Online]. Available: http://arxiv.org/abs/1804.02767

[14] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” arXiv:2004.10934 [cs, eess], Apr. 2020. Accessed: Mar. 29, 2021. [Online]. Available: http://arxiv.org/abs/2004.10934

[15] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “Scaled-YOLOv4: Scaling Cross Stage Partial Network,” arXiv:2011.08036 [cs], Feb. 2021. Accessed: Apr. 20, 2021. [Online]. Available: http://arxiv.org/abs/2011.08036

[16] A. Safonova, Plant Species Detection in Aerial and Satellite Images using Deep Learning. Universidad de Granada, 2021. Accessed: Dec. 03, 2021. [Online]. Available: https://digibug.ugr.es/handle/10481/69661

[17] X. X. Zhu et al., “Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources,” IEEE Geoscience and Remote Sensing Magazine, vol. 5, no. 4, pp. 8–36, Dec. 2017, doi: 10.1109/MGRS.2017.2762307.

[18] L. P. Osco et al., “A Review on Deep Learning in UAV Remote Sensing,” arXiv:2101.10861 [cs], Jan. 2021. Accessed: Mar. 30, 2021. [Online]. Available: http://arxiv.org/abs/2101.10861

[19] Y. Xia, P. d’Angelo, J. Tian, and P. Reinartz, “Dense matching comparison between classical and deep learning based algorithms for remote sensing data,” in The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Aug. 2020, vol. XLIII-B2-2020, pp. 521–525. doi: 10.5194/isprs-archives-XLIII-B2-2020-521-2020.

[20] R. Qiao, A. Ghodsi, H. Wu, Y. Chang, and C. Wang, “Simple weakly supervised deep learning pipeline for detecting individual red-attacked trees in VHR remote sensing images,” Remote Sensing Letters, vol. 11, no. 7, pp. 650–658, Jul. 2020. doi: 10.1080/2150704X.2020.1752410.

[21] H. Tao et al., “Deep learning-based dead pine tree detection from unmanned aerial vehicle images,” International Journal of Remote Sensing, vol. 41, no. 21, pp. 8238–8255, Nov. 2020. doi: 10.1080/01431161.2020.1766145.

[22] S. A. Krivets, E. M. Bisirova, I. A. Kerchev, E. N. Pats, and N. A. Chernova, “Transformation of taiga ecosystems in the Western Siberian invasion focus of four-eyed fir bark beetle Polygraphus proximus Blandford (Coleoptera: Curculionidae, Scolytinae),” Russ J Biol Invasions, vol. 6, no. 2, pp. 94–108, Apr. 2015. doi: 10.1134/S2075111715020058.

[23] A. Safonova et al., “Individual Tree Crown Delineation for the Species Classification and Assessment of Vital Status of Forest Stands from UAV Images,” Drones, vol. 5, no. 3, Art. no. 3, Sep. 2021, doi: 10.3390/drones5030077.

[24] N. H. Trang, Y. Diez, and L. Lopez, “Insect Damaged Tree Detection with Drone Data and Deep Learning Technique, Case Study: Abies Mariesii Forest, Zao Mountain, Japan,” vol. 22, p. 17917, May 2020.

[25] R. W. Reid, “Moisture changes in lodgepole pine before and after attack by the mountain pine beetle,” The Forestry Chronicle, vol. 37, no. 4, pp. 368–375, Dec. 1961, doi: 10.5558/tfc37368-4.

[26] Y. Yamaoka, R. H. Swanson, and Y. Hiratsuka, “Inoculation of lodgepole pine with four blue-stain fungi associated with mountain pine beetle, monitored by a heat pulse velocity (HPV) instrument,” Can. J. For. Res., vol. 20, no. 1, pp. 31–36, Jan. 1990, doi: 10.1139/x90-005.

[27] G. Amman, “The Mountain Pine Beetle-Identification, Biology, Causes of Outbreaks, and Entomological Research Needs,” Proc. of the Joint Canada / USA Workshop on Mountain Pine Beetle Related Problems in Western North America, pp. 7–12, Jan. 1982.

[28] J. Eichhorn et al., Part IV Visual assessment of crown condition and damaging agents. Thünen Institute of Forest Ecosystems, 2016. Accessed: Dec. 13, 2020. [Online]. Available: https://jukuri.luke.fi/handle/10024/537840

[29] E. Christiansen and A. Bakke, “The Spruce Bark Beetle of Eurasia,” in Dynamics of Forest Insect Populations: Patterns, Causes, Implications, A. A. Berryman, Ed. Boston, MA: Springer US, 1988, pp. 479–503. doi: 10.1007/978-1-4899-0789-9_23.

[30] L. J. GUO, “Balance contrast enhancement technique and its application in image colour composition,” International Journal of Remote Sensing, vol. 12, no. 10, pp. 2133–2151, Oct. 1991, doi: 10.1080/01431169108955241.

[31] S. Samanta et al., “Log Transform Based Optimal Image Enhancement Using Firefly Algorithm for Autonomous Mini Unmanned Aerial Vehicle: An Application of Aerial Photography,” Int. J. Image Graph., vol. 18, no. 04, p. 1850019, Oct. 2018, doi: 10.1142/S02194678185000195.