Available at www.sciencedirect.com

INFORMATION PROCESSING IN AGRICULTURE 9 (2022) 276–287

journal homepage: www.elsevier.com/locate/inpa

Automatic detection of *Acacia longifolia* invasive species based on UAV-acquired aerial imagery

Carolina Gonçalves [a,*], *Pedro Santana* [a,c], *Tomás Brandão* [a,b], *Magno Guedes* [d]

[a] *ISCTE - Instituto Universitário de Lisboa, Portugal*
[b] *Instituto de Telecomunicações, Lisboa, Portugal*
[c] *Information Sciences and Technologies and Architecture Research Center (ISTAR-IUL), Lisboa, Portugal*
[d] *IntRoSys SA, Portugal*

ARTICLE INFO

Article history:
Received 2 September 2020
Received in revised form
23 April 2021
Accepted 26 April 2021
Available online 30 April 2021

Keywords:
Pattern recognition
Convolutional neural networks
Invasive plants
Acacia longifolia

ABSTRACT

The *Acacia longifolia* species is known for its rapid growth and dissemination, causing loss of biodiversity in the affected areas. In order to avoid the uncontrolled spread of this species, it is important to effectively monitor its distribution on the agroforestry regions. For this purpose, this paper proposes the use of Convolutional Neural Networks (CNN) for the detection of *Acacia longifolia*, from images acquired by an unmanned aerial vehicle. Two models based on the same CNN architecture were elaborated. One classifies image patches into one of nine possible classes, which are later converted into a binary model; this model presented an accuracy of 98.6% and 98.5% in the validation and training sets, respectively. The second model was trained directly for binary classification and showed an accuracy of 98.8% and 98.7% for the validation and test sets, respectively. The results show that the use of multiple classes, useful to provide the aerial vehicle with richer semantic information regarding the environment, does not hamper the accuracy of *Acacia longifolia* detection in the classifier's primary task. The presented system also includes a method for increasing classification's accuracy by consulting an expert to review the model's predictions on an automatically selected sub-set of the samples.

1. Introduction

Invasive flora species are exotic species introduced into a non-native environment, being known for their rapid growth and proliferation, changing and reducing the biodiversity of the affected area. The invasive species *Acacia longifolia*, depicted in Fig. 1, is a small tree from southwest Australia, that exhibits characteristic yellow spike flowers. This species was introduced in Portugal for controlling dune erosion. However, due to its proliferation from excessive seed production (roughly 12,000 of seeds per m^2, per year [1]), it is currently considered an invasive species in Portugal and other countries. The overgrowth of this species poses tremendous pressure over resources, creating difficulties for native species to thrive.

Acacia longifolia invades forests and cultivation areas, altering the natural habitat composition of native species, with negative ecological and economic impacts. The mitigation of these negative impacts requires the application of early

* Corresponding author.
E-mail addresses: cdclg@iscte-iul.com (C. Gonçalves), pedro.santana@iscte-iul.pt (P. Santana), tomas.brandao@iscte-iul.pt (T. Brandão).
https://doi.org/10.1016/j.inpa.2021.04.007
2214-3173

(a) (b)

Fig. 1 – Dense presence of invasive species *Acacia longifolia* (a), with its well-known yellow spike flowers (b). Images from [1].

detection mechanisms for controlling the spread of invasive species. This can be done via physical mechanisms (*e.g.*, cutting) or via biological mechanisms (*e.g.*, biological agents that prevent seed growth). Due to these reasons, it is paramount to be able to detect the presence of this invasive species, allowing the early engagement of control and monitoring strategies.

A classical approach for vegetation mapping consists of using remote sensing imagery, *e.g.* hyperspectral satellite images, combined with traditional machine learning techniques, such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN), for the automatic classification of the various flora species [2]. However, traditional machine learning methods usually require context-specific feature extraction processes in order to provide input for the classifiers. Manually finding the most adequate set of features for a given image classification problem is usually a hard task. This can be avoided by using deep learning techniques for image classification, namely Convolutional Neural Networks (CNN). CNNs are able to simultaneously learn how to extract and how to use the most adequate image features for the learning problem at hand.

CNNs are becoming mainstream in several long-standing aerial image processing problems, such as segmentation and detection of vehicles [3–5] and object counting [6,7]. CNNs are also becoming widespread in the agriculture domain as preferred tool for detection and classification tasks based on UAV-acquired aerial imagery. Examples include coffee plant detection [8], classification of tree species [9], tobacco plant detection [10], classification of cultivation, grass, and other terrain categories [11–13], assessment of tree health stages [14], detection of individuals of the seaweed *Ulva prolifera* species [15], and land occupation classification [16–19]. In parallel to our work, recent articles related to the detection of invasive plants have been published [20,21]. Please refer to [22] for a survey on the application of deep learning techniques to the agriculture domain. However, to the best of our knowledge, the application of CNNs for detection and recognition of the *Acacia longifolia* invasive flora species remains unexplored.

This paper fills this gap by successfully showing that CNNs are a valuable tool for detecting the presence of *Acacia longifolia* species in aerial images captured by an Unmanned Aerial Vehicle (UAV). Imagery acquisition with a UAV allows an easier production of up-to-date data sets for environmental monitoring tasks [23–25], when compared to satellite-based alternatives [26].

This paper is organised as follows. First, the data set used to train and validate the CNN, as well as its acquisition process, are described in Section 2.1. Then, a set of preliminary experiments over a set of meaningful CNN configurations are presented in Section 2.2. The selected CNN architecture, as well as its training setup, are detailed in Section 2.3. Afterwards, an expert-based accuracy improvement mechanism is described in Section 2.4. This mechanism trades-off the potential accuracy gain resulting from asking for expert feedback and its associated cost. Subsequently, Section 3 presents and discusses the experimental results. Finally, a set of conclusions are drawn and suggestions for future research are provided in Section 4.

2. Materials and methods

2.1. Data set

A data set was prepared using a DJI Phantom 3 Pro (see Fig. 2 (a)) flying in autonomous navigation mode. The flight plans consisted of zig-zag patterns covering a set of rectangular regions. They were prepared with the DroneDeploy software and were executed autonomously on-board the UAV. Flight height was set to 25 meters from the ground launch position, maintaining this height regardless of the terrain irregularity. The flights occurred in 2016 and took place at three different Portuguese locations: Costa da Caparica, Palmela, and Sintra. They were usually performed between 10 AM and 3 PM, acquiring images with distinct illumination settings. Image acquisition was performed using the camera auto-focus while keeping a low exposure time, with the ISO setting at the minimum (100) for noise reduction. Overall, the UAV traversed 12 km, covering an area of 4 hectares. During the flights, 4000×3000 images were acquired with an on-board 2.7 k camera, mounted on a gimbal to ensure that it was always pointing downwards (see Fig. 2(b)). Table 1 summarises the specifications of the visual sensor equipped on the UAV.

The data set consists of 31 454 samples, which are 200×200 image patches extracted from a randomly selected sub-set of the images acquired by the UAV. The samples were hand-labelled by an engineering team and validated by a biology team. Each sample was tagged into one out of nine

(a) (b)

Fig. 2 – UAV used in the context of this work (a) and example of an image captured by the UAV during the data set acquisition phase (b).

Table 1 – Main parameters of the UAV's visual sensor.

Sensor	Sony EXMOR 1/2.3″
Effective pixels	12 M
Lens	FOV 94° 20 mm f/2.8, focus at 8
ISO Range	100~3200 (video) 100~1600 (photo)
Shutter Speed	8s −1/8000s
Image Max Size	4000 × 3000

possible classes: Acacia (7506 samples); non-Acacia yellow shrubs (1200 samples); cork oak (2912 samples); short herbs (3998 samples); wood (1200 samples); pine tree (2922 samples); other vegetation (6238 samples); dirt (2917 samples); and roadway (2561 samples). An *ad hoc* random sampling of the data set, for an additional human-based hand-labelling verification, showed an estimated hand-labelling error of 1.6%. This estimate resulted from the inspection of 7500 images in which 120 were found to be incorrectly labelled. This error rate is not expected to significantly interfere with the performance of the trained CNN since its frequency in the *Acacia longifolia* class samples is very low. All samples were randomly allocated to one of three sub-sets: training set (60% of the samples), validation set (20% of the samples), and test set (20% of the samples).

2.2. *Preliminary experiments*

For the automatic recognition of the invasive species *Acacia longifolia*, two models based on a convolutional neural network were developed and trained from scratch. This methodology allowed the development of a simple CNN model, suitable for training without the need for a larger data set, without the need for up-sampling the input images and without the need for powerful computational resources, as it would be required by general CNN models (*e.g.*, VGG, Inception or Resnet families). By studying the simplest possible model for the problem at hand, rather than applying a standard oversized model, the computational effort is reduced and, as a consequence, future porting to the UAV's onboard computer is facilitated.

It is well-known that the performance of neural networks in classification problems is influenced by the network's architecture and training settings. For this reason, several experiments were performed in order to evaluate the results achieved by varying the following structural and training settings: number of convolutional layers (4 to 8 layers, organized in pairs, where each pair is followed by a max pooling layer); convolution filter sizes ($3 \times 3, 5 \times 5, 7 \times 7$ and 11×11, where the size of the filter applied in the first layer is equal or greater than the ones applied in the remaining layers); learning rate (10^{-5} and 10^{-3}); and the optimization algorithm used during training (RMSProp or ADAM). Different configurations were therefore compared in terms of computational effort (number of trainable parameters) and class prediction error (prediction accuracy and cross-entropy loss function).

Table 2 depicts four architecture and training configurations, selected from the full suite of performed experiments. The number of convolution filters on the first pair of convolution layers is 64 on all configurations, and this number is doubled for each convolutional layer pair as one moves deeper into the network. Each pair of convolutional layers is followed by a max pooling layer with a 2×2 filter size and stride of 2 pixels, which are typically used in CNNs [27]. All network configurations end up with a 512-unit fully connected layer followed by a 9-unit output layer with softmax activation. The training process was performed up to a maximum of 200 epochs. An early training stop was triggered when no systematic decrease in the loss function values or evidence of overfitting was observed.

Configuration 1 is a CNN architecture consisting of eight convolutional layers (four pairs), with 7×7 and 5×5 sized

Table 2 – Studied architecture configurations and respective training results.

Configuration	Number of convolutional layers	Filter size		Number of trainable parameters		Optimization algorithm
		First layer	Remaining layers	Convolutional layers	Fully connected layers	
1	8	7×7	5×5	3,256,864	529,417	RMSProp
2	6	3×3	3×3	1,106,688	13,112,329	Adam
3	6	11×11	5×5	3,198,528	10,621,961	Adam
4	6	7×7	5×5	3,184,704	10,621,961	Adam

Configuration	Training time per epoch	Accuracy		Loss	
		Training	Validation	Training	Validation
1	≈ 36s	82.3%	76.5%	0.49	0.69
2	≈ 36s	93.8%	87.2%	0.17	0.39
3	≈ 1m 12s	91.3%	89.7%	0.24	0.27
4	≈ 54s	91.3%	91.9%	0.24	0.22

filters in the first and remaining layers, respectively. The training of this network was performed using the RMSProp optimizer and a learning rate of 10^{-5}. This configuration resulted in a low accuracy rate (68.7%) and a high loss value (0.81). A significant difference between the training and validation performance metrics was also noticed, suggesting the presence of over-fitting. Consequently, the number of convolutional layers was decreased to six (configurations 2–4). This structural variation reduced the amount of operations associated to the convolutions at the cost of increasing the number of trainable parameters at the fully connected layers. Nevertheless, the six layers configurations leads to substantially higher accuracy scores (87.2 to 91.9%) when compared with configuration 1. From these, configurations 3 and 4, which use larger sized filters, were the ones resulting in a lower discrepancy between training and validation outcomes. The difference between these two configurations relies on the filter size used in the first convolutional layer. However, using 11×11 filters (configuration 3) did not result in better accuracy and loss scores than using 7×7 filters (configuration 4). Configuration 4 also leads to a decreased number of convolution operations, which benefits both training and classification times.

Experiments with other mentioned structural and training setup combinations (not depicted in the table for the sake of simplicity), confirmed that the use smaller filters would lead to poorer learning generalization. They also showed that increasing the learning rate to 10^{-3} during the training process would lead to worse results. It was also verified that architectures with 4 convolutional layers, despite leading to much less convolution operations, contained a larger number of trainable parameters on the fully connected layers and attained rather unsatisfactory accuracy scores. Finally, networks trained using the Adam optimizer generally exhibited higher accuracy scores and lower generalization error than those trained with RMSProp.

Based on these preliminary experiments, it can be observed that configuration 4 was the one leading to the best results, showing higher accuracy and lower loss in the validation set. Furthermore, the results for training and validation are very close to each other, which casts away the possibility of over-fitting issues. Therefore, this configuration was selected as a baseline for implementation and further experimenting, both detailed in the following section.

2.3. *Model implementation and training*

Based on the preliminary results reported in the previous section, the implemented classification models were built upon configuration 4, from Table 2. For the sake of completeness, a full description of the devised CNN topology is provided in this section. The classification models are based on a CNN with six convolutional layers, with ReLu activation functions, where each two layers are interleaved with one max pooling layer. Two fully connected layers are appended at the end of the network to predict the sample's class. The first and second fully connected layers use ReLu and softmax activation functions, respectively. To reduce the chances of over-fitting to the training data, dropout was included after the max pooling layers as well as after the first fully connected layer, with rates of 0.2 and 0.5, respectively.

The input layer, *i.e.*, the first convolutional layer, receives 100×100 RGB colour images. To meet this input format, the 200×200 samples are first down-sampled with bilinear interpolation to 100×100 before being provided to the network. The number of filters for the first pair of convolutional layers is 64; a number that is doubled for each convolutional layer pair as one moves deeper into the network. The filter size for the first convolutional layer is 7×7, whereas the remaining convolutional layers are implemented using 5×5 filters. The layered structure of the network allows it to learn higher-level visual representations from the lower-level visual representations learned in the previous layers, in an end-to-end fashion. The pooling layers help bounding the number of network parameters to train and foster translation invariance.

Although the data set has been split into nine classes, the focus of this study is on the detection of the *Acacia longifolia* species, *i.e.*, to distinguish whether or not this species is present on a given image patch. To be able to produce a binary classification, the final softmax layer is composed of two output elements. This configuration is hereafter mentioned as CNNbin. Although the binary classification is the most relevant task for the purpose of this article, the use of multiple

classes may be useful for the UAV whenever it needs to obtain a rich semantic segmentation of the environment. For instance, a more detailed *in situ* analysis of the invasive species evolution (not covered in this article) would require the UAV to be able to select the best terrain patch to land on, for which the UAV would benefit from a detailed semantic segmentation of its surroundings. To assess whether training the model for classification of multiple (>2) classes hampers or not its performance in the primary task of *Acacia/non-Acacia* binary classification, a second network configuration was considered, hereafter mentioned as CNNmulti.

Additional implementation details can be found in Table 3, which summarises the structure of the network, and in Fig. 3, which provides a visual insight for the network topology.

The Adam optimizer [28] with a categorical cross-entropy loss function was used to train, from scratch, both network configurations during 200 epochs, with a batch size of 256 samples and a learning rate of 10^{-5}. Training was carried out using the Google's tool for deep learning experiments, Collaboratory, which allows running the algorithms using Tesla K80 GPUs. Both network configurations were implemented recurring to Keras and Tensorflow software packages. Training took roughly three hours. To train the binary classifier, CNNbin, the labels in the data set were changed to binary, that is, all samples from classes different from *Acacia* were altogether labelled as *non-Acacia*.

2.4. *Expert-based classification improvement*

Manual classification of aerial imagery is a time consuming task, and thus automating it as much as possible is a valuable endeavour. However, training an accurate machine learning-based classification system depends considerably on the amount of samples available on the training set, and its gathering is another time consuming task. In order to address these issues, the proposed system is endowed with a mechanism whose goal is to improve the post-training accuracy during run-time operation without the concern of having a more robust training set. This is attained by asking an expert to review low confidence predictions performed during run-time. By focusing expert invocations on likely relevant samples, the system trades-off the benefits of getting human help and its associated cost (*e.g.*, monetary, time).

The modelled CNNs output the probability of the input sample to belong to each of the possible classes. These probabilities can be interpreted as classification confidence levels, and thus they can be used by the system to determine which predictions are likely to be improved (*i.e.*, corrected) by a human expert. To accomplish this goal, the system applies a simple threshold-based decision making process: if a prediction is produced with a confidence level below a predefined threshold, it is submitted to the human expert for validation.

The higher the value for such confidence threshold, the higher will be the number of predictions submitted for revision by the human expert. The more relevant accuracy is for the task at hand, the higher this threshold should be set. On the other hand, the higher the cost for obtaining access to a human expert, the lower the threshold should be set. Hence, the optimal confidence threshold must be obtained under a multi-criteria optimisation framework, associating weights (relevance) to each of the criteria involved in the trade-off accuracy *vs.* expert invocation cost.

One can envision possible scenarios involving different trade-offs. Environmental intervention teams assembled to eliminate certain invasive plants in the field are not permitted to remove erroneous plants. Therefore, in this scenario, the confidence threshold should be pushed higher in order to ensure an higher overall accuracy, accepting the cost of also having an higher number of expert invocations for prediction verification. If these experts are part of the intervention team they should be available right way and thus they would be affordable. Conversely, let us consider a scenario in which a UAV is tasked to automatically scan a very wide area from a high-altitude and coarsely pinpoint potential presence of invasive plant spots, for subsequent low-altitude fine verification. In this case, the delay (cost) resulting from frequent consultation of a human-expert (over potentially multi-day missions) does not pay off the delay resulting from having the robot on hold before approaching the pinpointed spot. To avoid frequent human-expert invocations, the system's confidence threshold should be pushed low and, as a consequence, only infrequent low

Table 3 – CNN configurations. Convolutional layers: number of convolutional filters/ size of the convolutional filters/ activation function. Max pooling layers: pooling size/ dropout rate. Fully connected layers: number of neurons/ activation function/ dropout rate.

Layer	CNNbin	CNNmulti
Convolutional L1	64/7x7/ReLu	64/7x7/ReLu
Convolutional L2	64/5x5/ReLu	64/5x5/ReLu
Max Pooling L1	2x2/0.2	2x2/0.2
Convolutional L3	128/5x5/ReLu	128/5x5/ReLu
Convolutional L4	128/5x5/ReLu	128/5x5/ReLu
Max Pooling L2	2x2/0.2	2x2/0.2
Convolutional L5	256/5x5/ReLu	256/5x5/ReLu
Convolutional L6	256/5x5/ReLu	256/5x5/ReLu
Max Pooling L3	2x2/0.2	2x2/0.2
Fully Connected L1	512/ReLu/0.5	512/ReLu/0.5
Fully Connected L2	2/SoftMax/0	9/SoftMax/0

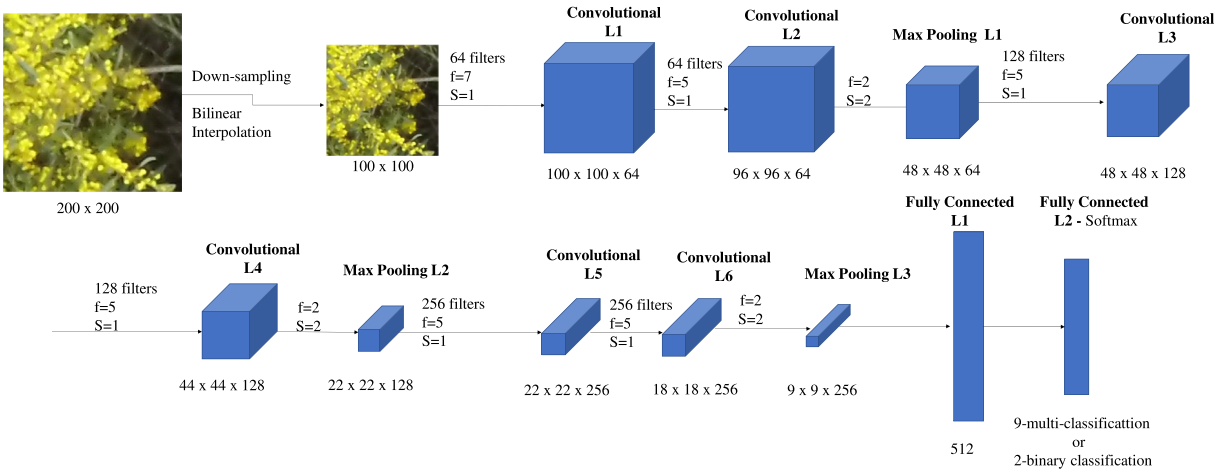

Fig. 3 – Architecture of the implemented CNN. The configuration of the output layer depends on the type of classification: nine units, for the multi-class classification case, or two units, for the binary case. 'f' represents the filter dimension (f × f) and 'S' is the stride.

confidence predictions would be verified by the human expert.

The aforementioned trade-off is herein handled by estimating the best confidence threshold according to a multi-criteria optimisation procedure. To formulate this procedure as a function minimisation process, a cost function needs to be defined. This cost function evaluates how poorly a given confidence threshold t allows the system to reach the desired trade-off. The two terms involved in the trade-off are weighted in the cost function according to an empirically defined scalar $\alpha \in [0, 1]$. The higher is α, the more relevant becomes the system's accuracy over the cost of invoking the expert for prediction correction. By tuning α, the system can be configured to perform differently, depending on the application scenario, as described in the previous paragraph.

Formally, for a given confidence threshold t, the cost function weights the cost associated to performing the expert calls required to evaluate all predictions below t, $c_c(t)$, and the (symmetric of) accuracy improvement obtained as a result of performing those expert calls, $c_a(t)$:

$$c(t, \alpha) = \alpha \cdot c_a(t) + (1 - \alpha) \cdot c_c(t). \quad (1)$$

The accuracy improvement term, $c_a(t)$, accounts for the difference between the classification accuracy obtained directly from the training data set, Φ, and the accuracy achieved once all predictions with confidence level below t are corrected by the human expert (assuming that the expert is flawless), $\phi(t)$:

$$c_a(t) = \Phi - \phi(t). \quad (2)$$

The term related to the cost of invoking the human expert, $c_c(t)$, is defined as the ratio between the number of predictions revised by the expert, $e(t)$, *i.e.*, those with a confidence level below t, and the total number of samples in the training set N_t:

$$c_c(t) = \frac{e(t)}{N_t}. \quad (3)$$

Finally, the confidence threshold that best handles the trade-off defined by a given α, $t_{min}(\alpha) \in [0 \ldots 100]$, is the one that minimizes the overall cost function:

$$t_{min}(\alpha) = \underset{t \in \{0, 1, 2, \ldots, 100\}}{\arg\min} \; c(t, \alpha). \quad (4)$$

As mentioned, this minimisation process is run offline on a training data set, given a user-defined trade-off α. The outcome is a confidence threshold, $t_{min}(\alpha)$, that can be used by the system during run-time, on the images that are to be acquired by a UAV in post-training flight missions. During run-time, all images that are classified by the CNN with a confidence level below the confidence threshold obtained during training, $t_{min}(\alpha)$, are submitted to the human expert validation and, potentially, corrected.

3. Results and discussion

This section presents the results achieved with the proposed system for automatic detection of the *Acacia longifolia* species from aerial images.

3.1. Automatic classification performance

Fig. 4 depicts the evolution of the loss function during training of the binary network configuration, CNNbin. The absence of

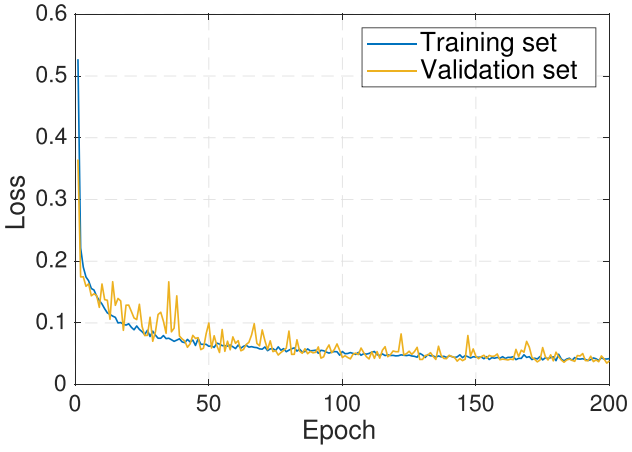

Fig. 4 – Loss evolution during the training of CNNbin.

a U-shaped curve for the loss function computed over the validation set shows that the network did not over fit the training set. The accuracy of the network at epoch 200 on the validation set is 98.8%, which can be confirmed by the high number of hits in the diagonal of the corresponding confusion matrix, depicted in Fig. 5.

Fig. 6 depicts the evolution of the loss during training of the multi-class network configuration, CNNmulti. Once again, the network did not over fit the training set, which can be visually confirmed by the evolution of the loss function in the training and validation sets. After 200 training epochs, the accuracy reached the value of 92.0%. The high number of hits in the main diagonal of the confusion matrix, depicted in Fig. 7, also shows that the network learned the ability to classify into nine classes. However, by observing the confusion matrix, it can be noticed that the *dirt* and *short herbs* classes present an higher classification error between themselves. These prediction errors may be due to the high content similarity between sample images belonging to those classes and/or due to a somewhat unbalanced data set.

To analyse how well the same network performs in the main classification task (*Acacia* vs. *non-Acacia*) without further training, the confusion matrix depicted in Fig. 7 was converted into a binary confusion matrix, depicted in Fig. 8. Based on this binary confusion matrix, the accuracy of the multi-class network on the binary classification problem is 98.6%.

Given the small decrease of 0.2 in accuracy, when compared with the binary network, CNNbin, it is possible to conclude that considering multiple classes does not hamper the network when performing its primary task: detecting the presence of the *Acacia longifolia* species.

Additionally, for the *Acacia* vs. *non-Acacia* classification task, it is also important to evaluate the precision, recall and F1-scores on both classification models. Table 4 presents such results. It also presents a summary of the results obtained with the test set, not used whatsoever during the training phase, showing that all performance measures are similar when the generalization requirements are pushed further. The recall value obtained using the CNNbin classification network is 95.2%, which means that only about 4.8% of the images belonging to the invasive species class were incorrectly classified; on the other hand, the precision achieved a high score of 99.1%, meaning that the occurrence of false positives is residual. The F1-score corresponding to these precision and recall values is 97.1%. The multi-classification network also presented satisfactory results with recall and precision scores of 96.0% and 97.6%, respectively (F1-score of 96.8%). These results show the advantage of using CNNs for the detection of *Acacia longifolia* from aerial images acquired by a UAV.

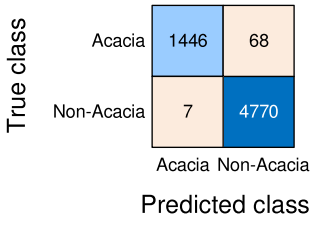

Fig. 5 – Binary confusion matrix obtained with CNNbin.

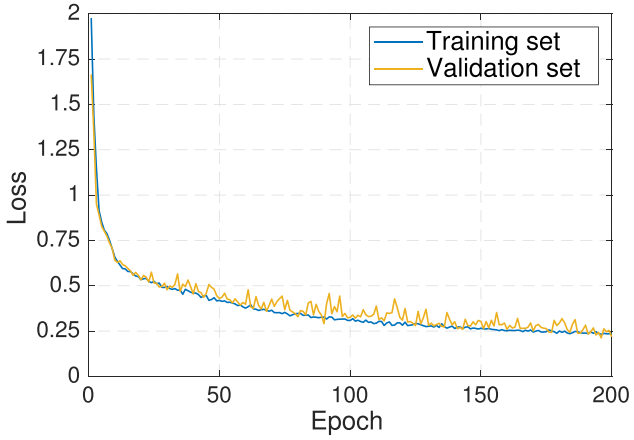

Fig. 6 – Loss evolution during the training of CNNmulti.

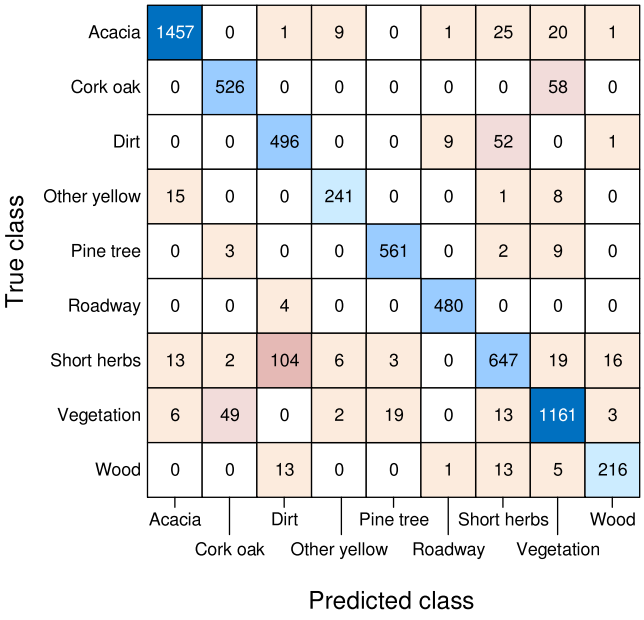

Fig. 7 – Confusion matrix obtained with CNNmulti.

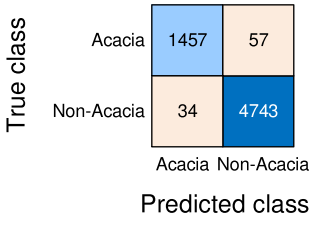

Fig. 8 – Binary confusion matrix obtained with CNNmulti.

For visual analysis of the prediction errors, Fig. 9 depicts a set of selected samples from the validation and test sets. The figure presents a few failure cases, which were often due to the presence of multiple classes in the same image patch. For instance, images in Figs. 9(c) and (g), hand-labelled as *Acacia*, were predicted as *non-Acacia* probably due to the fact that the plant only fills a small portion of the image. Images in Figs. 9 depict samples that were erroneously hand-labelled as *Acacia*, when they should have been labelled as *short herbs*.

Table 4 – Performance of *Acacia* vs. *non-Acacia* detection for both classification models.

	CNNMulti		CNNBin	
	Validation set	Test set	Validation set	Test set
Accuracy	98.6%	98.5%	95.5%	98.7%
Recall	96.2%	96.0%	96.9%	95.2%
Precision	97.7%	97.6%	99.5%	99.1%
F1-Score	96.9%	96.8%	98.1%	97.1%

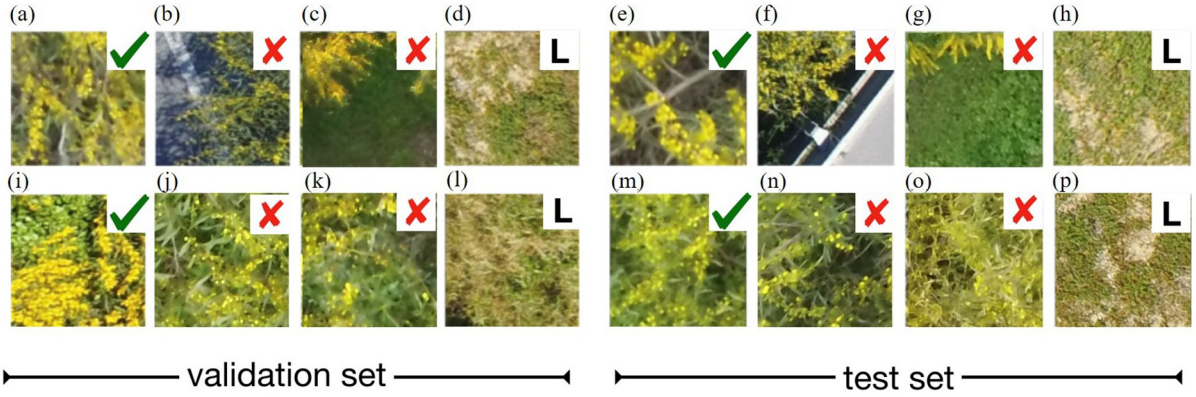

Fig. 9 – Selected samples from validation and test data sets that were correctly classified (green check), misclassified (red cross), and incorrectly hand-labelled though correctly classified by the network (letter L) by CNNbin (bottom row) and CNNmulti (top row) networks.

Nevertheless, the CNNmulti and CNNbin networks were able to correctly classify them as *short herbs* and *non-Acacia*, respectively. This shows that the CNNs can cope with the noise that often pollutes hand-labelled data sets.

Table 5 presents the class probabilities predicted by the networks for a sub-set of the samples depicted in Fig. 9. The table shows that there are misclassified samples whose probability is nevertheless close to the actual predominant class. For instance, the sample depicted in Fig. 9(k) exhibits probabilities that are strongly concentrated, yet equally distributed, on the *Acacia* and *non-Acacia* classes. These results suggest that the false negative rate could be eventually reduced by classifying a sample as *Acacia* if that class is the one predicted as the most likely or, if not, it is close to the most likely. Under this assumption, the samples depicted in Figs. 9(k) and (o) would have been correctly classified as *Acacia*. However, additional testing would be required in order to take the above assumption for granted.

Since the proposed system provides classifications for 200×200 image patches, an additional mechanism is required for segmenting an entire input image acquired by the UAV. Two approaches can be considered. The fastest approach is to sample the input image with a regular grid, extracting non-overlapping samples of 200×200 pixels which are individually submitted to the classification network. The output is a low resolution segmentation of the input image, which can be sufficient if the UAV only needs to obtain a coarse estimation of the *Acacia longifolia* presence. An example is depicted in Fig. 10.

A finer segmentation output can be produced by applying the classification network to a sliding window. In this case, the prediction produced by the network is used for classifying

Table 5 – Class probabilities predicted for a set of selected samples. Labels for CNNmulti network: *Acacia* (0); *Cork oak* (1); *Dirt* (2); *Other yellow* (3); *Pine tree* (4); *Roadway* (5); *Short herbs* (6); *Vegetation* (7); and *Wood* (8). Labels for CNNbin network: *Acacia* (0); and *Non-Acacia* (1).

Network	Sample	Class Probabilities
CNNmulti	Fig. 9(d)	[0: 0.027; 1 to 5: ≈ 0; **6: 0.969**; 7: 0.004; 8: ≈ 0]
	Fig. 9(h)	[0: 0.100; 1 to 5: ≈ 0; **6: 0.892**; 7: 0.008; 8: ≈ 0]
CNNbin	Fig. 9(k)	[**0: 0.486**; **1: 0.514**]
	Fig. 9(l)	[0: 0.047; **1: 0.953**]
	Fig. 9(o)	[**0: 0.461**; **1:0.539**]
	Fig. 9(p)	[0: 0.0983; **1:0.902**]

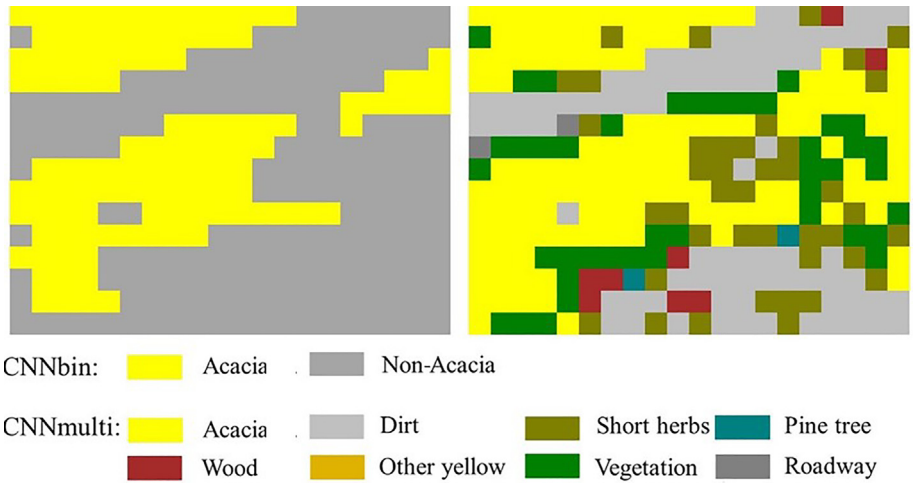

Fig. 10 – Full image segmentation obtained with the classifier applied on top of a regular grid. Left: CNNbin applied to the image depicted in Fig. 2b. Right: CNNmulti applied to the image depicted in Fig. 2b.

the central pixel of the processed patch. The result will be an image with smoother regions that correspond to the different classes. An example can be observed in Fig. 11. It is worth to mention that this approach produced segmented images that did not exhibit significant noise, which denotes that the system is not sensitive to small input variations.

3.2. *Expert-based accuracy improvement*

As described in Section 2.4, the proposed system includes a mechanism that computes an optimal prediction confidence threshold. This mechanism weights the cost of calling a human expert for classification revision and the benefit of an improved accuracy resulting from those revisions. The user is able to control the importance (weight) of each of these two conflicting criteria by tuning the scalar α in Eq. (1). When α is zero, the cost of calling the expert is given the maximum importance, whereas when α is one, the accuracy gain is given the maximum importance. All run-time predictions associated to a confidence level below the optimal confidence threshold are submitted for revision by a human expert.

Fig. 12 plots the optimal confidence thresholds, $t_{min}(\alpha)$, computed for each possible value of α in the range $[0 \ldots 1]$ with increments of 0.01, according to Eq. (4). It can be observed that, as α increases, the confidence threshold that minimizes the cost function, $t_{min}(\alpha)$, also increases. With higher α, the cost of improving the accuracy is emphasized and, consequently, the confidence threshold increases; this ensures a gain in terms of accuracy that results from correcting the samples that have been classified with a confidence level below $t_{min}(\alpha)$. On the other hand, with lower values of α, emphasizing the cost of performing expert calls, the optimal confidence threshold decreases, resulting in a smaller amount of sample classifications to be revised by the expert.

Fig. 13 depicts the evolution of the accuracy gain over the percentage of samples reviewed by the expert, using the training set on both the CNNbin and CNNmulti classification

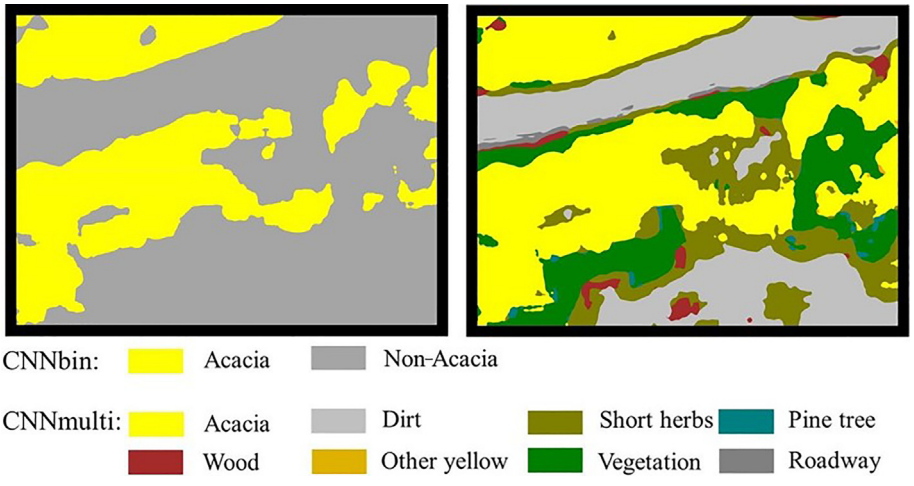

Fig. 11 – Full image segmentation obtained with the classifier applied with a sliding window. Left: CNNbin applied to the image depicted in Fig. 2b. Right: CNNmulti applied to the image depicted in Fig. 2b.

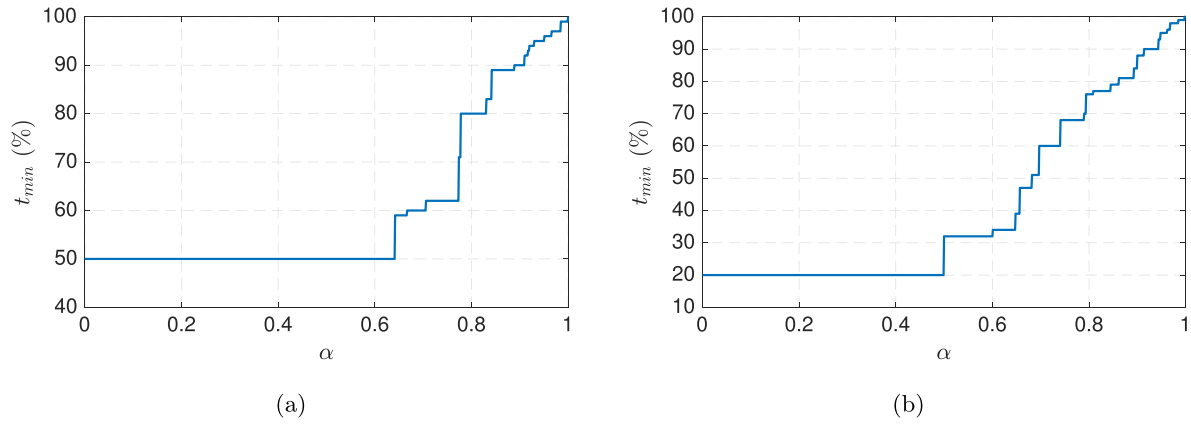

Fig. 12 – Confidence threshold, t_{min}, that minimizes the trade-off cost as a function of α, for CNNbin (a) and CNNmulti (b).

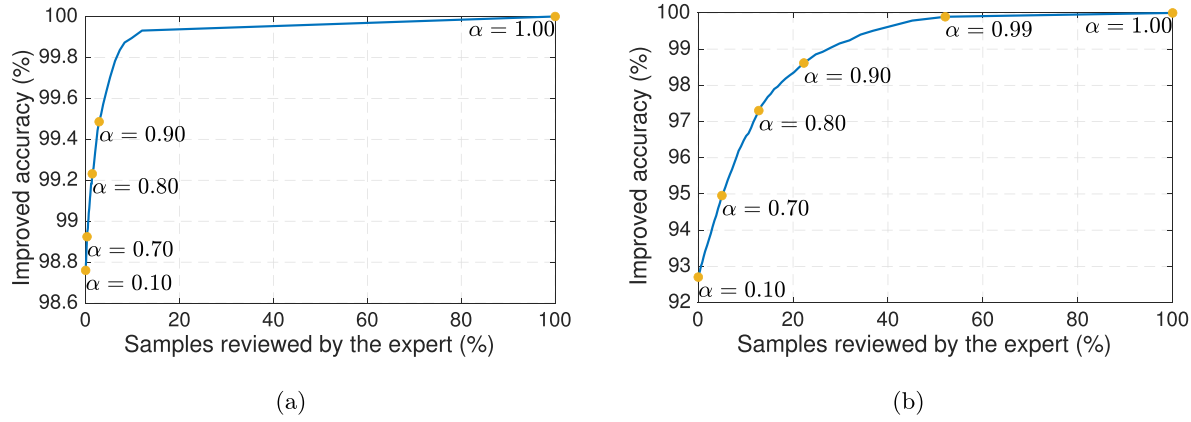

Fig. 13 – Accuracy improvement as a function of the number of expert calls using CNNBin (a) and CNNMulti (b).

models. As expected, the accuracy increases with an increasing number of reviewed samples. Since the original system's accuracy for the multi-class classification was lower than the accuracy for the binary classification, its curve exhibits a smoother increase as a higher percentage of samples need to be reviewed to achieve a given accuracy target. The figure also shows that the proposed expert-based method for post-training accuracy improvement displays a predictable behaviour over the range of α values. Concretely, the expected accuracy gain increases as α increases, meaning that the a system's user is able to predict the outcome of a given trade off between accuracy gain and expert calls cost. The user is thus able to fine tune the system as a function of the task at hand and the cost of using the specialist. If the task is related to the physical control of the invasive species, namely grubbing or cutting, it is important to ensure that the flora subject to the control procedures belongs to the invasive species. Therefore, the confidence level of the prediction should be close to 100%. Since sample predictions of the invasive species may present lower confidence values, it may be required to invoke the specialist, accepting the cost of it, in order to ensure that no costly labour resources will be spent for unnecessary physical control.

The analysis presented in the previous paragraph is based on the results obtained with the training set. It is also necessary to verify whether these results are consistent with the ones obtained with the validation data set, that is, if they generalise to data that was not observed during training. Hence, the validation data set is herein used as a surrogate of the data sets that are to be acquired and processed in post-training flight missions.

To assess the generalisation capabilities of the method proposed to compute a confidence threshold, Eq. (4) is applied, for every possible α, to both training and validation data sets, separately. As a result, for every possible α, two confidence thresholds are produced, one for the training data set, $t_{min}(\alpha)$, and another for the validation data set, $t^*_{min}(\alpha)$. The confidence values obtained for the two data sets can be compared using the RMSE error metric:

$$RMSE = \sqrt{\frac{\sum (t_{min}(\alpha) - t^*_{min}(\alpha))^2}{N_\alpha}}, \quad (5)$$

where N_α represents the total number of possible values for α. The application of the RMSE error metric aims at testing the similarity between the obtained confidence thresholds for both training and validation data sets. The resulting mean squared error (dissimilarity) was very small, corresponding to 0.105 and 0.249, for the binary and multi-class classification models, respectively. These values suggest that the calculated confidence threshold using the training set may be

generalized to the validation set and, possibly, to data that was not observed during the training phase. As expected, the error is higher for the CNNmulti network, which displays higher difficulty in determining a properly generalisable threshold, when compared with the binary classification case.

3.3. Discussion

The results depicted along this section show that the developed CNN architecture accurately predicts the presence of *Acacia longifolia* species. Satisfactory results were also achieved for the prediction of other terrain classes, which can be useful for providing the UAV with semantic information about the environment. The proposed end-to-end solution does not depend on a manual feature extraction process, as required by traditional classification methods, such as shallow artificial neural networks.

Previous work in detection and classification of flora from images in the visible spectrum does not cover the specific case of the *Acacia longifolia* species, hampering a direct comparison with the present work. Nevertheless, the following discusses the main differences between the present and previous work. Transfer learning was often employed to boost flora detection from images using general purpose classification and semantic segmentation CNNs: a GoogLeNet-based solution achieved an overall accuracy of 89.0% for the classification of seven distinct tree species [9]; a fully convolutional network-based approach achieved an accuracy of 88.3% for weed detection in rice fields [11]; and a Segnet encoder-decoder network achieved a F1-score of about 84.9% for weed detection in sugar beet crop fields [12]. More recently, Qian et al. [20] proposed a CNN architecture inspired on concepts taken from the AlexNet, GoogLeNet and VGG networks, achieving a global accuracy of 93.4% for the classification of seven different tree-like invasive species. Despite presenting good results, the complexity associated to the solutions based on pre-existing CNN models poses challenges when deploying in small UAVs with limited energy, memory, and computational resources. Conversely, the present work proposes a simpler CNN architecture, with fewer convolution layers, while achieving an overall accuracy of 98.7% and a F1-score of 97.1% for the detection of *Acacia longifolia*.

4. Conclusion

The application of convolutional neural networks for detecting the *Acacia longifolia* invasive species from aerial images acquired by unmanned aerial vehicles was studied. Two models based on the same CNN architecture were elaborated: one for the distinction of nine classes, and another focused on the Acacia/non-Acacia binary classification. The accuracy scores attained for the multi-class and binary-class models were of 98.5% and 98.7%, respectively. These results show the validity of the CNN-based approach and, consequently, the viability of using aerial vehicles for automated large-scale mapping of *Acacia longifolia* individuals. Moreover, it was also shown that the use of a multi-class classifier does not degrade the system's performance when applied to the primary binary classification task. Therefore, the aerial vehicle may exploit the multi-class classifier to obtain a richer semantic description of the environment without hampering its ability to accurately detect *Acacia longifolia* individuals.

The proposed system includes a mechanism to determine when to invoke an expert for revision and correction of low confidence predictions. These predictions are selected by trading off the benefit of improving the classification accuracy and the cost of invoking the expert. Given the final application requirements, the system user is allowed to manage this trade-off by tuning a single free parameter.

Future work involves allowing the aerial vehicle approaching detected individuals for closer inspection and recognition of other Acacia species. This detailed inspection would allow to distinguish the various species by using imagery of the leaf structure or other relevant plant characteristics.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The fieldwork carried for data set creation was partially funded by FCT and COMPETE/FEDER, through project INVADER-IV, grant PTDC/AAG-REC/4896/2014. The authors are grateful to Raquel Caldeira, E. Marchante, H. Mar- chante, J. Palhas, M. Dinis, N. César de Sá, R. Matos, B. Pato, and D. Deppen for their commitment to this project.

REFERENCES

[1] InvasorasPt. Acacia longifolia. link: https://www.invasoras.pt/sites/default/files/acacia_longifolia_torrinha.pdf;. 2015 (publishing time)/ 2021 (referencing time). (in Portuguese).
[2] Martins FD. Utilização de técnicas de deteção remota na identificação de Acacia sp. na Região Centro Sul de Portugal Continental [PhD thesis]. IPCB. ESA; 2012. (in Portuguese).
[3] Audebert N, Le Saux B, Lefèvre S. Segment-before-detect: Vehicle detection and classification through semantic segmentation of aerial images. Remote Sensing 2017;9(4):368.
[4] Tang T, Zhou S, Deng Z, Lei L, Zou H. Fast multidirectional vehicle detection on aerial images using region based convolutional neural networks. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE; 2017. p. 1844–47.
[5] Weizheng S, Zhang A, Zhang Y, Wei X, Sun J. Rumination recognition method of dairy cows based on the change of noseband pressure. Informat Process Agric 2020;7(4):479–90.
[6] Liu Y, Sun P, Highsmith MR, Wergeles NM, Sartwell J, Raedeke A, et al. Performance comparison of deep learning techniques for recognizing birds in aerial images. In: 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC). IEEE; 2018. p. 317–24.
[7] Gray PC, Fleishman AB, Klein DJ, McKown MW, Bezy VS, Lohmann KJ, et al. A convolutional neural network for detecting sea turtles in drone imagery. Methods Ecol Evol 2019;10(3):345–55.

[8] Castelluccio M, Poggi G, Sansone C, Verdoliva L. Land use classification in remote sensing images by convolutional neural networks. ArXiv preprint ArXiv:150800092. 2015.
[9] Onishi M, Ise T. Automatic classification of trees using a UAV onboard camera and deep learning. ArXiv preprint ArXiv:180410390. 2018.
[10] Fan Z, Lu J, Gong M, Xie H, Goodman ED. Automatic tobacco plant detection in UAV images via deep neural networks. IEEE J Sel Top Appl Earth Obser Remote Sens 2018;11 (3):876–87.
[11] Huang H, Deng J, Lan Y, Yang A, Deng X, Zhang L. A fully convolutional network for weed mapping of unmanned aerial vehicle (UAV) imagery. PloS One 2018;13(4):e0196302.
[12] Sa I, Chen Z, Popović M, Khanna R, Liebisch F, Nieto J, et al. weednet: Dense semantic weed classification using multispectral images and mav for smart farming. IEEE Robot Automat Lett 2017;3(1):588–95.
[13] Sa I, Popović M, Khanna R, Chen Z, Lottes P, Liebisch F, et al. WeedMap: A large-scale semantic weed mapping framework using aerial multispectral imaging and deep neural network for precision farming. Remote Sensing 2018;10(9):1423.
[14] Safonova A, Tabik S, Alcaraz-Segura D, Rubtsov A, Maglinets Y, Herrera F. Detection of fir trees (Abies sibirica) damaged by the bark beetle in unmanned aerial vehicle images with deep learning. Remote Sensing 2019;11(6):643.
[15] Wang S, Liu L, Qu L, Yu C, Sun Y, Gao F, et al. Accurate Ulva prolifera regions extraction of UAV images with superpixel and CNNs for ocean environment monitoring. Neurocomputing 2019;348:158–68.
[16] Ham S, Oh Y, Choi K, Lee I. Semantic segmentation and unregistered building detection from Uav images using a deconvolutional network. Int Arch Photogram, Remote Sensing Spatial Informat Sci 2018;42(2).
[17] Ševo I, Avramović A. Convolutional neural network based automatic object detection on aerial images. IEEE Geosci Remote Sensing Lett 2016;13(5):740–4.
[18] Deng Z, Sun H, Zhou S, Zhao J, Lei L, Zou H. Multi-scale object detection in remote sensing imagery with convolutional neural networks. ISPRS J Photogramm Remote Sens 2018;145:3–22.
[19] Sun Y, Zhang X, Xin Q, Huang J. Developing a multi-filter convolutional neural network for semantic segmentation using high-resolution aerial imagery and LiDAR data. ISPRS J Photogramm Remote Sens 2018;143:3–14.
[20] Qian W, Huang Y, Liu Q, Fan W, Sun Z, Dong H, et al. UAV and a deep convolutional neural network for monitoring invasive alien plants in the wild. Comput Electron Agric 2020;174:105519.
[21] Singh G, Reynolds C, Byrne M, Rosman B. A remote sensing method to monitor water, aquatic vegetation, and invasive water hyacinth at national extents. Remote Sensing 2020;12 (24):4021.
[22] Kamilaris A, Prenafeta-Boldú FX. Deep learning in agriculture: A survey. Comput Electron Agric 2018;147:70–90.
[23] Pinto E, Marques F, Mendonça R, Lourenço A, Santana P, Barata J. An autonomous surface-aerial marsupial robotic team for riverine environmental monitoring: Benefiting from coordinated aerial, underwater, and surface level perception. In: 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014). IEEE; 2014. p. 443–50.
[24] Deusdado P, Guedes M, Silva A, Marques F, Pinto E, Rodrigues P, et al. Sediment sampling in estuarine mudflats with an aerial-ground robotic team. Sensors 2016;16(9):1461.
[25] Silva A, Mendonça R, Santana P. Monocular trail detection and tracking aided by visual SLAM for small unmanned aerial vehicles. J Intell Robotic Syst 2020;97(3):531–51.
[26] Cruzan MB, Weinstein BG, Grasty MR, Kohrn BF, Hendrickson EC, Arredondo TM, et al. Small unmanned aerial vehicles (micro-UAVs, drones) in plant ecology. Appl Plant Sci 2016;4 (9):1600041.
[27] Stanford. Convolutional neural networks for visual recognition; 2015. [Online; accessed July 27, 2019]. Available from: https://cs231n.github.io/convolutional-networks/.
[28] Kingma D, Ba J. Adam: A method for stochastic optimization. In: International Conference on Learning Representations. 2014.