2022 IEEE 22nd International Symposium on Computational Intelligence and Informatics and 8th IEEE International Conference on Recent Achievements in Mechatronics, Automation, Computer Science and Robotics (CINTI-MACRo) | 979-8-3503-9882-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CINTI-MACRo57952.2022.10029588

Profiling the Arion rufus snails with computer vision

Grega Vrbančič*, Rok Kukovec*, Iztok Fister* and Vili Podgorelec*, Sancho Salcedo-Sanz†, Iztok Fister Jr.*†
*, Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia
Email: *grega.vrbancic@um.si
†, Dept. of Signal Processing and Commun., Universidad de Alcalá
Alcalá de Henares, Spain

***Abstract*—Smart agriculture takes advantage of modern computational approaches that vary from IoT, cloud computing, and artificial intelligence. The primary aim is to assist the farming process. Pest detection is one of the objectives within the area of smart agriculture. It is mainly solved by computer vision approaches, usually combined with machine learning (ML) algorithms. In this paper, we propose a solution for detecting Arion rufus snails that have emerged in Central Europe and are one of the most prolific threats to agriculture in that place. Practical experiments reveal that our method is helpful in this real-world application and opens several future challenges and lines of research.**

***Index Terms*—computer vision, machine learning, convolutional neural networks, pest detection**

I. Introduction

Agriculture has become one of the essential domains for dealing with the challenge of feeding ten billion expected people on Earth in the future. In line with this, the new discipline of smart farming has emerged. The challenge of providing such a large amount of food encourages higher production of food through findings of the latest computer technology (e.g., internet of things (IoT), big data, artificial intelligence (AI), machine learning (ML)), and full automation of food production [1]. One of the critical fields in smart farming is searching for natural solutions against different pests that prevent the expected growth of crops.

Slugs (Arion vulgaris, Arion rufus, Arion ater) are one of the most destructive pests in Europe [2]. Although some slugs are cannibals, eating the other slugs and snails, the most harmful are those that feed on living plants. Slugs are invasive species with practically no natural enemies in our part of the world. Slugs and snails can also host lungworms. Accidentally eating raw slugs, i.e., the lettuce picked from the garden is not washed thoroughly, brings significant disease risk. These slugs are the most common cause of human eosinophilic meningoencephalitis [3].

There are many different ways to get rid of them. Farmers typically use two different baits in the battle against these pests, i.e., iron phosphate baits and more toxic metaldehyde. However, both means leave harmful effects on fertile soil and are not following ecological food production. A more environmental and, at the same time, cheaper way of getting rid of those pests involves human intervention through hand-picking. Nevertheless, this approach is impractical due to the following facts: Slug's trail of movement may be slightly visible due to the slime left behind, but that is not nearly enough to locate them. Their color can be very similar to their surroundings, and their size makes finding them even more difficult for a human.

Today, AI is used at every step of our lives. Therefore, an approach to detect slugs in a garden has been tested. A Convolutional Neural Network (CNN) is proposed for the classification of the pest. More precisely, a custom object-detection computer vision algorithm is proposed based on the framework You-Only-Look-Once, version 4 (YOLOv4) [4]. The results of the experiment are based on digital images of the garden, taken with our cameras, and detecting the Arion Rufus snail on them using the computer vision method. The objective is to show that slugs can be discovered automatically. Thus, the final aim of this research is to develop a robust Arion rufus snail detection method and show that the proposed method works well even though this kind of snail is hard to detect in its natural environment. In line with this, the snail image test set is prepared to make it publicly available on the Internet. The preliminary results of the utilized approach show that it has massive potential in the battle against these pests.

The main contributions of the conducted research can be summarized as follows:

- collected and labeled Arion rufus dataset,
- trained YOLOv4-tiny CNN against the collected dataset, and
- quantitative and qualitative evaluation of the trained model.

The structure of the remainder of the paper is as follows: Next section discusses and compares some of the related work that uses computer vision for animal recognition. In Section III, we present the obtained dataset and used method. Section IV explains the experimental setup and training process, together with an evaluation of the obtained results. The last section concludes the article with a summary of our findings and outlines the possibilities for future work.

II. Computer vision for animal recognition

Formally, computer vision is a sub-field of artificial intelligence that enables computer systems to extract and derive meaningful information from different inputs, such as digital images or videos, and take actions or make recommendations

based on that information [5]. In recent years, a major increase in the field of computer vision can be observed in various fields of study such as medicine [6], biology [7], sports [8], astronomy [9]. as well as smart farming [10] and agriculture [11].

The utilization of computer vision systems for identifying animals dates back to the early 1990s [12], [13]. The field has developed quickly since then, primarily due to the significant increase in computing power and the development of various advanced methods, techniques, tools, and libraries, which made application to the specific domain problem easier. The advancement of new state-of-the-art ML approaches and techniques, especially deep learning for computer vision [14]–[16], offers powerful methods for improving the accuracy of image-based identification analyses. Such approaches for the animal detection tasks are essential, adding information such as wildlife accidents to other animal recognition and classification approaches [17]–[19]. Moreover, automatic animal identification and counting could improve all biological missions that require identifying species and counting individuals, including animal monitoring and management, examining biodiversity, and estimating population [20].

Many attempts to automatically or semi-automatically identify animals from various video sources have been reported; however, many of them relied on hand-designed features [21], [22] to detect animals and/or count them. The majority of reported modern approaches exploit the capabilities of CNNs [19], [23], [24] to address the problem of animal identification. Based on the authors' knowledge and papers studied, only a few research papers address the problem of detecting Arion rufus snails in existence [25].

III. Materials and methods

The main characteristic of the Arion rufus snails is that they are inactive during the day. They start appearing in the late afternoon and stay active during the night. The main problem that arises in observing the snails in the dark is that they are difficult to identify due to their insensitivity to infrared radiation cameras. Therefore, each step in helping eradicate these pests is appreciated. For the task of identifying Arion rufus snails, we first obtained and prepared an image dataset, which was then used to train and evaluate against using the selected CNN architecture. In this section, the process of obtaining and preparing the dataset and selected method are described in-depth.

A. Snails dataset

In order to train the YOLOv4-tiny CNN model for the task of identification of Arion rufus snails, it is required to have a big enough dataset of the images capturing the Arion rufus snails. The captured images must also be labeled (marked) with bounding boxes so that the model can learn how to detect them. Therefore, we collected and prepared our own dataset of Arion rufus images[1]. The process of obtaining the images in a real environment, image pre-processing, and the process of labelling are described briefly in the following subsections.

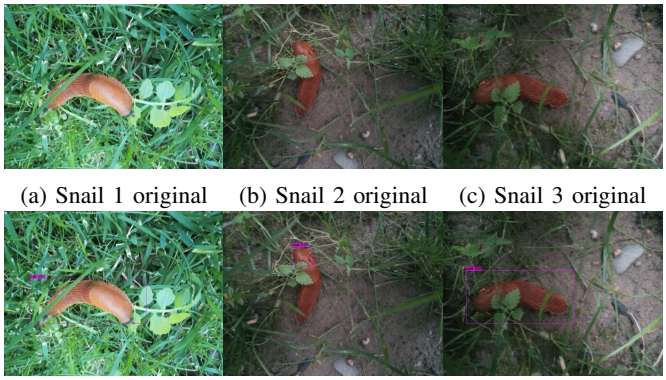
1) Obtaining images of snails in a real environment: The training of the predictive CNN model requires a sufficient amount of images in the process of training in order to be able to successfully learn to detect Arion rufus snails. The amount and also the quality of images are crucial in the process of training CNN since it has a direct impact on the predictive performance of the trained model. Since not many datasets are devoted to detecting of Arion rufus snails, we decided to obtain and process images on our own. The images were captured in the northeastern region of Slovenia between June and August of 2022. The micro-locations of places where images were captured vary from ordinary house backyards to the meadow next to the forest and forest footpaths. Different micro-locations of captured images are essential due to diverse image surroundings. This helps to train the prediction model for identifying snails when the image background around the snail is changing. The images were captured using two different cameras. The images were captured from around 10cm - 50cm above the ground at different angles. While two different cameras were used to capture the Arion rufus snails images in the real environment, the collected images are in different sizes: 2310×3072 pixels and 1734×2306 pixels. In total, more than 500 images of Arion rufus snails were obtained, with a different number of snails present in each image.

2) Image pre-processing: Photographs are transferred from our camera to the computer in the image pre-processing step. The images were firstly manually inspected and evaluated. The primary goal of the manual process was to eliminate potential duplicates, remove blurry images and remove potentially captured images without the Arion rufus snail present. This process was conducted independently by two researchers and in such a way that the first researcher went through all the captured images and removed the unsuitable ones. After that, the second researcher reviewed the remaining images, which the first researcher marked as suitable. This way, we tried to reduce potential human error and increase the overall quality of the prepared snails dataset. After the inspection process was completed, a total of 396 images were selected as suitable for the dataset.

3) Labelling snails using CVAT: After the images were collected, and manually inspected by two researchers, the process of labeling the Arion rufus snails was conducted. From the object detection standpoint, labeling refers to annotating the area or multiple areas on each image where the object targeted for detection is present. Commonly, those areas are in the form of rectangles and are stored in a text file for each corresponding image. Since this is probably the most burdensome and time-consuming task, the software named CVAT [26] was used to make labeling the images easier. CVAT is a graphical image annotation tool in the form of a web application that can be easily used to label the various objects in images and classify them into different classes. In our case, we labeled only one class of objects - Arion rufus snails. These

[1]The dataset is available at https://github.com/firefly-cpp/snail-dataset

steps were conducted in a way that one researcher labeled all 396 images. Afterward, the second researcher went through all the labels and inspected whether the position of the bounding boxes was correct. Images were also inspected for any snails on the image that had not been labeled. The final outcome of the labeling process can be observed in figure 1.



(a) Snail 1 original (b) Snail 2 original (c) Snail 3 original

(d) Snail 1 marked (e) Snail 2 marked (f) Snail 3 marked

Fig. 1: Sample images from a dataset, where the top row images are without labels and the bottom row shows the labeled images.

B. Object detection with YOLOv4

The task of object detection is not only to focus on classification but also to precisely estimate the concepts and location of objects contained in each image. Object detection is one of the fundamental computer vision problems since it can provide us with valuable information for a semantic understanding of images and videos. The pipeline of traditional object detection models can be, in general, divided into three phases: informative region selection, feature extraction, and classification. Through the years, many successful attempts to address the problem of object detection have been proposed. They can be mainly categorized into two groups. The first group follows the traditional object detection pipeline, producing proposed regions and then classifying each proposed region into different object categories. The second group is tackling object detection as a regression or classification problem, adopting a unified framework directly to achieve final results (categories and locations). The most typical representatives of the first group are the R-CNN [27] based methods, while the most known representatives of the second group are AttentionNet [28], YOLO [29], and its variations and SSD [30] and its derivatives [31].

In our case, we have chosen a YOLOv4 based variation of the object detection model called YOLOv4-tiny. YOLOv4 became quite a widely used algorithm for object detection tasks. Since training such a model is quite time-consuming, we decided to utilize the previously mentioned YOLOv4-tiny, which provides us with an excellent trade-off between training time and accuracy.

All of the YOLOv4 variations and derivatives exploit CNN capabilities to detect objects in real-time with only one forward propagation through the neural network, as the name of the methods suggests. In the beginning, the image is passed in on input, which is then divided into $S \times S$ uniform grid of non-overlapping cell, where for each cell boundary box is predicted. Boundary box is composed of x, y, w, h values and confidence $C(Object)$. The values x and y represent the position coordinates of the detection boundary box relative to the grid, while the values (w, h) denote the width and height of the detection boundary box. Denoted with $C(Object)$ is a prediction of C categories, where the confidence score reflects the probability of the model to include the target object and the accuracy of the prediction detection box. Formally, the $C(Object)$ is defined as:

$$C(Object) = Pr(Object) \cdot IOU(Pred, Truth), \quad (1)$$

where Pr defines whether there is a target object failing into this detection box, and IOU defines the overlapping of the generated candidate bound and ground truth bound, that is, the ratio of their intersection and union, formally defined in equation 2 [32]:

$$IOU(Pred, Truth) = \frac{area(box_{truth}) \cap area(box_{pred})}{area(box_{truth}) \cup area(box_{pred})}. \quad (2)$$

Confidence score $C(Object) = 0$ when it is determined that the detection box does not have a target object. If the $CR(Object)$ is greater than 0, the IOU is calculated. Ideally, the IOU should be close to 1, indicating that the predicted bounding box is close to the ground truth [33].

1) Building image detection model: For the purpose of snail identification, we adopted quite a popular variation of the well-known YOLOv4 model, called YOLOv4-tiny. The YOLOv4-tiny was chosen because it has a very good speed (training)/accuracy (identification) ratio. Using the official YOLOv4 framework, called Darknet [34], we constructed YOLOv4-tiny network, which is composed of 30 layers in total. Since our dataset is relatively small, we utilized a recently quite popular approach to training the model - transfer learning to conduct training of ML algorithms. The basic concept of transfer learning is applying previously obtained knowledge from a different yet similar problem to another problem [15]. In general, conducting training of a model utilizing a transfer learning approach results in faster training and better predictive performance of the model. In our case, we adopted the model weights from trained YOLOv4-tiny against the Common Objects in Context (COCO) [35] dataset.

2) Training object detection model: Typically, when training a CNN, we set a number of epochs, a parameter determining the number of forward and backward passes of the entire dataset through the trained network of an algorithm. Epochs are the number of forward and backward passes of the entire dataset through the trained network. In the case of YOLOv4, the duration of training is defined by a maximum number of batch iterations. The value for batch size commonly varies based on the dataset size; however, it is set to 32 or

64. Additionally, the warm-up phase (commonly set to 1,000 iterations) is introduced, in which the learning rate is slowly increasing to the defined starting learning rate to slowly and gently adapt the network weight and the selected optimizer to training data. The training is executed as defined by the learning rate schedule after the warm-up phase [4].

3) Evaluating object detection model: In order to objectively evaluate the performance of the object detection model, we utilized a commonly used metric for object detection called mean average precision (mAP). The precision-recall presents the trade-off between precision and recall metrics for different thresholds. A high area under the curve indicates both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. The mAP can be formally expressed as:

$$mAP = \frac{\sum_{q=1}^{Q} avgP(q)}{Q}, \tag{3}$$

where Q is the number of queries in the set, and $avgP(q)$ is the average precision for a given query q.

Commonly, the mAP is calculated at a given threshold (value of IOU), which defines at which value the predicted boundary box is to be treated as correctly positioned.

IV. Experiments and results

Our experimental work aimed to explore whether the selected YOLOv4-tiny model can be utilized for slug detection in the real environment. In line with this, we conducted the experiment in which the YOLOv4-tiny neural network was trained on our snail dataset.

A. Experimental setup

The experiments were conducted on a dataset of 396 snail images. The dataset was randomly split into two subsets, i.e., train and test subsets, in a ratio of 90:10, respectively. For the purpose of conducting a validation, the training subset was further randomly divided into train and validation subsets in the process of training, again in a ratio of 90:10, respectively. Thus 318 snail images were used for training and 36 images for validation. The remaining 40 images (i.e., the test set) were used to evaluate the performance of the trained model.

The training process was run for 6,000 batch iterations, validating the performance on every 100 iterations by calculating the validation mAP metric. In the training process, the model with the highest achieved mAP was stored and used for the evaluation phase.

Each image was resized to size 416×416 in the process of training and augmented using the following mechanisms: randomly rotating images and changing saturation, randomly changing the exposure, and randomly changing the hue of the images. The batch size was set to 64, the optimizer momentum to 0.9, decay to $5 \cdot 10^{-4}$ and the learning rate to $2.61 \cdot 10^{-3}$ as are default parameter settings of the darknet framework.

TABLE I: Performance metrics of the trained predictive model at different IOU values.

IOU	mAP	Precision	Recall
0.10	0.957	0.92	0.92
0.20	0.957	0.92	0.92
0.30	0.957	0.92	0.92
0.40	0.957	0.92	0.92
0.50	0.936	0.91	0.91
0.60	0.875	0.88	0.88
0.70	0.842	0.86	0.86
0.80	0.585	0.66	0.66
0.90	0.090	0.22	0.22

B. Results

The trained model was evaluated against the test subset of images, which were not used in the training process. The evaluation of the trained model was conducted against the test subset of images, which were not used in the process of training. This enables us to evaluate the trained model more objectively. In order to obtain more insight into the predictive model's performance, we calculated mAP metrics for different threshold values.

The results of the slug detection method are presented in table I. Focusing on the mAP values, we can observe that, interestingly, even at a small IOU value, the model performs best with the mAP value at 0.957, meaning that it is not falsely detecting more objects as snails, as we would expect. However, after the IOU value started to increase above the 0.5 value, the mAP started rapidly dropping. Nonetheless, even at the default value of IOU (0.5), the model performs well, achieving mAP of 0.936. When comparing our results (mAP values) with a similar research [25], we can see that our approach outperformed the compared one by a margin of 0.245. While the results from the mentioned study did work on a different type of snails with a lower-quality dataset, it is the most similar research we are aware of. Nonetheless, the obtained results from our study show the promising capability of the used approach to successfully identify Arion Rufus snails.

The results are shown in Fig. 2-4. Let us notice that the violet bounding box surrounds the detected slugs. On the top of each bounding box, the label name (in our case, "snail") is displayed together with probability, which indicates the level of certainty on a scale between 0 and 1, where value 1 represents the highest degree of certainty.

C. Discussion

The obtained quantitative results support the idea that the proposed approach for Arion rufus snails detection can be used in real environments. Furthermore, we have also conducted a qualitative analysis, where each test image was manually evaluated, to gain knowledge on how to potentially improve the detection of snails.

Looking at Fig. 2, we can observe that the trained model successfully detected all four snails, even though they were positioned on top of the old tree leaves, which makes them

hard to detect based only on color. Additionally, some of the snails in the picture were also a bit blurry, making the detection even harder. Not only under such hard circumstances, the model is also able to detect the snails in cases when only part of the snail is visible and/or a bit blurry. This can be seen in fig. 3. Also, different angles seem not to disturb the model too much since the snails' positioning varies from image to image.

Interestingly, in Fig. 1c, we can see the falsely detected snail, which was, in fact, a wild strawberry. Two snails were successfully detected even though they are hard to detect for humans visually.

Fig. 2: Example of multiple snails detected in complex visual conditions.

Fig. 3: Example of multiple snails detected at a different angle.

Based on all qualitatively evaluated test images, we can conclude that the model works well regardless of the surroundings; however, there is room for improvement regarding falsely detected snails on a few specific occasions.

V. Conclusion

We have shown that machine learning methods can also be applied in the agriculture domain. In line with this, the Arion Rufus snail detection approach is proposed to help us reduce their number in some pieces of land. Additionally, the dataset containing images of Arion rufus snails was prepared and labeled. We are encouraged by the results of the identification

Fig. 4: Example of multiple snails detected with false detection of wild strawberry.

performance of the trained model and would like to continue to explore if the proposed approach could significantly contribute to knowing when the slugs are active and how many are there. The problem remains if they are covered by a leaf or something similar so they blend in with the surroundings. Furthermore, slugs might only be visible only from one side or not at all.

This is only the preliminary step in our research work. The next steps need to cover more topics. For usage in a real environment, it would be necessary to train a larger model on a bigger dataset. It would also be beneficial to prepare an independent test set, which would help increase the indicative real-life performance of different approaches to detecting Arion rufus snails. This demands taking more and better pictures, i.e., a tripod for the camera is needed so that the images will be clear. The method could be expanded to different types of slugs (pests), and a different object detection model could be applied. An interesting idea is also slug detection based on images taken by night-vision or infrared cameras.

Acknowledgment

The authors acknowledge the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0057).

References

[1] Lucio Colizzi, Danilo Caivano, Carmelo Ardito, Giuseppe Desolda, Annamaria Castrignanò, Maristella Matera, Raj Khosla, Dimitrios Moshou, Kun-Mean Hou, François Pinet, Jean-Pierre Chanet, Gao Hui, and Hongling Shi. Introduction to agricultural iot. In Annamaria Castrignanò, Gabriele Buttafuoco, Raj Khosla, Abdul M. Mouazen, Dimitrios Moshou, and Olivier Naud, editors, *Agricultural Internet of Things and Decision Support for Precision Smart Farming*, pages 1–33. Academic Press, 2020.

[2] Miriam A. Zemanova, Eva Knop, and Gerald Heckel. Phylogeographic past and invasive presence of arion pest slugs in europe. *Molecular Ecology*, 25(22):5747–5764, 2016.

[3] Robert Hollingsworth, M. Kathleen Howe, and Susan Jarvi. Control measures for slug and snail hosts of angiostrongylus cantonensis, with special reference to the semi-slug parmarion martensi. *Hawai'i journal of medicine & public health : a journal of Asia Pacific Medicine & Public Health*, 72:75–80, 06 2013.

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020.

[5] What is computer vision?

[6] Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean, and Richard Socher. Deep learning-enabled medical computer vision. *NPJ digital medicine*, 4(1):1–9, 2021.

[7] Moritz D Lürig, Seth Donoughe, Erik I Svensson, Arthur Porto, and Masahito Tsuboi. Computer vision, machine learning, and the promise of phenomics in ecology and evolutionary biology. *Frontiers in Ecology and Evolution*, 9:642774, 2021.

[8] Vili Podgorelec, Špela Pečnik, and Grega Vrbančič. Classification of similar sports images using convolutional neural network with hyper-parameter optimization. *Applied Sciences*, 10(23):8494, 2020.

[9] Jan Kremer, Kristoffer Stensbo-Smidt, Fabian Gieseke, Kim Steenstrup Pedersen, and Christian Igel. Big universe, big data: machine learning and image analysis for astronomy. *IEEE Intelligent Systems*, 32(2):16–22, 2017.

[10] Arthur Francisco Araújo Fernandes, João Ricardo Rebouças Dórea, and Guilherme Jordão de Magalhães Rosa. Image analysis and computer vision applications in animal sciences: an overview. *Frontiers in Veterinary Science*, 7:551269, 2020.

[11] Boaz Zion. The use of computer vision technologies in aquaculture–a review. *Computers and electronics in agriculture*, 88:125–132, 2012.

[12] Charles J Krebs. *Ecological methodology*. Harper and Row Publishers Inc., 1989.

[13] Stefan Schneider, Graham W Taylor, Stefan Linquist, and Stefan C Kremer. Past, present and future approaches using computer vision for animal re-identification from camera trap data. *Methods in Ecology and Evolution*, 10(4):461–470, 2019.

[14] Md Zahangir Alom, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal, and Vijayan K Asari. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3):292, 2019.

[15] Grega Vrbančič and Vili Podgorelec. Transfer learning with adaptive fine-tuning. *IEEE Access*, 8:196197–196211, 2020.

[16] Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data*, 8(1):1–74, 2021.

[17] Lacey F Hughey, Andrew M Hein, Ariana Strandburg-Peshkin, and Frants H Jensen. Challenges and solutions for studying collective animal behaviour in the wild. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1746):20170005, 2018.

[18] Sachin Umesh Sharma and Dharmesh Shah. *Design and Development of Animal Detection Algorithm Using Image Processing*. PhD thesis, Gujarat Technological University, 2017.

[19] Chiagoziem C Ukwuoma, Zhiguang Qin, Sophyani B Yussif, Monday N Happy, Grace U Nneji, Gilbert C Urama, Chibueze D Ukwuoma, Nimo B Darkwa, and Harriet Agobah. Animal species detection and classification framework based on modified multi-scale attention mechanism and feature pyramid network. *Scientific African*, 16:e01151, 2022.

[20] JD Nichols, KU Karanth, and K Ullas. Camera traps in animal ecology: methods and analyses, 2011.

[21] Kristijn RR Swinnen, Jonas Reijniers, Matteo Breno, and Herwig Leirs. A novel method to reduce time investment when processing videos from camera trap studies. *PloS one*, 9(6):e98881, 2014.

[22] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S Palmer, Craig Packer, and Jeff Clune. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25):E5716–E5725, 2018.

[23] Guobin Chen, Tony X Han, Zhihai He, Roland Kays, and Tavis Forrester. Deep convolutional neural network based species recognition for wild animal monitoring. In *2014 IEEE international conference on image processing (ICIP)*, pages 858–862. IEEE, 2014.

[24] Diego Fabian Collazos Huertas, Gloria Stephany Gómez Gómez, and Andrés Marino Álvarez Meza. Image-based animal recognition based on transfer learning. *Scientia et Technica*, 26(03):406–411, 2021.

[25] Zhiyan Wang, Ivan Lee, Yun Tie, Jinhai Cai, and Lin Qi. Real-world field snail detection and tracking. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1703–1708. IEEE, 2018.

[26] Boris Sekachev, Nikita Manovich, Maxim Zhiltsov, Andrey Zhavoronkov, Dmitry Kalinin, Ben Hoff, TOsmanov, Dmitry Kruchinin, Artyom Zankevich, DmitriySidnev, Maksim Markelov, Johannes222, Mathis Chenuet, a andre, telenachos, Aleksandr Melnikov, Jijoong Kim, Liron Ilouz, Nikita Glazov, Priya4607, Rush Tehrani, Seungwon Jeong, Vladimir Skubriev, Sebastian Yonekura, vugia truong, zliang7, lizhming, and Tritin Truong. opencv/cvat: v1.1.0, August 2020.

[27] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[28] Donggeun Yoo, Sunggyun Park, Joon-Young Lee, Anthony S Paek, and In So Kweon. Attentionnet: Aggregating weak directions for accurate object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2659–2667, 2015.

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[31] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[32] Wang Yang and Zheng Jiachun. Real-time face detection based on yolo. In *2018 1st IEEE international conference on knowledge innovation and invention (ICKII)*, pages 221–224. IEEE, 2018.

[33] Dweepna Garg, Parth Goel, Sharnil Pandya, Amit Ganatra, and Ketan Kotecha. A deep learning approach for face detection using yolo. In *2018 IEEE Punecon*, pages 1–4. IEEE, 2018.

[34] AlexeyAB. Darknet. https://github.com/AlexeyAB/darknet, 2021.

[35] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.