BioAuxNet : Reconnaissance de la faune nocturne vivant au sol à l'aide d'un réseau neuronal convolutionnel à régions masquées

Yassine Sohbi[1]
Corentin Barbu[1]
Antoine Gardarin[1]
Nicolas Parisey[2]

Correspondance
yassine.sohbi@inrae.fr

Résumé.

L'étude de la faune nocturne vivant au sol est une tâche de recherche stimulante en raison de ses multiples implications dans différents domaines tels que la lutte biologique par conservation ou l'identification des ravageurs des cultures. Cet article présente une méthode automatique de reconnaissance de cette faune à partir d'images de terrain acquises sur plusieurs années. Cette méthode, appelée BioAuxNet, est basée sur un algorithme d'apprentissage profond qui combine les dernières avancées dans les domaines des réseaux neuronaux et de la vision par ordinateur.
À partir de plus de 100 000 images brutes prises sur le terrain pendant quatre ans, nous avons créé un premier jeu de données réaliste de 8 espèces communes de la faune nocturne vivant au sol : carabes, souris, opilion, limaces, musaraigne et vers de terre. Pour notre modèle, nous avons classiquement utilisé de l'apprentissage par transfert (affinant un réseau pré-entraîné) et de l'augmentation de données (démultipliant certaines images pour mieux généraliser). Le résultat est que notre modèle peut reconnaître notre type de faune avec un critère de précision de 84,31 %, ce qui est très encourageant. Les perspectives de cette étude sont multiples, on citera typiquement la possibilité d'un suivi au champ du nombre de certains auxiliaires.

Mots-clés

Reconnaissance multi-classe, faune nocturne au sol, biodiversité, deep learning, Mask-RCNN, vision par ordinateur.

1 UMR Agronomie, INRAE AgroParisTech, Université-Paris-Saclay, 22 place de l'agronomie, bâtiment F, F-91123 Palaiseau.
2 UMR IGEPP, INRAE, Domaine de La Motte, F-35653 Le Rheu.

BioAuxNet : Recognition of ground-dwelling nocturnal fauna using a mask region based convolutional neural network

Yassine Sohbi[1]
Corentin Barbu[1]
Antoine Gardarin[1]
Nicolas Parisey[2]

Correspondence
yassine.sohbi@inrae.fr

Abstract

The study of ground-dwelling nocturnal fauna is a challenging research task due to multiple implications in different domains such as the control of pesticide use, the prediction of the crop yields or for the identification of plant diseases. This paper presents an automatic method for recognizing this fauna from field images acquired over several years. This method, named BioAuxNet, is based on a deep learning algorithm which combines the latest advances in the fields of neural networks and computer vision.
Using more than 100,000 raw images taken in the field over four years, we created the first realistic dataset of 8 commons ground-dwelling nocturnal fauna species: ground beetles, mice, opilion, slugs, shrews and earthworms. For our model, we classically used transfer learning (refining a pre-trained network) and data augmentation (multiplying certain images for better generalization). Consequently, our model can recognize the fauna with an accuracy criterion of 84.31% which is very encouraging for its use in real-world situations. The perspectives of this study are manifold, including the possibility of monitoring the numbers of certain beneficials in the field.

Keywords

Multi-class recognition, ground-dwelling nocturnal fauna, biodiversity, deep learning, Mask-RCNN, computer vision.

1 UMR Agronomie, INRAE AgroParisTech, Université-Paris-Saclay, 22 place de l'agronomie, bâtiment F, F-91123 Palaiseau.
2 UMR IGEPP, INRAE, Domaine de La Motte, F-35653 Le Rheu.

Introduction

Le contrôle des ravageurs des plantes et la valorisation de la biodiversité à travers leurs ennemis naturels est un enjeu crucial pour l'agronomie. Cela passe nécessairement par la compréhension de l'impact des pratiques agricoles sur cette biodiversité et, inversement, de l'impact de cette faune sur les rendements, les maladies des plantes et plus généralement sur l'environnement. Nous nous intéressons ici à l'étude de la faune nocturne au sol et à sa détection automatique, car les observations nocturnes in situ sont contraignantes. Parmi les travaux classiques de suivi de terrain et d'étude des parasites/auxiliaires, nous pouvons citer ceux de Grieshop (2012) basés sur des caméras d'acquisition automatique d'images et un traitement manuel de celles-ci.

Traiter les images à l'œil nu est un travail fastidieux et chronophage donc un bon candidat pour une automatisation. Pour mener à bien notre recherche, dans un premier temps, nous avons élaboré sur le terrain plusieurs expériences d'acquisition automatique d'images. Sur une période de 4 ans, nous avons acquis une large base de données de plus de 100 000 images temporelles. Cependant, il est impossible d'exploiter ces images une à une par les méthodes traditionnelles de traitement d'images. Parallèlement, avec l'augmentation de la puissance de calcul des unités de traitement graphique (GPU), de nouvelles méthodes basées sur la vision par ordinateur et par modèle d'apprentissage profond deviennent non seulement opérationnelles mais aussi plus performantes que les méthodes de détection et de classification d'objets (Jiao, 2019). Par conséquent, la solution à notre problème de détection de la faune nocturne vivant au sol est idéale : un modèle d'apprentissage profond. Mais il existe une multitude de méthodes d'apprentissage profond qui peuvent convenir en fonction de leurs avantages et de leurs inconvénients et en fonction de la faune étudiée.
Ceci est rendu difficile par le fait que nos dispositifs expérimentaux ne sont pas standardisés, la luminosité n'est pas très constante, les sols au différents points d'échantillonnage sont de textures variés avec des résidus et divers couverts de végétation. Il en résulte que les individus sont sous ou surexposés, cachés par la végétation ou parfois se confondent avec le sol. Aussi, cette faune est souvent en mouvement et ses formes sont élastiques : la limace ou le ver de terre peuvent s'enrouler sur eux-mêmes (rotation de 360°) ou être dans des positions différentes, comme par exemple la musaraigne grimpant verticalement contre une grille délimitant le dispositif expérimental. Les formes et les tailles sont très disparates entre les individus des différentes classes et sont fonction du stade de développement des individus d'une même classe. Dans ces conditions expérimentales, nous avons décidé de conserver toutes les images dans notre base de données, bonnes ou mauvaises, car elles reflètent les conditions d'utilisation réelles.

La détection d'objets est un des domaines d'étude de la vision par ordinateur et de l'apprentissage automatique. Les méthodes de détection d'objets utilisent différentes techniques. Après une étude de l'existant, notre choix s'est porté sur l'un des frameworks les plus populaires de l'apprentissage automatique : Mask-RCNN (He, 2018) et une de ses implémentations opensource Matterport (Abdulla, 2017) qui préentraîne un modèle pondéré sur l'ensemble de données internationales MS-COCO (Lin, 2014). Cette solution opensource nous a permis d'adapter l'algorithme à nos besoins, pour reconnaître plusieurs classes d'objets par image, dans le cadre d'une chaîne complète qui va du prétraitement des images brutes à la visualisation des résultats et à l'estimation des performances de notre modèle de détection, en passant par l'apprentissage. Cet article détaille cette chaîne.

Dispositifs expérimentaux et construction de l'ensemble d'images

Dans le cadre de divers essais expérimentaux, pour répondre aux besoins de visualisation sur le terrain, nous avons développé notre propre instrument de capture d'images en temps réel, basé sur un nano-ordinateur Raspberry Pi. Ainsi, plusieurs caméras (Bushnell et Berger Et Shröter) ont été installées dans différentes parcelles. Ces caméras sont équipées d'une carte mémoire SD et sont programmées pour prendre une image toutes les quinze secondes. Ces caméras Bushnell et Berger Et Schröter sont contrôlées par un Raspberry Pi, un nano-ordinateur équipé d'un microprocesseur ARM, d'une mémoire RAM de 4 Giga octets, d'une carte vidéo et d'un module wifi. Ce dispositif est facilement transportable sur le terrain (Figure 1, image de gauche), de même qu'une interface web facilite son utilisation.
La carte SD est vidée tous les jours et les données sont enregistrées sur un disque dur de notre serveur d'images. Une image brute prise sur le terrain au format JPEG fait 3 264 par 2 448 de pixels. Nous disposons de 2 dispositifs expérimentaux : PiScope1 (Figure 1, image du milieu), un dispositif d'acquisition d'images avec lequel, de 2016 à 2019, plus de 70 k d'images brutes en couleur RVB et de multiples vidéos ont été prises ; et le second, PiScope2 (Figure 1, image de droite) qui nous a fourni plus de 33 k d'images

Dispositif de capture d'images en accéléré PiScope, basé sur un nano-ordinateur Raspberry Pi.

PiScope1 : L'abondance de la faune nocturne au sol, ici des limaces, entraîne la disparition quasi systématique de toutes les proies apportées.

PiScope2 : Activité de vers de terre en surface, en particulier autour des débris de navette délibérément déposés à la surface du sol.

Figure 1. Dispositifs expérimentaux sur le terrain

brutes sur les deux années 2017-2018. Ces images étant très volumineuses (entre 2 et 3,5 Mégaoctets), une première étape a consisté à réduire leur taille sans perte d'information. Cette phase de réduction de la taille des images est nécessaire, les algorithmes de deep learning travaillant sur des représentations de ces images en mémoire centrale ou sur GPU. Nous avons redimensionné les images de manière à conserver le ratio d'aspect. La réduction de la taille des images brutes a été effectuée en fonction de leur rapport largeur/hauteur afin de conserver les mêmes proportions des individus les uns par rapport aux autres. Pour ce faire, toutes les images réduites ont eu une largeur fixe de 1 000 pixels. Résultat : les images réduites ont des tailles qui varient entre 14,7 et 750 ko. Le tableau 1 nous donne une idée de la taille des individus, du plus petit au plus grand, et de la proportion de chaque individu par rapport à la taille de l'image qui le contient. En fonction du contenu de nos images, nous avons mis l'accent sur les ravageurs/auxiliaires suivants : carabe, limace, vers de terre, musaraigne, opilion, petit carabe, petite limace et souris.

Annotation et augmentation d'image

Nous avons ensuite annoté ces images réduites en utilisant l'application python LabelImg. Dans le cas où le nombre

Tableau 1: Proportion de chaque individu par rapport à la taille de l'image qui le contient

	Taille minimale d'un individu en pixels	Ratio minimal (%)	Taille maximale d'un individu en pixels	Ratio maximal (%)
Carabe	2760	0,23	28200	0,56
Limace	3120	0,41	911790	18,09
Lombric	720	0,09	226324	30,17
Musaraigne	38150	5,08	850450	16,87
Opilion	16884	0,33	39360	0,78
Petit carabe	1040	0,02	6138	0,12
Petite limace	248	0,02	19822	0,39
Souris	18984	2,53	64064	4,29

On constate que le ratio taille (ici, la taille de l'image est le produit hauteur*largeur) des carabes, des opilions, des petits carabes et des petites limaces varie de 0,02 % jusqu'à 0,78 % de la taille de l'image. Soit moins de 1 % de la taille de l'image. En revanche, une limace peut occuper 18 % de l'image et le lombric même 30 %.

d'un ravageur/auxiliaire est insuffisant pour le processus d'apprentissage, nous avons créé de nouvelles images en utilisant des techniques d'augmentation d'image tout en respectant les contraintes de déplacement de ces individus dans l'espace.

Une grande différence par rapport aux images « de laboratoire » où l'individu est entièrement visible avec tous les détails concernant ses organes, les individus dans nos images peuvent être cachés par la végétation, dans une scène peu éclairée ou simplement à la limite du cadre de l'image. Pour entraîner l'algorithme, il faut lui fournir des données avec des images correctement annotées. Ce processus d'annotation peut être très lent. Dans cette phase d'annotation, nous nous sommes efforcés de maintenir des règles d'annotation claires pour garantir l'apprentissage le plus précis et donc un meilleur modèle de détection. Ces règles peuvent être schématisées comme suit :

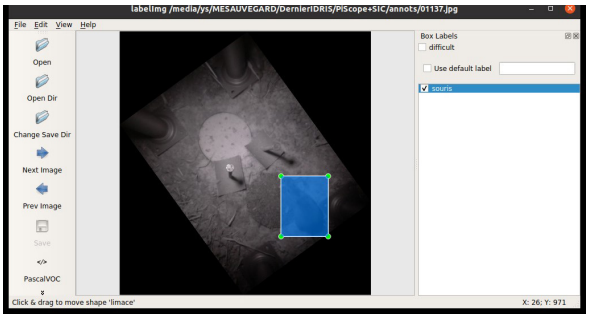
- annotation d'un individu entier
- annotation des individus cachés
- création de boîtes de délimitation qui entourent les individus aussi étroitement que possible.

De cette manière, nous avons annoté 1 644 images PiScope1 et 711 images PiScope2, soit un total de 2 355 images annotées. Au final, et par augmentation d'image, nous avons construit un dataset de 7 470 images. La Figure 2 montre quelques exemples d' annotation de la faune nocturne vivant au sol.

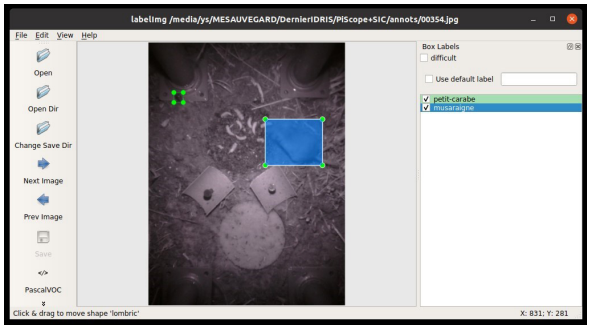
Une limace, une petite limace et un opilion

Un opilion

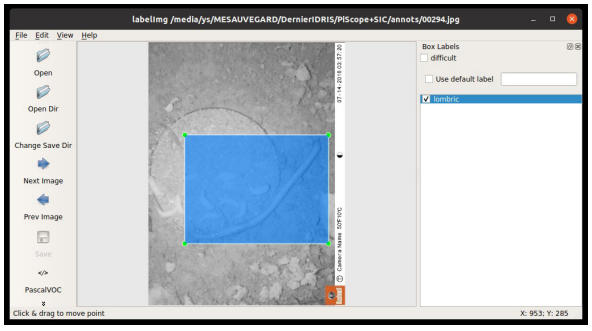
Une souris



Une musaraigne et un petit carabe



Un vers de terre



Une limace

Figure 2. Quelques exemples d'annotation de la faune nocturne vivant au sol

Cycle d'apprentissage itératif

Pour des raisons d'efficacité et de reproductibilité, nous avons construit deux chaînes de scripts python, l'une pour la phase d'apprentissage et l'autre pour la phase de visualisation et de validation. La première chaîne pour la phase d'apprentissage peut être décrite par une séquence linéaire d'opérations telles que : (Entrée), (Redimensionnement), (Annotation), (Augmentation), (Apprentissage), (Prédiction), (Évaluation). Et la deuxième chaîne est la suivante : (Entrée), (Prédiction), (Évaluation). Le paradigme du développement du Deep Learning (DL) est différent du développement informatique classique : ici, le développeur agit davantage sur les données d'image et/ou affine les valeurs des paramètres d'apprentissage (méta-paramètres). L'apprentissage profond nécessite de nombreux ensembles de données d'entraînement. Plus le nombre de paramètres est élevé, plus le nombre d'individus et la taille de l'ensemble de données doivent être augmentés. Le nombre minimum d'individus par classe devrait être d'environ 1 000 ou plus, mais c'est rarement le cas. L'étape « Augmentation » est facultative ; elle est donc appelée seulement si le nombre d'individus par classe est faible. Ce cycle d'opérations séquentielles est répété autant de fois que nécessaire jusqu'à ce qu'un niveau acceptable de précision de détection soit atteint.

Métriques d'évaluation

Dans la classification, nous mesurons la performance d'un modèle en vérifiant la précision de ses prédictions. Tout algorithme d'apprentissage automatique pour la classification fournit des résultats sous forme de probabilité, c'est-à-dire la probabilité qu'une instance appartienne à une classe particulière. Pour attribuer une classe à une instance dans le cadre d'une classification binaire, nous comparons la valeur de la probabilité au seuil, c'est-à-dire que nous déterminons si la valeur est supérieure ou inférieure au seuil. Si la probabilité est supérieure au seuil, l'individu appartient à la classe, sinon il n'appartient pas à la classe. Pour mesurer la précision d'une prédiction sur une image, nous définissons le ratio de chevauchement IoU (Intersection over Union) qui est donné par le ratio de la zone d'intersection et de la zone d'union de la boîte de délimitation prédite et de la boîte de délimitation de la vérité de terrain :

$$IoU = \text{zone d' intersection} / \text{zone d' union}$$

En général, le seuil de l'indice de confiance est fixé à 0,5. L'IoU est utilisé pour déterminer si une boîte englobante prédite est **TP, FP, FN** ou **TN** :

- TP (True Positive ou Vrai positif), si IoU ≥ 0,5 et si le label prédit correspond bien à l'objet de vérité de terrain.
- FP (False Positive ou Faux positif), nous avons 2 cas de figure possibles (Figure 3) :
  1. IoU < 0.5 ou
  2. un ou plusieurs individus sont détectés plusieurs fois.
- **FN** (False Negative ou Faux Négatif), nous avons deux cas de figure possibles :
  1. pas de détection du tout alors que l'objet est présent dans l'image.
  2. IoU ≥ 0,5 mais l'objet est mal classé.
- **TN** (True Negative ou Vrai Négatif). Ce cas suppose qu'il n'y ait pas d'objet à l'intérieur d'une boîte englobante. Ce cas n'existe pas car on suppose que si l'annotateur crée une boite, cette boite n'est jamais vide, elle contient forcément un individu.

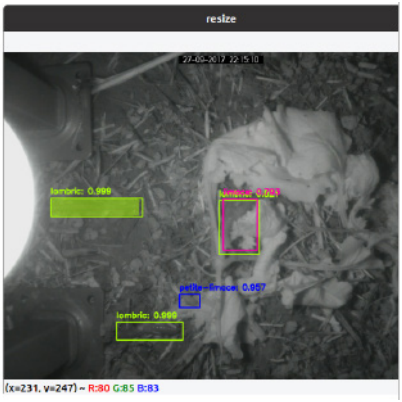


Figure 3. Exemple de problème de détection multiple. Un vers de terre dans la vérité terrain est prédit à la fois comme un vers de terre (0.823) et comme une limace (0.793). En couleur blanche, des feuilles de navet ont été disposées sur le sol.

Précision et Rappel

La précision peut être définie comme le pourcentage de prédictions correctes faites par notre modèle de classification. Pour chaque classe, la formule est la suivante :

$$\text{Précision} = TP / (TP + FP)$$

La Précision n'est pas toujours une mesure de performance adéquate. C'est une bonne mesure à utiliser lorsque les classes sont équilibrées, c'est-à-dire lorsque la proportion d'instances de toutes les classes est quelque peu similaire.
Le Rappel est une autre métrique permettant de déterminer le nombre de prédictions correctement faite par le modèle. Le Rappel consiste à trouver et à localiser tous les objets d'une classe:

$$\text{Rappel} = TP / (TP + FN) \text{ (relation 1)}$$

Un bon modèle d'apprentissage présente à la fois une bonne Précision et un bon Rappel.

mAP (mean Average Precision)

La mAP est la moyenne de l' ensemble des Précisions de chaque classe. De même, la mAR (mean Average Recall) est la moyenne de l'ensemble des Recalls de chaque classe.

Matrice de confusion

Une matrice de confusion est un tableau utilisé pour mesurer les performances d'un algorithme d'apprentissage automatique. Chaque ligne de la matrice de confusion représente les instances d'une classe réelle et chaque colonne représente les instances d'une classe prédite. Un élément de la matrice de confusion MCij exprime le nombre de fois où un objet de la classe i est prédit comme étant un objet de la classe j. Les éléments situés sur la diagonale, c'est-à-dire MCii, sont les éléments bien classés ou bien prédits. À partir de là, nous pouvons exprimer la somme des colonnes lorsque nous fixons une ligne et la somme des lignes lorsque nous fixons une colonne. Pour une classe I de la vérité de terrain :

$$\sum \text{colonnes} = TP + FN$$

D'où le nombre de FN :

$$FN = \sum \text{colonnes} - TP$$

D'autre part, pour une classe j prédite :

$$\sum \text{lignes} = TP + FP$$

D'où le nombre FP :

$$FP = \sum \text{lignes} - TP$$

Par conséquent, connaissant les valeurs de FP et FN, nous pouvons calculer la précision et le rappel pour chaque classe I comme dans la relation (1) :

$$\text{Précisioni} = Mii / \sum jMji = Mii / \sum \text{lignes}$$
$$\text{Recalli} = Mii / \sum jMij = Mii / \sum \text{colonnes}$$

Sachant que lorsque FP tend vers 0, la précision est maximale, c'est-à-dire 100 %, la précision est donc un bon indicateur du taux de FP. En revanche, lorsque FN tend vers 0, le Recall est maximal, c'est-à-dire 100 %. Le rappel est un bon indicateur du taux de FN.

Matériels et logiciels

Mask-RCNN est écrit en Python et est basé sur les frameworks Keras et TensorFlow. En sortie, il génère des boîtes de délimitation et des masques de segmentation pour chaque instance d'un objet détecté dans l'image. Nous avons souvent utilisé différentes versions des bibliothèques OpenCV, Scikit, Numpy, ... dans un environnement virtuel Conda pour gérer la compatibilité entre les différentes versions de ces bibliothèques. Notre chaîne de développement logicielle est constituée de l'écosystème python : keras, tensorflow, maskrcnn, opencv, numpy, matplotlib, conda, anaconda, pip.

Le temps d'apprentissage est très lent, voire prohibitif, sur un ordinateur ordinaire, mais il est très confortable sur un HPC (High Performance Computing). En revanche, une fois l'apprentissage effectué, le modèle de détection peut être utilisé sur un ordinateur portable ordinaire. Pour ce travail, nous avons bénéficié de 2 000 heures de calcul sur GPU du centre de ressources informatiques et d'expertise en HPC et IA, l'IDRIS (www.idris.fr), une unité de service du Centre National de la Recherche Scientifique (CNRS).

Une itération d'apprentissage (1 EPOCH) dure entre 35 et moins de 55 minutes (selon que l'on fait l'apprentissage en se limitant aux couches d'en-tête ou pour toutes les couches du réseau de neurones) pour notre dataset de 7 470 images. Sur un serveur graphique dédié, comme celui de notre unité, cette itération dure plus de 18 heures. Soit un ratio de 36 ! Sur un ordinateur portable ordinaire, ce rapport est d'au moins 100.

Validation du modèle

Comme nous pouvons le voir sur la Figure 4, la matrice de confusion donne de bons scores de détection. Pour le carabe nous avons 98,55 % de précision et 90,82 % de rappel, et pour le vers de terre (lombric) 78,94 % de précision et 87,07 % de rappel. La précision globale ou mAP (Mean Average Precision) est de 84,31 %.

À partir du modèle d'apprentissage BioauxNet, nous avons annoté automatiquement plusieurs datasets d'images brutes prises au champ. Les résultats sont stockés dans un fichier texte CSV avec les colonnes suivantes : identificateur du dataset, nom de l'image, et une colonne pour chaque classe (carabe, …, souris) et un indicateur de présence ou non. La date et l'heure sont aussi enregistrées. À partir de ce fichier CSV, nous pouvons avoir des sorties telles que le comptage des individus sur une période donnée (Figure 5) ou encore la dynamique (temporalité) des individus (Figure 6).

Figure 4. Matrice de confusion d'un dataset de 7 470 images et de 8 classes y compris la classe sol (ground). Cette dernière n'est pas prise en compte dans le calcul de la mAP. L'axe horizontal représente les observés(vérité de terrain) et l'axe vertical les prédictions. La ligne en bas de l'image donne la précision et les effectifs pour chaque classe et, respectivement, la colonne à droite donne le rappel et les effectifs. Les TP sont sur la diagonale en vert. Nous pouvons constater, par exemple, qu'une musaraigne est prédite comme souris dans 0,43 % des cas (FN).

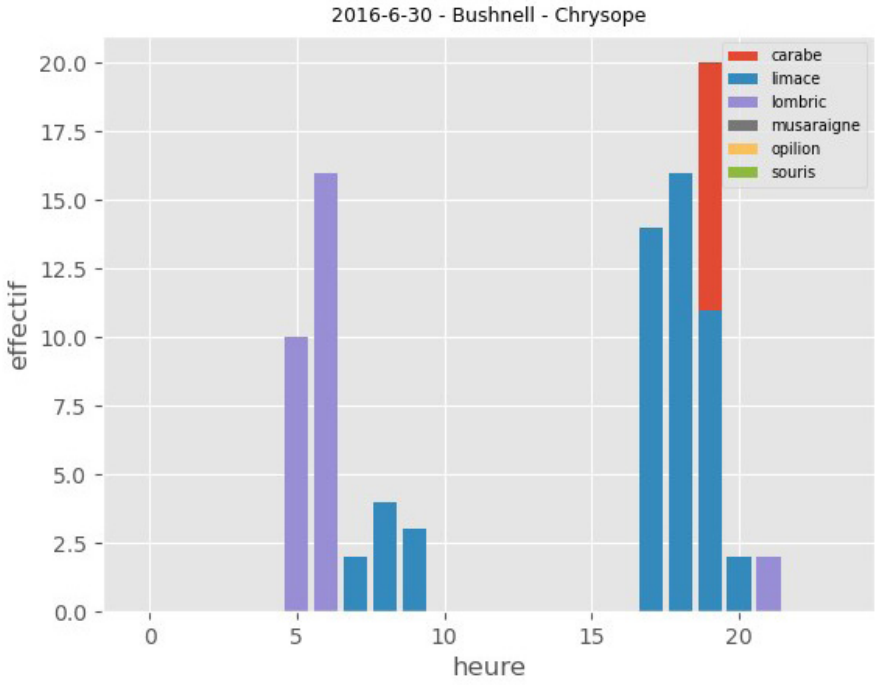


Figure 5. Comptage des individus, sur 24 H, d'un dataset annoté automatiquement : "2016-6-30- Bushnell – Chrysope"

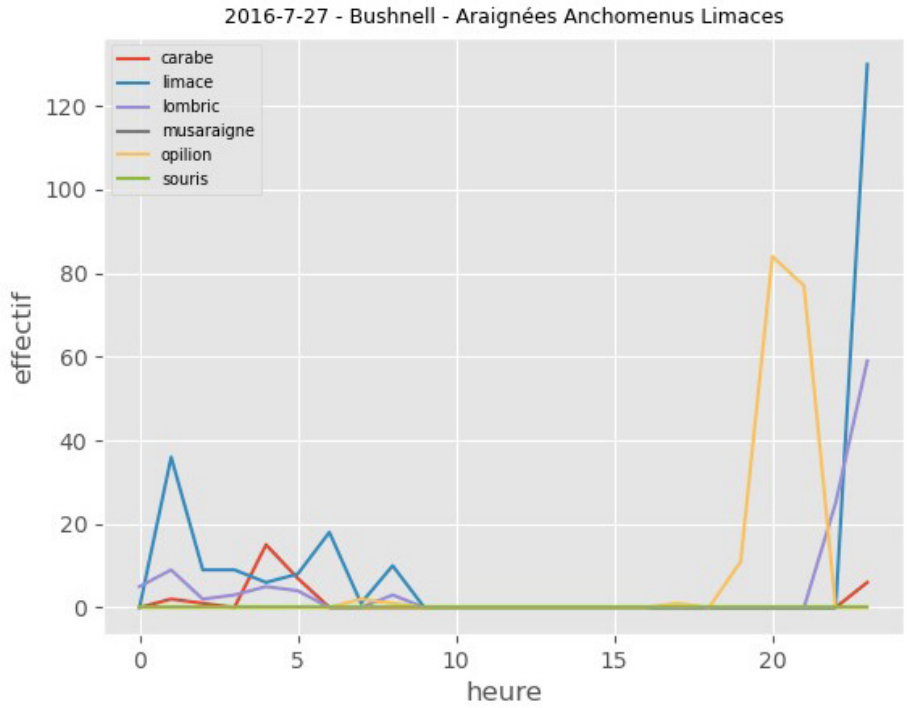


Figure 6. Dynamique, sur 24H, des individus d'un dataset annoté automatiquement : "2016-7-27- Bushnell – Araignées Anchomenus Limaces"

Conclusion et perspectives

Notre objectif était de construire un modèle de détection, au champ, de la faune nocturne vivant au sol. Pour ce faire, nous avons construit une chaîne logicielle qui va de l'acquisition d'images à la reconnaissance d'insectes en passant par l'annotation d'images, l'augmentation d'images, le processus d'apprentissage jusqu'à l'estimation des performances du modèle. Ce modèle, que nous avons appelé BioAuxNet, reste efficace même avec des individus de petite taille comme le petit carabe, la petite limace ou l'opilion, si le nombre d'individus par classe est conséquent. En outre, nous avons créé un dataset à grande échelle, de 7 470 images annotées, des principaux auxiliaires et ravageurs de nos cultures les plus courantes, à partir de milliers d'images acquises dans le cadre de deux expérimentations sur le terrain.

Pour améliorer encore la précision des prédictions de certaines classes de notre modèle dans des situations réelles, il faut l'alimenter d'images en plus grand nombre et de meilleure qualité. Grâce aux réseaux neuronaux et à la vision par ordinateur, la détection précoce de la faune nocturne vivant au sol, tant qualitative que quantitative, permettra de trouver les solutions les plus appropriées pour une agriculture plus durable.

Par ailleurs, notre chaine logicielle nous sert actuellement de base pour développer un modèle de reconnaissance d'oiseaux nuisibles au champ comme les corneilles et les pigeons ■

Remerciements

Ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS à travers l'allocation de ressources 2021-(AD011011832R1) attribuée par GENCI.

Références

He K., Georgia, Gkioxari G., Dollar P., Girshick R. Facebook AI Research (FAIR), Mask R-CNN. Cornell University, 2018. Disponible sur : https://arxiv.org/abs/1703.06870.

Abdulla W. Matterport_maskrcnn_2017, Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. 2017. Disponible sur : https://github.com/matterport/Mask_RCNN.

Lin T., Maire M., Belongie S., Bourdev L., Girshick R., Hays J., Perona P., Ramanan D., Zitnick C., Dollar P. Microsoft COCO: Common Objects in Context. European conference on computer vision, 2014/9/6 ; Pages 740-755. Springer, Cham editor.

Grieshop M.J., Werling B., Buehrer K., Perrone J., Isaacs R., Landis D. Big Brother is Watching: Studying Insect Predation in the Age of Digital Surveillance. American Entomologist; Volume 58; Number 3; 2012.

Collett R.A., Fisher D.O. Time-lapse camera trapping as an alternative to pitfall trapping for estimating activity of leaf litter arthropods. Ecology Evolution, 2017,7:7527–7533. Disponible sur : https://doi.org/10.1002/ece3.3275.

Jiao L., Zhang F., Liu F., Yang S., Li L., Feng Z. & al. A survey of deep learning-based object detection. IEEE Access, 2019, arXiv:1907.09408v2.