Automatisation de la reconnaissance d'espèces animales dans des vidéos de pièges photographiques installés dans les forêts tropicales en Afrique centrale, grâce à l'apprentissage profond

Auteur : Campers, Harold
Promoteur(s) : Lejeune, Philippe; Delplanque, Alexandre
Faculté : Gembloux Agro-Bio Tech (GxABT)
Diplôme : Master en bioingénieur : sciences et technologies de l'environnement, à finalité spécialisée
Année académique : 2021-2022
URI/URL : http://hdl.handle.net/2268.2/15521

LIÈGE université
Gembloux
Agro-Bio Tech

Automatisation de la reconnaissance d'espèces animales dans des vidéos de pièges photographiques installés dans les forêts tropicales en Afrique centrale, grâce à l'apprentissage profond

Harold Campers

Travail de fin d'études présenté en vue de l'obtention du diplôme de Master bioingénieur en sciences et technologies de l'environnement

Année académique 2021 - 2022

Co-Promoteurs : Pr. Philippe Lejeune, Ir. Alexandre Delplanque

1. Dans ce cas, l'autorité académique est représentée par le(s) promoteur(s) membre du personnel(s) enseignant de GxABT

LIÈGE université
Gembloux
Agro-Bio Tech

Automatisation de la reconnaissance d'espèces animales dans des vidéos de pièges photographiques installés dans les forêts tropicales en Afrique centrale, grâce à l'apprentissage profond

Harold Campers

Travail de fin d'études présenté en vue de l'obtention du diplôme de Master bioingénieur en sciences et technologies de l'environnement

Année académique 2021 - 2022

Co-Promoteurs : Pr. Philippe Lejeune, Ir. Alexandre Delplanque

Avant-propos

Les scripts qui ont permis d'entraîner et d'exploiter les différents modèles d'apprentissage profond présentés dans ce travail, ont été rendus disponibles dans un répertoire GitHub : `https://github.com/Harold0690/TFE_camera_traps_videos`. De même, l'environnement python utilisé a également été mis à disposition dans ce répertoire.

Remerciements

Tout d'abord, je voudrais sincèrement remercier mon co-promoteur Philippe Lejeune d'être mon encadrant académique et de m'avoir donné l'opportunité de travailler sur ce sujet de mémoire.

Je tiens évidemment à remercier profondément mon co-promoteur Alexandre Delplanque pour son expertise en apprentissage profond (*deep learning*) et en vision d'ordinateur (*computer vision*). Il m'a transmis de nombreux savoirs tout au long de la réalisation de ce mémoire. Ses remarques, conseils et commentaires ont toujours été très utiles et très pertinents.

Merci à Alexandre Delplanque, Benoît Mercatoris, Catherine Charles (présidente), Philippe Lejeune et Hélène Soyeurt d'avoir accepté de faire partie de mon jury et d'évaluer ce mémoire.

Je remercie grandement le centre de coopération internationale en recherche agronomique pour le développement (Cirad) et le département de forêt tropicale de l'université de Gembloux Agro-Bio Tech de m'avoir fourni les données qui sont à la base de mon travail.

Je souhaite également remercier particulièrement Davy Fonteyn pour avoir été le représentant du Cirad et du département de forêt tropicale de l'université de Gembloux Agro-Bio Tech. Il s'est rendu fort disponible et m'a apporté une grande aide, notamment grâce à ses connaissances sur les pièges photographiques et sur la faune et la flore d'Afrique centrale.

Mes remerciements vont aussi à toutes les personnes qui travaillent au département de forêt tempérée de l'université de Gembloux Agro-Bio Tech dans lequel j'ai eu la chance de réaliser mon mémoire. Je les remercie pour leur aide, leur bienveillance et leur gentillesse.

Ensuite, je remercie tous mes amis pour leur aide et leur soutien. Ils m'ont permis de m'épanouir que ce soit dans le cadre des activités académiques ou en dehors. Ils font partie de ma réussite scolaire et ont rendu ces 5 années d'études inoubliables. Un merci particulier à Bambi, Beaumont, Castillo et Daubry qui ont toujours été présents durant mon cursus.

Je remercie infiniment mes parents pour avoir cru en moi, m'avoir soutenu et m'avoir permis de faire ces études. Merci également à mes 2 frères Guillaume et Lambert pour les moments partagés ensemble et ceux à venir.

Pour terminer, j'aimerais remercier ma Didiche, qui partage ma vie depuis plus de 4 ans. Merci de toujours croire en moi, de m'encourager sans cesse et d'être toujours là pour moi.

Résumé

Le monde actuel est menacé par une crise de la biodiversité dramatique. Il devient donc primordial de surveiller les populations animales et végétales qui habitent les écosystèmes de la Terre. Dans ce sens, les pièges photographiques sont des caméras qui capturent des images ou des vidéos lorsqu'elles détectent un mouvement. Ces caméras sont de plus en plus utilisées dans le monde scientifique et pourraient devenir un outil essentiel dans des systèmes de surveillance de la faune et de la flore. Elles possèdent notamment l'avantage d'être très peu intrusives et de pouvoir être installées dans des endroits reculés et difficilement accessibles. Le point faible de cette technologie est qu'elle génère une quantité très importante de données. L'analyse de ces dernières par l'homme est donc très chronophage et fastidieuse. Une solution pourrait être apportée à ce problème grâce à l'utilisation de l'apprentissage profond. Celui-ci permet d'entraîner des réseaux de neurones profonds afin d'automatiser une tâche habituellement réalisée par l'homme. Certaines approches d'apprentissage profond ont permis d'atteindre de meilleurs résultats lors la résolution de problèmes complexes. L'objectif principal de ce travail est donc d'utiliser l'apprentissage profond afin d'automatiser la reconnaissance d'espèces animales dans des vidéos de pièges photographiques installés dans les forêts tropicales d'Afrique centrale. Pour ce faire, trois jeux de données ont été constitués et 22 classes ont été définies. Différentes architectures de modèles ont ensuite été testées. Ces dernières sont composées de réseaux de neurones convolutifs (ResNet à deux dimensions et à trois dimensions) et de réseaux de neurones récurrents (mémoire convolutive à long court terme (ConvLSTM) et mémoire à long court terme (LSTM)). Ce travail aborde également la comparaison de différents outils qui ont été développés afin de classifier automatiquement des données de pièges photographiques. Les meilleurs modèles entraînés ont atteint, sur un jeu de données de test, une exactitude globale de 67,93 % pour la classification multi-espèces et de 84,89 % pour la classification binaire (animal / arrière-plan). Ces modèles ont mieux performés que les autres outils testés, pour la classification multi-espèces mais pas pour la classification binaire. Enfin, les modèles développés pourraient être utilisés sous certaines conditions dans le but d'aider à l'analyse des données de pièges photographiques. Les résultats obtenus sont prometteurs.

Mots-clés : piège photographique, forêt tropicale, apprentissage profond, réseau de neurones convolutifs, mémoire convolutive à long court terme, classification, faune sauvage.

Abstract

The world today is threatened by a dramatic biodiversity crisis. It is therefore becoming essential to monitor the animal and plant populations that inhabit the earth's ecosystems. In this sense, camera traps are cameras that capture images or videos when they detect movement. These cameras are increasingly used in the scientific world and could become an essential tool in wildlife monitoring systems. They have the advantage of being very low-intrusive and of being able to be installed in remote and difficult-to-access places. The main weakness of this technology is that it generates a huge amount of data. The analysis of this data by humans is therefore very time-consuming and tedious. A solution to this problem could be found in the use of deep learning. This allows deep neural networks to be trained to automate a task usually performed by humans. Some deep learning approaches have achieved better results in solving complex problems. The main objective of this work is therefore to use deep learning to automate the recognition of animal species in videos of camera traps installed in the tropical rainforests of Central Africa. To this end, three datasets were created and 22 classes were defined. Different model architectures were then tested. These are composed of convolutional neural networks (two-dimensional and three-dimensional ResNet) and recurrent neural networks (convolutional long short-term memory (ConvLSTM) and long short-term memory (LSTM)). This work also discusses the comparison of different tools that have been developed to automatically classify camera traps data. The best trained models achieved, on a test dataset, an overall accuracy of 67,93 % for multispecies classification and 84,89 % for binary classification (animal/background). These models performed better than the other tested tools for the multispecies classification but not for the binary classification. Finally, the models developed could be used under certain conditions to assist in the analysis of camera traps data. The results obtained are promising.

Keywords : camera trap, tropical forest, deep learning, convolutional neural network, convolutional long short-term memory, classification, wildlife.

Table des matières

Liste des figures

Liste des tables

Nomenclature

AP = Apprentissage profond
Cirad = Centre de coopération internationale en recherche agronomique pour le développement
CB = Classification binaire
CM = Classification multi-espèces
CNN = Réseau de neurones convolutifs ou *convolutional neural network*
ConvLSTM = *Convolutional long short-term memory*
DFT de GxABT = Département de forêt tropicale de l'université de Gembloux Agro-Bio Tech
GRU = *Gated recurrent unit*
LR = Taux d'apprentissage ou *learning rate*
LSTM = *Long short-term memory*
PP = Piège photographique
RGB = Rouge, vert, bleu ou *red, green, blue*
RNN = Réseau de neurones récurrents ou *recurrent neural network*

1 Introduction

1.1 Théorie et définitions

1.1.1 Pièges photographiques

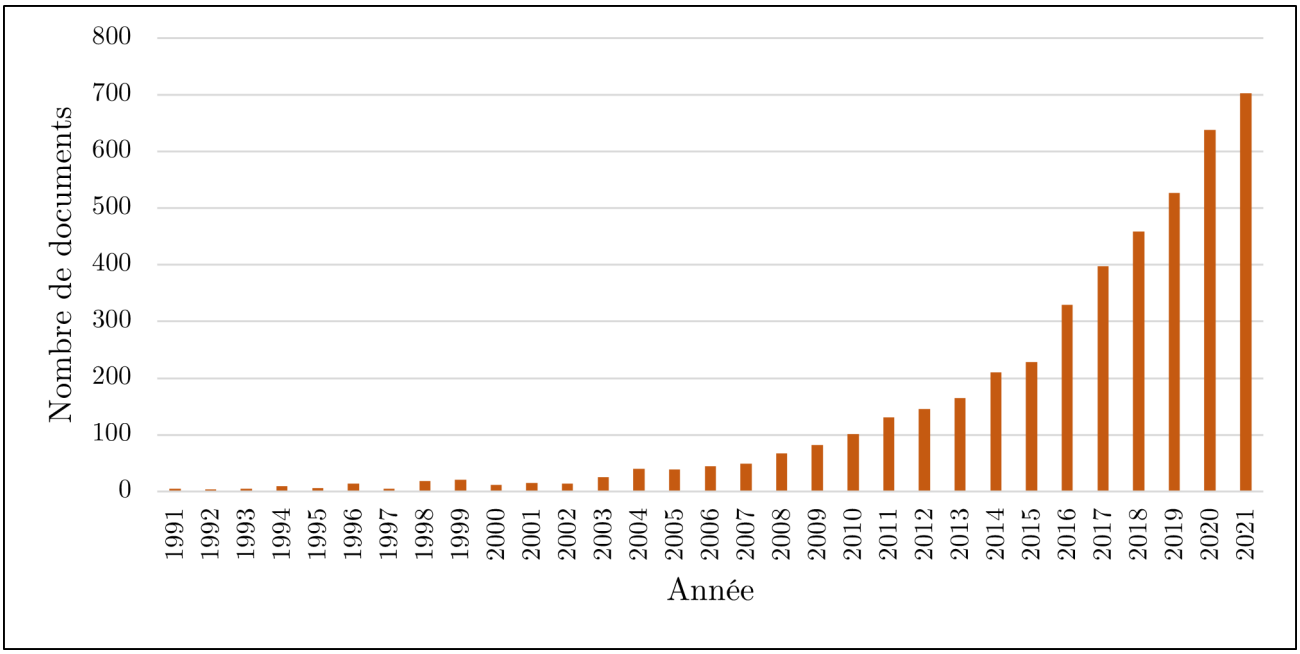
Ces dernières années, la biodiversité mondiale a subi des dégradations sans précédent (Almond *et al.*, 2020). De nombreuses espèces animales ou végétales sont en danger. La gestion et la protection de la faune et de la flore est un des défis majeurs de notre époque. Pour répondre à cette problématique, les pièges photographiques (PP) pourraient devenir un outil indispensable afin de surveiller la faune et la flore (Loos *et al.*, 2018).

Les PP sont des caméras qui se déclenchent automatiquement par l'activation d'un capteur de mouvements. Celui-ci peut être un capteur infrarouge, laser ou à micro-ondes, une plaque de pression ou un fil tendu. Cependant, les caméras modernes sont le plus souvent équipées d'un capteur infrarouge passif de mouvements qui détecte le rayonnement infrarouge émis par les animaux à sang chaud (Apps et McNutt, 2018). Ces dispositifs peuvent être placés dans tout environnement que ce soit en zone urbaine ou dans des endroits plus reculés comme dans une forêt tropicale. Ils sont principalement installés afin de capturer des images ou des vidéos des animaux qui passent dans leur champ de vision (Wearn et Glover-Kapfer, 2017).

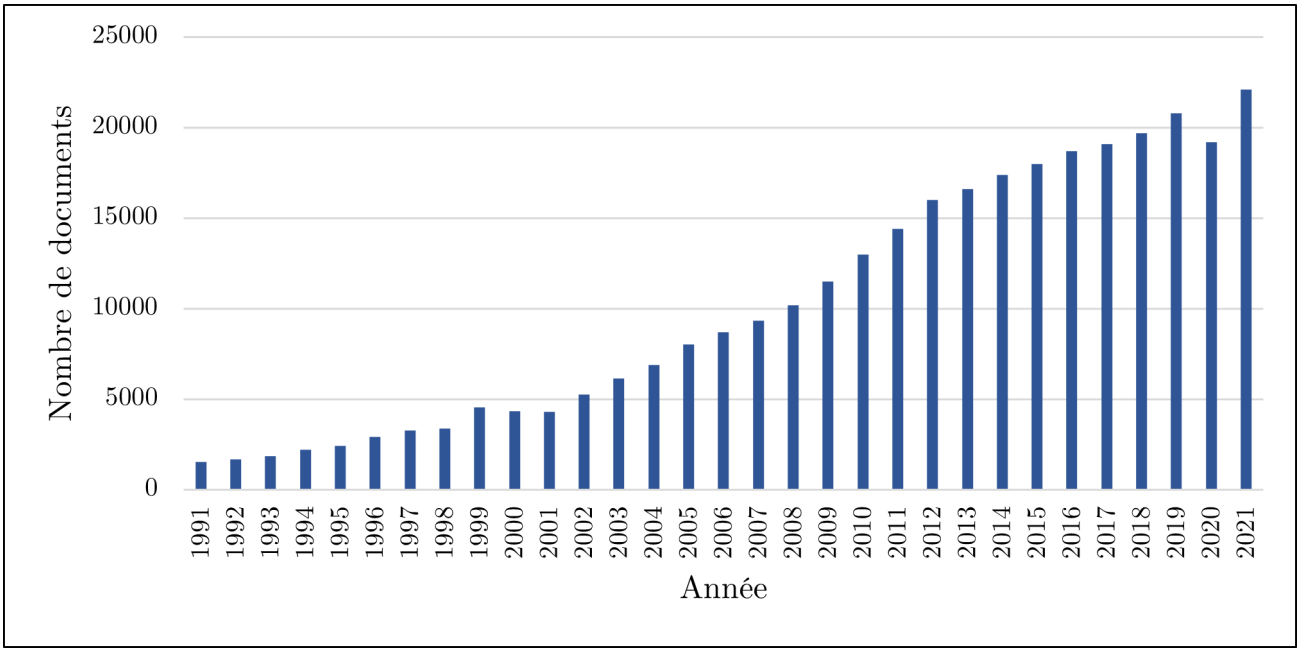
Cet outil permet un échantillonnage continu dans des zones qui peuvent être difficilement accessibles (Trolliet *et al.*, 2014). Il est peu intrusif et permet d'étudier à la fois des espèces animales communes et à la fois des espèces animales élusives et rares (O'Connell *et al.*, 2011; Sollmann, 2018). L'utilisation de PP constitue une alternative économiquement intéressante et plus éthique que les autres méthodes de surveillance du monde sauvage comme les relevés aériens, le piégeage d'animaux vivants et la radiotélémétrie (Meek *et al.*, 2014).

Au cours des dernières années, les PP ont été de plus en plus utilisés par les scientifiques dans leurs recherches (O'Connell *et al.*, 2011; Meek *et al.*, 2014; Trolliet *et al.*, 2014; Sollmann, 2018). L'augmentation, d'année en année, du nombre de publications scientifiques concernant ces caméras en est une preuve (figure 1). Les applications qui découlent de cette technologie sont très nombreuses : étude de la présence, de l'abondance ou de la densité d'espèces animales à un endroit donné (Kalle *et al.*, 2011; Garrote *et al.*, 2012; Oliveira-Santos *et al.*, 2012; Liu *et al.*, 2013; Assou *et al.*, 2021), études comportementales (Soley et Alvarado-Díaz, 2011; Hénaut et Charruau, 2012; Oliveira-Santos *et al.*, 2012; Luo *et al.*, 2019; Assou *et al.*, 2021), observation des interactions animal-plante (Babweteera et Brown, 2010; Nyiramana *et al.*, 2011; Campos *et al.*, 2012; Koike *et al.*, 2012; Pender *et al.*, 2013), description de l'habitat de certains animaux (Moruzzi *et al.*, 2002; Blake *et al.*, 2010; Gil-Sánchez *et al.*, 2011; Gray et Channa, 2011; Luo *et al.*, 2019) et analyse de l'organisation sociale des espèces animales (Lopucki, 2007; Srbek-Araujo *et al.*, 2012). Cette liste n'est pas exhaustive mais montre bien l'étendue des possibilités que peut offrir l'utilisation de pièges photographiques.

Malgré tout, une limitation de ces caméras est qu'elles génèrent un nombre très important d'images ou de vidéos. Par conséquent, l'analyse de ces données est assez longue, fastidieuse et fatigante pour l'homme (Petso *et al.*, 2021). Cette problématique motive ce travail.



(a)



(b)

Figure 1 – Nombre de documents disponibles sur Scopus (a) et Google Scholar (b), par année, entre 1991 et 2021.

1.1.2 Apprentissage profond

L'apprentissage profond (AP) est une branche de l'intelligence artificielle. Cette dernière se définit comme l'ensemble des techniques qui servent à automatiser une tâche intellectuelle qui est normalement performée par l'homme. L'AP utilise des réseaux neuronaux profonds afin d'automatiser ces tâches. Ces réseaux sont des successions de nombreuses couches de neurones. Plus le nombre de couches qui constituent un réseau est important, plus il est profond (Chollet, 2021). Le progrès au niveau des systèmes informatiques et l'augmentation de la taille des jeux de données ont permis des avancées importantes dans le domaine de l'AP (LeCun *et al.*, 2015; Wu *et al.*, 2019; Chollet, 2021).

Lors de l'entraînement d'un modèle d'AP, ce dernier reçoit des *inputs* et les *outputs* qui leur correspondent et il recherche les relations qui existent entre ceux-ci. Il apprend par lui-même le moyen d'arriver à la bonne solution en partant d'une donnée en entrée. En d'autres mots, il cherche la bonne représentation des données qui va permettre d'automatiser une tâche intellectuelle. Pour ce faire, le modèle va être entraîné sur plusieurs époques. Une époque correspond au passage de l'entièreté d'un jeu de données dans un modèle lors de son entraînement (Chollet, 2021).

Dans ces réseaux de neurones, les données subissent des transformations en passant d'une couche à l'autre. L'information est comme « distillée » ou « purifiée » en passant dans des filtres qui sont les différentes couches de neurones. La nature des transformations dépend des poids et biais que possèdent chacune des couches. La figure 2 représente schématiquement un exemple de réseau de neurones très simple. Les ronds bleus sont des neurones, les ronds rouges sont des biais et les liens sont des poids. Une fonction de combinaison permet de faire le calcul entre les entrées, les poids et les biais et une fonction d'activation est appliquée en bout de réseau de neurones afin d'en déterminer les sorties (Chollet, 2021).

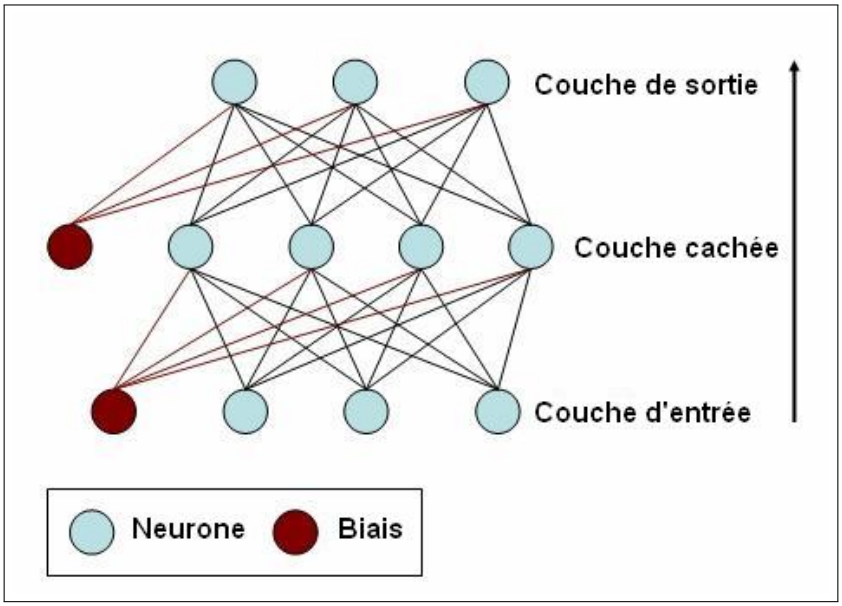


Figure 2 – Représentation schématique d'un réseau de neurones simple.

Les poids et les biais sont des valeurs numériques qui vont être ajustées au cours de l'entraînement d'un modèle. Le but est de trouver la valeur de l'ensemble des poids et des biais du modèle qui permet de faire correspondre les *inputs* avec leur solution et ainsi d'automatiser une tâche. Pour ce faire, le modèle va faire une prédiction en fonction de la donnée qu'il reçoit en entrée et va calculer la distance entre sa prédiction et la cible à atteindre. Le calcul de cette distance est appelé fonction de perte (ou *loss function* en anglais). Le modèle utilise ensuite cette distance comme un signal de rétroaction pour ajuster ses poids dans la bonne direction (calcul de gradients) afin de se rapprocher de la bonne solution et minimiser la fonction de perte. Cette dernière peut être représentée par un paysage vallonné dont on cherche le point le plus bas. Cette étape correspond à l'algorithme de rétropropagation (ou *backpropagation algorithm* en anglais), et est géré par l'optimiseur. Cet algorithme est le principal algorithme de l'AP (LeCun *et al.*, 2015; Chollet, 2021).

L'utilisation de l'AP permet d'atteindre de meilleurs résultats lorsqu'il faut résoudre des problèmes complexes tels que la reconnaissance d'images, la reconnaissance vocale, la découverte de structures dans des jeux de données de grande dimension, la détection d'objets dans une image, etc. Cette technologie tire facilement profit d'une augmentation du nombre de données et de la capacité informatique disponible (LeCun *et al.*, 2015). De plus, l'AP nécessite moins d'expertise parce qu'il profite de la largeur des jeux de données

pour être performant. Les valeurs des paramètres d'un réseau de neurones profond sont déterminées automatiquement. Il ne faut pas les définir manuellement comme pour la programmation traditionnelle. L'AP est aussi plus flexible et robuste que les autres approches (apprentissage automatique (ou *machine learning* en anglais), algorithmes programmés,...) qui sont plus spécifiques à un certain domaine (O'Mahony *et al.*, 2020).

Cependant, l'AP exige un très grand nombre de données, une grande capacité de calculs informatiques, ainsi que du temps pour entraîner les modèles. C'est pourquoi d'autres approches pourraient être privilégiées pour des problèmes plus simples. De surcroît, un modèle d'AP pourrait rencontrer des difficultés à traiter des données trop différentes de son jeu de données d'entraînement. Ceci ne sera pas forcément le cas avec d'autres méthodes telles que l'apprentissage automatique ou des ensembles de règles codées manuellement (O'Mahony *et al.*, 2020).

Les modèles d'AP sont également sujets au surapprentissage. Le surapprentissage se produit dès lors qu'un modèle s'adapte de trop à son jeu de données d'entraînement et qu'il n'arrive pas à généraliser l'information. Par conséquent, le modèle atteint de bons résultats sur les données d'entraînement mais pas sur de nouvelles données. Diverses possibilités existent afin d'éviter ce phénomène. Premièrement, plusieurs jeux de données peuvent être créés, un pour entraîner le modèle et un autre qui sert à vérifier que les résultats sur ce jeu de données restent corrects tout au long de l'entraînement du réseau. L'augmentation de données est une autre possibilité. C'est une méthode qui permet d'augmenter artificiellement la taille d'un jeu de données grâce à des transformations. Celles-ci sont appliquées sur les données lors de l'entraînement d'un modèle, à chaque époque, avec une certaine probabilité. Enfin, des couches de dropout peuvent également être utilisées. Celles-ci sont caractérisées par une probabilité qu'un neurone de ces couches soit désactivé lors de l'entraînement du réseau. Si un neurone est désactivé, il n'intervient pas dans le calcul de la prédiction et son poids n'est pas ajusté lors de la rétropropagation du gradient de l'erreur. Cela permet d'éviter qu'un neurone soit trop dominant lors de la prise de décision d'un modèle.

L'AP a déjà été utilisé pour automatiser de nombreuses et diverses tâches : la classification d'images, la reconnaissance vocale, la retranscription de textes écrits à la main, les voitures à conduite autonome, la traduction automatique, la détection d'objets, la compréhension du langage naturel, etc (LeCun *et al.*, 2015; Mu et Zeng, 2019; Chollet, 2021).

1.2 État de l'art

1.2.1 Classification d'images et de vidéos grâce à l'apprentissage profond

Une vidéo est un ensemble d'images reliées par un lien temporel. Elle peut être considérée comme une séquence d'images acquises à un très court intervalle de temps. De ce fait, la classification de vidéos peut se faire via deux approches différentes. Le modèle d'apprentissage profond peut être appliqué soit à l'échelle de l'image, soit à celle de la vidéo. Le traitement de vidéos par AP est plus complexe que le traitement d'images à cause de ce lien temporel qui existe entre les images d'une vidéo (Rehman et Belhaouari, 2021).

La plupart du temps, la classification d'images à l'aide d'un modèle d'AP se fait grâce à des réseaux de neurones convolutifs profonds (CNN) (Pak et Kim, 2017; Wu *et al.*, 2019; Rehman et Belhaouari, 2021).

Les CNN permettent de prendre en compte le lien qu'il y a entre des pixels proches dans une image. En d'autres termes, ils prennent en compte la dimension spatiale d'une image. Ce sont des réseaux neuronaux constitués de deux composantes : un extracteur de caractéristiques (ou *features*) qui comprend des couches de convolutions (*convolutional layers*) et des couches de *pooling* (*pooling layers*), et un classificateur qui correspond à un perceptron multicouche (*multilayer perceptron*) (Goodfellow *et al.*, 2016; Elgendy, 2020). La structure d'un CNN servant à la classification d'images est décrite sur la figure 3.

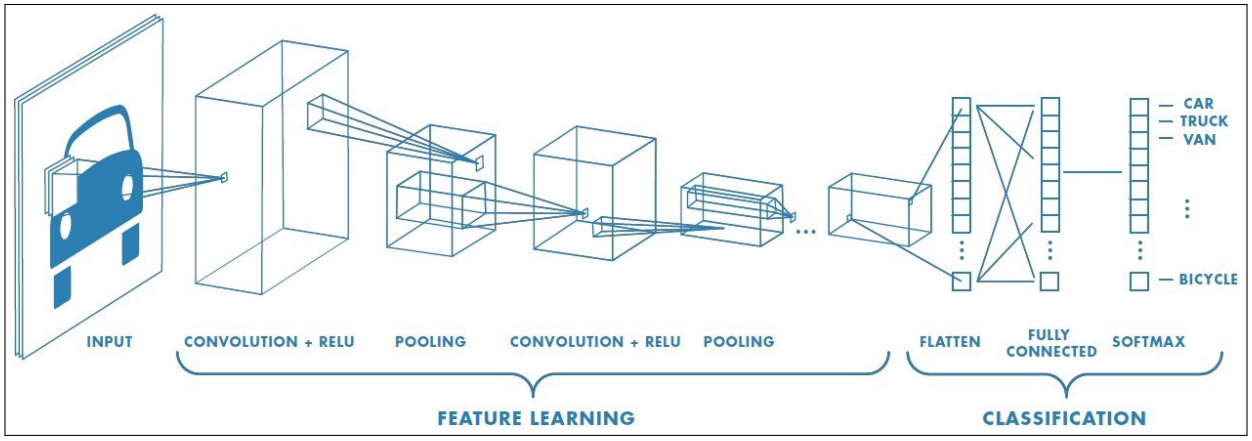


Figure 3 – Schéma d'un réseau de neurones convolutifs servant à la classification d'images.

Dans les couches de convolutions, des filtres opèrent des convolutions. Ces filtres sont appelés des kernels et glissent de pixel en pixel sur l'entièreté de l'image afin de calculer de nouvelles images qui sont appelées cartes de caractéristiques (ou feature maps). Un kernel est une matrice de nombres qui correspondent aux poids de la couche convolutive et qui vont être ajustés au cours de l'entraînement du modèle. Une couche convolutive peut posséder plusieurs kernels. Par conséquent, plusieurs cartes de caractéristiques peuvent être calculées en sortie de cette couche. Ceci provoque une augmentation de la profondeur de l'image (Goodfellow *et al.*, 2016; Elgendy, 2020). La figure 4 décrit une opération de convolution.

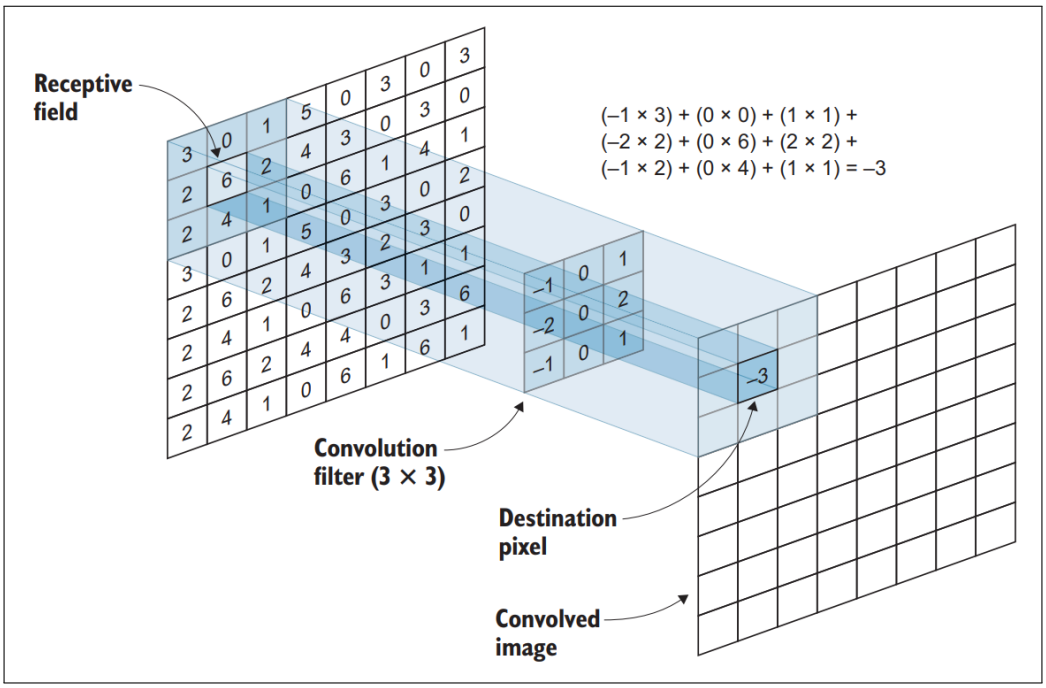


Figure 4 – Exemple de convolution avec un kernel qui a une dimension de 3x3 pixels (Elgendy, 2020).

Les couches de pooling servent à réduire la dimension d'une image. En effet, les couches convolutives vont faire augmenter la profondeur d'une image car plusieurs filtres par couche vont lui être appliqués. Cela a pour conséquences d'augmenter le nombre de paramètres du modèle à entraîner et d'accroître la complexité des calculs. Les couches de pooling vont donc résumer l'information pour ne garder que les informations importantes et ainsi réduire la dimension de l'image. Pour ce faire, un kernel est de nouveau utilisé. Ce kernel n'est plus un ensemble de poids mais est caractérisé par une fonction statistique. Les plus utilisées sont le maximum et la moyenne. Les kernels vont donc prendre le maximum ou la moyenne des valeurs

analysées par ce dernier dans le but de calculer une nouvelle image. Cette opération fait diminuer la taille (hauteur et largeur), mais pas la profondeur des images. En général, les couches de pooling sont placées après chaque couche convolutive (Goodfellow *et al.*, 2016; Elgendy, 2020). La figure 5 présente l'effet d'une couche de pooling maximum sur une image.

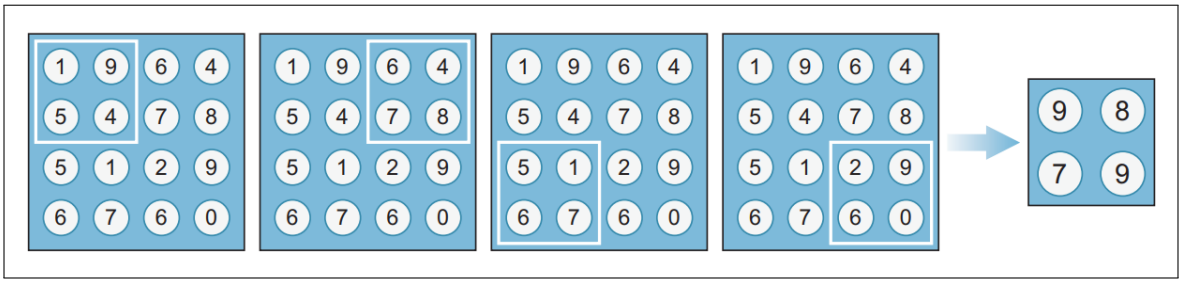


Figure 5 – Exemple de *pooling* maximum sur une image grâce à un kernel qui a une dimension de 2x2 pixels (Elgendy, 2020).

Dans l'extracteur de caractéristiques, les neurones vont apprendre à extraire des caractéristiques qui correspondent à des *patterns* présents dans l'image. De couche en couche, les cartes de caractéristiques passent dans des filtres, deviennent de plus petites tailles, plus nombreuses, plus complexes et abstraites pour la vision de l'homme mais plus révélatrices pour la vision de l'ordinateur. Par exemple, les premières couches vont mettre en évidence des lignes et des bords, les suivantes des formes, des coins ou des cercles et les dernières des pattes, des oreilles ou des museaux (Elgendy, 2020). Ceci est bien représenté à la figure 6.

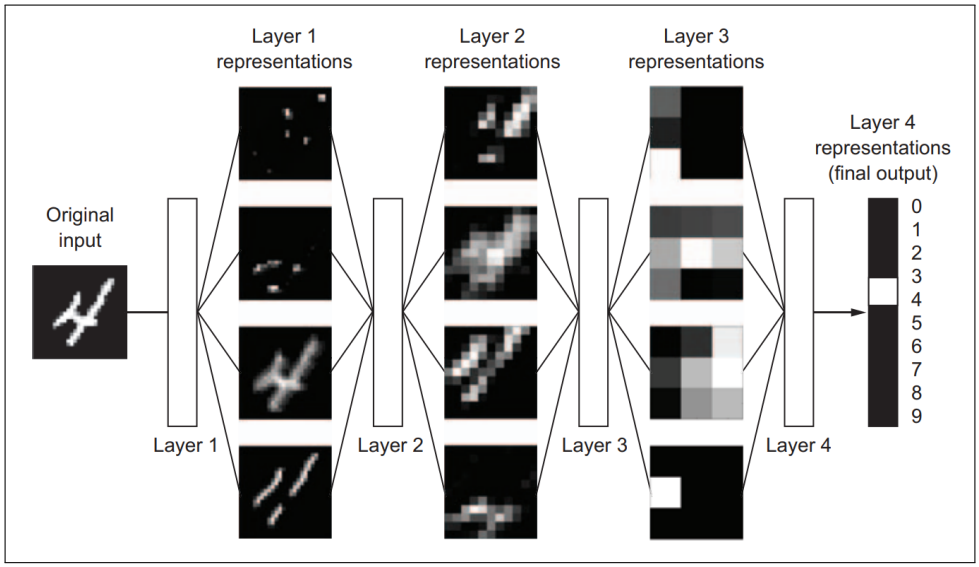


Figure 6 – Exemple de classification d'une image d'un chiffre écrit à la main, par un réseau de neurones convolutifs (Chollet, 2021).

En ce qui concerne le perceptron multicouche, c'est un ensemble de couches de neurones dans lequel chaque neurone d'une couche est connecté à chaque neurone de la couche précédente et de la couche suivante. Certaines de ces couches peuvent être des couches de dropout afin de limiter le surapprentissage (Elgendy, 2020).

Les CNN peuvent aussi comprendre des couches de normalisation de *batch* (ou *batch normalization layers*). Celles-ci sont implémentées après un enchaînement de différentes couches de neurones et de plusieurs fonctions d'activation, afin de normaliser les données. C'est-à-dire, de les remettre à la même échelle. On parle

de normalisation de *batch* car les données ne sont pas normalisées par rapport à l'entièreté du jeu de données mais par rapport au *batch* qui est en train d'être traité par le réseau de neurones. Un *batch* est un ensemble de données qui sont envoyées simultanément au modèle. Les paramètres de ce modèle ne seront ajustés qu'après chaque *batch*. L'erreur qui va être utilisée pour la rétropropagation est la moyenne des erreurs du *batch*. Ce genre de couche permet d'augmenter la vitesse d'apprentissage des modèles (Elgendy, 2020).

À la fin de l'extracteur de caractéristiques, les cartes de caractéristiques sont aplaties dans un vecteur de cartes de caractéristiques afin de passer dans le classificateur (perceptron multicouche). Celui-ci va classifier l'image dans une ou plusieurs classes (Goodfellow *et al.*, 2016; Elgendy, 2020).

LeNet-5 est le premier CNN qui a été entraîné. Il était capable d'identifier des chiffres écrits à la main (LeCun *et al.*, 1998). La vision par ordinateur (ou *computer vision* en anglais) a gagné de l'intérêt grâce à l'organisation de certaines compétitions de vision par ordinateur comme le *ImageNet Large Scale Visual Recognition Competition (ILSVRC)* (ou concours de reconnaissance visuelle à grande échelle ImageNet) qui se base sur le jeu de données « ImageNet » [2] (1 431 167 images pour 1000 classes). Cela a permis une avancée dans le domaine de la classification d'images et le développement d'algorithmes d'AP tels que certaines architectures de réseaux de neurones convolutifs profonds : AlexNet (Krizhevsky *et al.*, 2012) (erreur top 5 de 16,4 % sur le jeu de données « ImageNet ») , VGG Net (Simonyan et Zisserman, 2014) (7,3 %), GoogLeNet (Szegedy *et al.*, 2015) (6,7 %), ResNet-152 (He *et al.*, 2016) (3,6 %), Inception-v2 (Szegedy *et al.*, 2016) (4,8 %) et Inception-v4 (Szegedy *et al.*, 2017) (3,1 %). Le DensNet (Huang *et al.*, 2017) est également une architecture de CNN assez populaire, assez récente et plus profonde. La tendance actuelle est d'augmenter la profondeur des réseaux afin d'atteindre de meilleurs résultats. Cependant, des réseaux de neurones plus profonds requièrent des jeux de données plus fournis pour leur entraînement, des ressources informatiques plus importantes et plus de temps lors de l'entraînement ou de l'utilisation de ceux-ci (Rehman et Belhaouari, 2021). La figure 7 reprend le classement, en terme de profondeur de réseau, des architectures de CNN citées ci-dessus.

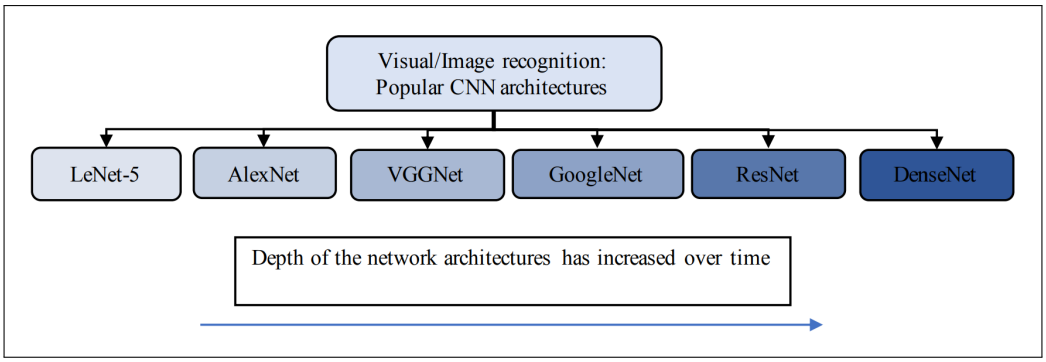


Figure 7 – Classement des architectures de réseaux de neurones convolutifs de la moins profonde à la plus profonde (Rehman et Belhaouari, 2021).

Pour la classification de vidéos, les architectures les plus utilisées sont principalement composées de CNN et de réseaux de neurones récurrents (ou recurent neural networks) (RNN). Les CNN ont déjà été définis largement plus haut. Ces CNN n'ont pas de mémoire. Chaque donnée est traitée indépendamment des autres (Rehman et Belhaouari, 2021). La prédiction faite sur une donnée n'est pas influencée par les prédictions antérieures. Cet aspect est pris en compte dans les RNN (Elman, 1990). Ce genre de réseau de neurones sert à analyser des données séquentielles. Il possède un état interne (mémoire) qui dépend de la prédiction précédente. Grâce à une boucle, cet état interne est pris en compte et influence la prochaine prédiction. Les RNN possèdent donc une mémoire à court terme (Chollet, 2021). La figure 8 schématise un RNN.

2. `https://www.image-net.org/index.php` (consulté le 21/02/2022)

Néanmoins, ce genre de réseau est fortement exposé au *vanishing gradient problem* (ou problème de la disparition du gradient en français) (Yin *et al.*, 2017). Le problème de la disparition du gradient apparaît lorsque, lors de la rétropropagation, les valeurs de gradients calculées diminuent fortement. Ceci amène alors l'annulation des gradients. La valeur des poids n'évolue donc plus et le modèle ne s'entraîne plus (Hochreiter, 1998). Pour pallier à cela, plusieurs évolutions du RNN ont été développées.

Le Long short-term memory (LSTM) (Hochreiter et Schmidhuber, 1997) est un réseau de neurones qui comprend une cellule qui est capable de mémoriser de l'information à plus long terme. Il est composé de trois portes qui gèrent le flux d'informations au cours du temps. La porte d'entrée (*input gate*) définit ce qui va être stocké de la donnée qui est en cours d'analyse, dans l'état de la cellule. La porte d'oubli (*forget gate*) va définir ce qui peut être oublié par la cellule. Et, la porte de sortie (*output gate*) va définir quelles informations de la cellule vont servir à faire la prédiction en étant combinées à la donnée analysée. Cette cellule est réinitialisée après le traitement de chaque séquence.

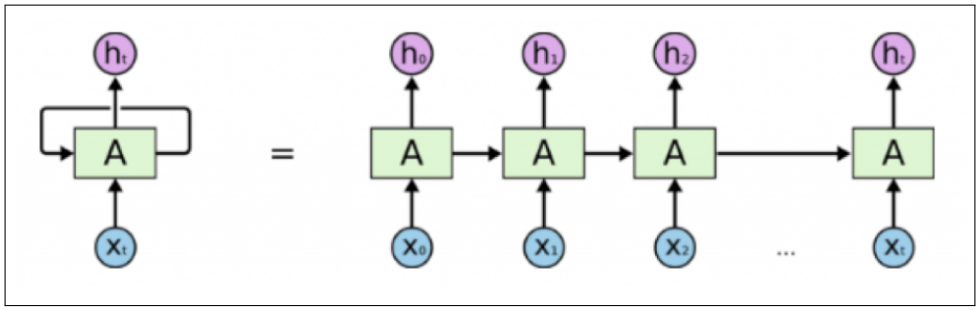


Figure 8 – Schéma d'un réseau de neurones récurrents classique.

Le Gated Recurrent Unit (GRU) (Cho *et al.*, 2014) est une variante du LSTM avec moins de poids à entraîner car il ne possède que deux portes dans sa structure. Ce sont les portes de mise à jour (*update gate*) et de réinitialisation (*reset gate*) qui gèrent le flux d'informations d'une séquence.

Une autre alternative au RNN est le convolutional long short-term memory (ConvLSTM) (Shi *et al.*, 2015). C'est le même principe qu'un LSTM mais avec des structures convolutives comme entrée, sortie et état de la cellule. Ces modèles apprennent quelles informations doivent être gardées ou non au cours du temps dans une séquence d'images.

Il existe différentes catégories d'architectures de modèles d'AP qui servent à la classification de vidéos. D'abord, des CNN à deux dimensions (comme ceux présentés précédemment) peuvent être utilisés afin d'extraire les caractéristiques de chaque image d'une vidéo avant de classer la vidéo en fonction de l'ensemble de ces caractéristiques (Zha *et al.*, 2015).

Des CNN à trois dimensions peuvent également être employés (Tran *et al.*, 2015; Carreira et Zisserman, 2017; Diba *et al.*, 2017). Ceux-ci prennent en compte la dimension spatiale et la dimension temporelle des vidéos grâce à des opérations de convolutions et de *pooling* en trois dimensions.

Il est aussi possible de créer des réseaux de neurones à deux ou plusieurs flux (Simonyan et Zisserman, 2014; Feichtenhofer *et al.*, 2016; Carreira et Zisserman, 2017). Par exemple, un flux d'informations temporelles peut être géré par un type de RNN et un flux d'informations spatiales par un CNN. Feichtenhofer *et al.* (2019) ont créé une architecture d'AP à deux flux, le SlowFast : une voie lente (un petit nombre d'images sont extraites des vidéos) et une voie rapide (un plus grand nombre d'images sont extraites des vidéos). Les deux voies sont des ResNets à 3 dimensions. Ces deux voies sont reliées par des connexions latérales.

Une autre catégorie est constituée par les réseaux de neurones convolutifs mixtes dans lesquels sont combinés des CNN à deux dimensions et des CNN à trois dimensions (Tran *et al.*, 2018).

Pour terminer, la catégorie hybride rassemble les architectures composées d'une combinaison de CNN et de RNN (Wu *et al.*, 2015 ; Carreira et Zisserman, 2017 ; Sudhakaran et Lanz, 2017a ; Sudhakaran et Lanz, 2017b).

1.2.2 Traitement des données de pièges photographiques grâce à l'apprentissage profond

Le traitement de données de pièges photographiques par apprentissage profond est un sujet qui a déjà été fortement étudié par le monde scientifique (Beery *et al.*, 2019). Cependant, il existe encore des difficultés récurrentes. Premièrement, les performances des modèles d'AP diminuent (parfois de manière critique) quand ils sont appliqués sur des données qui proviennent d'une autre zone géographique ou d'un autre environnement que celles qui ont permis de développer le modèle. Cela peut être dû au fait que la végétation et le paysage varient d'un endroit à l'autre du globe mais c'est surtout dû au fait que certaines espèces animales peuvent être présentes dans une région mais absentes dans une autre. Les modèles ne sont pas entraînés sur toutes les espèces animales qui existent au monde (Beery *et al.*, 2019; Schneider *et al.*, 2020). Deuxièmement, il peut être difficile pour un modèle d'AP de faire des prédictions correctes sur des photos ou des vidéos qui ont été capturées avec des caméras différentes et dont l'arrière-plan est différent de ceux des données qui ont servi à la construction de ce modèle (Beery *et al.*, 2019; Schneider *et al.*, 2020). Ensuite, les jeux de données de PP sont déséquilibrés du point de vue de la représentation des différentes espèces animales. Effectivement, les espèces animales les plus rares sont moins présentes que les espèces animales communes. Cela peut avoir des conséquences sur les performances du modèle et notamment pour les classes sous-représentées (LeCun *et al.*, 2015; Schneider *et al.*, 2020). Enfin, la taille du jeu de données peut également avoir un impact important sur la qualité d'un modèle d'AP. Certains projets nécessitant des PP n'ont pas la capacité et l'intérêt de récolter d'énormes jeux de données (Li *et al.*, 2017; Schneider *et al.*, 2020).

La reconnaissance d'espèces animales dans des données de PP peut se faire soit par détection d'objets, soit par classification d'images ou de vidéos. La détection d'objets peut être réalisée en deux étapes : il y a d'abord une recherche des propositions de régions et ensuite ces régions sont classifiées. Dans certain cas, ces deux étapes sont effectuées grâce à l'AP (Beery *et al.*, 2019; Carl *et al.*, 2020; Beery *et al.*, 2020; Norouzzadeh *et al.*, 2021; Schindler et Steinhage, 2021; Vecvanags *et al.*, 2022). Dans d'autres cas, seule la seconde étape de classification est effectuée par un modèle d'AP (Chen *et al.*, 2014; Zhang *et al.*, 2016; Yousif *et al.*, 2017; Weinstein, 2018; Schindler et Steinhage, 2021). Ces deux étapes peuvent aussi être réalisées en même temps via l'utilisation de l'AP (Loos *et al.*, 2018; Vecvanags *et al.*, 2022).

En ce qui concerne la classification d'images, les jeux de données qui sont utilisés pour entraîner les modèles d'AP peuvent être très différents en terme de taille. Certains sont composés de plus d'un million d'images (Gomez Villa *et al.*, 2017; Norouzzadeh *et al.*, 2018; Tabak *et al.*, 2019; Willi *et al.*, 2019), d'autres entre 1 million et 100 000 images (Willi *et al.*, 2019; Kutugata *et al.*, 2021; Whytock *et al.*, 2021; Yang *et al.*, 2021), encore d'autres entre 100 000 et 10 000 (Nguyen *et al.*, 2017; Beery *et al.*, 2018; Willi *et al.*, 2019; Schneider *et al.*, 2020; Zualkernan *et al.*, 2020) et d'autres encore moins (Gomez Villa *et al.*, 2016; Chen *et al.*, 2019). Cependant, Willi *et al.* (2019) ont montré que des jeux de données plus importants permettent d'entraîner des modèles plus performants.

Les jeux de données de PP sont la plupart du temps déséquilibrés. Ceci est dû aux effectifs naturellement déséquilibrés des espèces animales dans les écosystèmes. Un modèle a tendance à mieux classifier les espèces les plus représentées car il a une plus grande hétérogénéité au sein de celles-ci et plus d'échantillons de ces classes pour développer des filtres performants (Goodfellow *et al.*, 2016; Tabak *et al.*, 2019). Il est tout de même possible d'équilibrer un jeu de données soit en sélectionnant dans chaque classe, aléatoirement, un nombre de données égal à l'effectif de la classe minoritaire (Gomez Villa *et al.*, 2017; Yang *et al.*, 2021), soit en enrichissant les classes minoritaires par duplication des données de celles-ci (Kutugata *et al.*, 2021).

Cette dernière méthode permet d'augmenter les effectifs des espèces minoritaires. Néanmoins, cela provoque un manque de robustesse du modèle, par rapport à ces classes, qui a du mal à généraliser l'information et prédire correctement ces espèces sur un nouveau jeu de données (Kutugata *et al.*, 2021). Gomez Villa *et al.* (2017) ont obtenu de meilleures performances avec un jeu de données plus petit mais équilibré (une amélioration de 5 à 15% d'exactitude globale) mais Nguyen *et al.* (2017) ont tiré des conclusions contraires à cela en utilisant un jeu de données différent.

Une autre technique utilisée en AP est le transfert d'apprentissage (ou *transfer learning* en anglais) ou le fait de préentraîner un modèle. C'est une méthode qui permet de raccourcir le temps d'entraînement et d'atteindre de meilleurs résultats. Un modèle est dit préentraîné quand ses poids ne sont pas initialisés aléatoirement mais qu'ils sont copiés d'un modèle déjà entraîné auparavant (Elgendy, 2020). Cette méthode est couramment employée en classification de données de PP (Nguyen *et al.*, 2017; Beery *et al.*, 2018; Tabak *et al.*, 2019; Schneider *et al.*, 2020; Zualkernan *et al.*, 2020; Kutugata *et al.*, 2021; Schindler et Steinhage, 2021; Whytock *et al.*, 2021).

Dans la littérature, la classification d'images de PP a été réalisée grâce à différentes architectures de CNN. La table 1 reprend les architectures qui ont été utilisées pour ce but. Il est également possible de combiner plusieurs modèles. Dans ce cas, la prédiction finale correspond à la moyenne des prédictions des ceux-ci. Les performances de cette combinaison sont généralement légèrement supérieures à un CNN unique. Cependant le temps de traitement des données se voit impacté négativement (Norouzzadeh *et al.*, 2018; Schneider *et al.*, 2020).

Norouzzadeh *et al.* (2018) ont comparé plusieurs architectures (AlexNet, NiN, VGG Net, GoogLeNet, ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152). Ce sont le VGG Net pour de la classification binaire (CB) (animal / arrière-plan) et le ResNet-152 pour la classification multi-espèces (CM) qui ont permis d'avoir les meilleurs résultats. Gomez Villa *et al.* (2017) ont également testé plusieurs modèles (AlexNet, VGG Net, GoogLenet, ResNet50, ResNet101 et ResNet152). Le ResNet101 s'est montré le plus avantageux. De leur côté, Nguyen *et al.* (2017) ont comparé le AlexNet, le ResNet-50 et VGG Net. Le ResNet-50 a été le plus efficace pour la CB alors que c'est le VGG Net qui a été choisi pour la CM. Le DenseNet201 a été privilégié par Schneider *et al.* (2020) parmi le Inception-ResNet-V3, le InceptionV3, le NASNetMobile, le MobileNetV2, et le Xception. Gomez Villa *et al.* (2016) ont eux testé les architectures AlexNet, VGG Net, GoogLenet, ResNet-50 ResNet-101 et ResNet-152. Les trois architectures ResNet ont atteint des résultats significativement supérieurs.

Plusieurs types de classifications ont déjà été envisagés dans la littérature : une classification binaire, une classification multi-espèces (Gomez Villa *et al.*, 2017; Nguyen *et al.*, 2017; Beery *et al.*, 2018; Tabak *et al.*, 2019; Schneider *et al.*, 2020; Zualkernan *et al.*, 2020; Kutugata *et al.*, 2021; Whytock *et al.*, 2021) ou une classification en deux étapes. La CB peut servir à détecter soit la présence ou non d'un animal (Nguyen *et al.*, 2017), soit la présence ou non d'une espèce spécifique (Chen *et al.*, 2019; Kutugata *et al.*, 2021). La CM se fait avec plus de deux classes. Pour la classification en deux étapes, un modèle va d'abord être utilisé pour classifier une première fois les données et une partie de ces données seront reclassées de manière plus précises grâce à un second modèle. Par exemple, Norouzzadeh *et al.* (2018) et Willi *et al.* (2019) ont utilisé un modèle afin de séparer les images vides des images avec des animaux et ont ensuite réalisé une CM sur les images avec des animaux. De même, Gomez Villa *et al.* (2016) ont réalisé une première CB : oiseau ou pas d'oiseau et puis ont classé les images correspondant à « pas d'oiseau » en deux groupes de mammifères.

Pour rappel, l'augmentation de données est une méthode qui permet d'augmenter artificiellement la taille d'un jeu de données grâce à des transformations. Cette technique peut avoir un impact positif sur la qualité d'un modèle. En classification de données de PP, les transformations souvent utilisées sont : le flip horizontal, les distorsions, le flou et un changement d'intensité, de contraste ou de luminosité (Nguyen *et al.*, 2017; Beery *et al.*, 2018; Tabak *et al.*, 2019; Schneider *et al.*, 2020; Kutugata *et al.*, 2021; Schindler et Steinhage,

2021; Whytock *et al.*, 2021).

Table 1 – Architectures de réseaux de neurones convolutifs utilisées pour classifier des images de pièges photographiques, dans la littérature.

Architecture	**Références**
AlexNet	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Nguyen *et al.*, 2017 ; Norouzzadeh *et al.*, 2018 ; Yang *et al.*, 2021
NiN	Norouzzadeh *et al.*, 2018
VGG Net	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Nguyen *et al.*, 2017 ; Norouzzadeh *et al.*, 2018
GoogLeNet	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Norouzzadeh *et al.*, 2018
ResNet-18	Norouzzadeh *et al.*, 2018 ; Tabak *et al.*, 2019 ; Willi *et al.*, 2019 ; Zualkernan *et al.*, 2020 ; Yang *et al.*, 2021
ResNet-34	Norouzzadeh *et al.*, 2018
ResNet-50	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Nguyen *et al.*, 2017 ; Norouzzadeh *et al.*, 2018 ; Whytock *et al.*, 2021
ResNet-101	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Norouzzadeh *et al.*, 2018
ResNet-152	Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Norouzzadeh *et al.*, 2018
Inception-V3	Beery *et al.*, 2018 ; Schneider *et al.*, 2020 ; Zualkernan *et al.*, 2020 ; Yang *et al.*, 2021
Xception	Schneider *et al.*, 2020
Inception-ResNet-V3	Schneider *et al.*, 2020
MobileNet-V2	Schneider *et al.*, 2020 ; Zualkernan *et al.*, 2020
NASNetMobile	Schneider *et al.*, 2020
DenseNet121	Zualkernan *et al.*, 2020
DenseNet201	Schneider *et al.*, 2020

Un seuil de confiance peut être défini afin de trouver un compromis entre l'utilisation de l'AP et la classification par l'homme. Les prédictions qui possèdent une probabilité d'appartenir à une classe supérieure à ce seuil seront acceptées. Les autres seront vérifiées par l'homme. Cela permet d'augmenter l'exactitude globale du modèle et de réduire la charge de travail du scientifique malgré que tout le traitement des données ne soit pas entièrement automatisé (Gomez Villa *et al.*, 2016; Willi *et al.*, 2019; Whytock *et al.*, 2021; Yang *et al.*, 2021).

En classification de vidéos ou de séquences d'images de PP, Schindler et Steinhage (2021) ont fait de la reconnaissance d'actions sur des vidéos grâce à quatre architectures : un ResNet à trois dimensions, un mélange entre des ResNets à trois dimensions et des ResNets à deux dimensions, un ResNet à deux dimensions plus une dimension temporelle et le SlowFast. Les trois premières architectures ont donné de bons résultats (92,8 % à 94,1 % d'exactitude globale) alors que l'architecture SlowFast a moins bien performé (66,3 % d'exactitude globale). Le jeu de données était constitué de 356 vidéos pour trois classes. Chen *et al.* (2019) ont utilisé deux CNN afin de créer des classificateurs d'images. Ils ont ensuite adapté ces classificateurs pour des séquences d'images. Pour ce faire, ils ont calculé un indice qui dépend de la différence de valeur de pixels entre deux images et du nombre de pixels différents. Les images qui possèdent une valeur de cet indice supérieure à un seuil défini, ont été considérées comme des images avec des animaux. Ces images ont ensuite été traitées par leurs deux modèles et les prédictions n'ont été acceptées qui si elles étaient identiques pour des images proches dans le temps. Pour finir, Yang *et al.* (2021) ont aussi fait de la CB (animal / arrière-plan)

sur des images et des séquences d'images. Ils ont entraîné six modèles grâce à trois architectures : ResNet-18, InceptionV3 et AlexNet, et deux jeux de données : un équilibré et l'autre déséquilibré. Ils ont également testé différentes combinaisons de ces six modèles. Individuellement, le meilleur modèle s'est avéré être le Inception-V3 mais les meilleures performances ont été atteintes grâce à la combinaison des six modèles. Une séquence d'images a été considérée comme « vide » uniquement lorsque toutes ses images ont été prédites comme telles.

1.3 Objectifs

Comme expliqué précédemment, le monde actuel est frappé par une crise de la biodiversité. Les forêts tropicales nécessitent une attention particulière. Selon Wilson (1988), au moins deux tiers des organismes de la planète vivent dans cet habitat qui est soumis à de fortes dégradations à cause de la pression anthropique. Les forêts tropicales d'Afrique centrale n'échappent pas à cette vulnérabilité (de Wasseige *et al.*, 2012). Les PP représentent donc un outil intéressant pour surveiller la biodiversité de cet écosystème. Néanmoins, ces derniers produisent une quantité très importante de données. L'analyse de ces données est très chronophage et énergivore pour l'homme. Cependant, un modèle d'AP pourrait aider à surmonter cette problématique.

Dans le cadre de ce travail, l'information qui nous intéresse en premier lieu est de savoir quelles espèces animales sont présentes dans une vidéo capturée par un PP qui a été placé en forêt tropicale, en Afrique centrale. L'objectif principal est donc de développer un modèle d'AP afin de classifier les vidéos prises par ces appareils. Le choix de faire de la classification permet de gagner du temps à l'annotation des données tout en extrayant l'information utile. En effet, l'annotation pour de la classification se résume à donner une étiquette à chaque image ou vidéo. Tandis que pour de la détection d'objets, il faut marquer chaque animal au moyen d'une boîte, d'un point ou d'un ensemble de points, sur chaque donnée.

Le département de forêt tropicale de l'université de Gembloux Agro-Bio Tech (DFT de GxABT) et le centre de coopération internationale en recherche agronomique pour le développement (Cirad) ont fourni les vidéos qui ont permis la réalisation de ce travail. Ces données ont été prises dans des forêts tropicales d'Afrique Centrale. Le domaine d'utilisation du modèle créé se limitera donc aux espèces animales et aux paysages de cette zone géographique. Le but est que ce modèle puisse être utilisé à l'avenir par des utilisateurs de PP qui font des études dans cette région. Cela pourrait notamment être utile pour ces deux institutions dans leurs futures recherches.

L'objectif secondaire de ce travail est de créer un modèle d'AP qui est capable de détecter si un animal est présent ou non dans une vidéo prise par un PP qui a également été installé en forêt tropicale, en Afrique centrale. En effet, une partie des vidéos capturées par ces caméras sont vides, il n'y a pas d'animal dedans. Ce sont des fausses détections. Cela arrive quand la caméra détecte un mouvement qui n'est pas celui d'un animal (par exemple, une branche d'un arbre qui bouge à cause du vent). Cette tâche pourrait s'avérer plus simple que la première et faciliterait à moindre mesure le travail des utilisateurs de PP. Ces personnes n'auraient plus qu'à analyser les vidéos avec des animaux.

Un objectif tertiaire a également été défini. Celui-ci est de comparer les modèles créés avec des outils qui existent déjà : Wildlife Insights, Mbaza AI et Zamba Cloud. Ceci permettra d'évaluer l'apport réel des modèles développés et de définir leur domaine d'utilisation.

2 Matériels et méthodes

2.1 Outils informatiques

Les différents modèles d'apprentissage profond ont été développés grâce au langage de programmation Python 3.9.12. La distribution logicielle Anaconda 2.2.0 (64 bits) et l'environnement de développement Spyder 5.1.5 ont été utilisés. Différentes bibliothèques ont été exploitées : albumentations 1.2.0, decord 0.6.0, numpy 1.21.5, opencv 4.6.0.66, pandas 1.4.2, pillow 9.0.1, pytorchvideo 0.1.5, sklearn 1.0.2, torch 1.12.0, torchvision 0.13.0, wandb 0.12.21.

L'ordinateur qui a servi à entraîner les différents modèles, possède un processeur Intel(R) Core(TM) i9-9900X CPU 3,50GHz et une mémoire vive de 64,0 GB. Il est équipé d'une carte graphique NVIDIA GeForce RTX 2080 dont la mémoire est de 8,0 Go.

Le site internet « Weights & Biases » [3] est un outil qui a servi à suivre l'évolution de l'erreur sur le jeu de données d'entraînement et de l'erreur, de l'exactitude globale et de l'exactitude moyenne sur le jeu de données de validation, au cours des entraînements. Cet outil a également permis de sauver toutes les caractéristiques, les hyperparamètres et les résultats des modèles entraînés.

2.2 Données brutes

Pour la suite de ce travail, la classification multi-espèces (CM) fera référence au premier objectif et la classification binaire (CB) se rapportera au second.

Deux bases de données brutes ont été constituées. La première est composée de 43 944 vidéos et servira à développer les différents modèles d'AP. 30 517 vidéos de cette base de données proviennent du DFT de GxABT et ont été prises entre septembre 2018 et avril 2021. Le reste des vidéos (13 427) a été collecté par le Cirad entre janvier 2016 et juin 2019. La deuxième base de données comprend 21 965 vidéos acquises par le Cirad entre mai 2021 et octobre 2021. Elle a servi à évaluer la robustesse des modèles sur un jeu de données indépendant de ceux qui ont servi à les concevoir. Ce jeu de données peut être assimilé à une situation de données nouvellement collectées sur le terrain qui nécessite une identification complète. Celui-ci servira également à comparer les différents modèles développés avec des outils existants. Toutes les vidéos constituant ces 2 bases de données ont été acquises en forêt tropicale, en Afrique centrale (Cameroun et Gabon). Pour illustrer cela, la figure 9 présente un exemple de 5 images extraites d'une vidéo de 5 secondes (1 image par seconde) qui constitue le jeu de données brutes numéro 1.

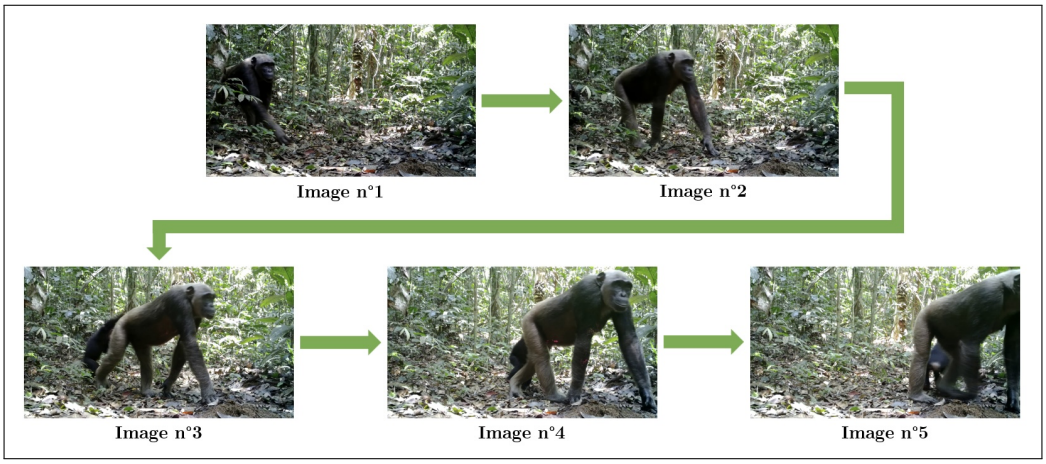


Figure 9 – Exemple de 5 images (1 par seconde) extraites d'une vidéo de 5 secondes capturée par un piège photographique en forêt tropicale, en Afrique centrale.

3. `https://wandb.ai/site` (consulté le 05/05/2022)

Les vidéos ont été capturées grâce au modèle de pièges photographiques Bolyguard SG 2060X (Boly, Victoriaville, QC, Canada). Ce sont soit des vidéos RGB (*Red, Green, Blue* ou Rouge, vert, bleu) prises quand la luminosité était assez forte, soit des vidéos infrarouges prises en situation de plus faible luminosité ou la nuit. Elles sont de durées et de formats différents. Les tables 2 et 3 reprennent les principales caractéristiques des vidéos des 2 bases de données.

Table 2 – Caractéristiques des vidéos de la base de données numéro 1.

Format	**Durée (seconde)**	**Largeur x hauteur (pixel)**	**Images par seconde**	**Nombre de vidéos**	**Pourcentage**
MOV	5	1920 x 1080	30	21262	48,3
MP4	30	1920 x 1080	30	9352	21,2
MP4	5	1920 x 1080	30	13415	30,5

Le DFT de GxABT et le Cirad ont également fourni des tableurs Excel comprenant des informations à propos des données. Ces fichiers renseignent notamment pour chaque vidéo : les espèces présentes dans celle-ci, l'heure et la date à laquelle elle a été prise et l'identifiant de la caméra qui l'a capturée. Les caractéristiques des PP sont aussi disponibles dans ces tableurs : le pays, le site et les dates de déploiement, le modèle, ainsi que sa position géographique (latitude et longitude).

Table 3 – Caractéristiques des vidéos de la base de données numéro 2.

Format	**Durée (seconde)**	**Largeur x hauteur (pixel)**	**Images par seconde**	**Nombre de vidéos**
MOV	5	1920 x 1080	30	21965

2.3 Définition des classes

De toutes les espèces animales qui avaient été renseignées par le DFT de GxABT et le Cirad, 22 classes ont été définies en collaboration avec un représentant de ces 2 institutions. Ces classes correspondent à 20 espèces ou groupes d'espèces, une classe « Autre » et une classe « No_sp » qui reprend toutes les vidéos dans lesquelles il n'y a aucun animal. Les espèces pour lesquelles le nombre de vidéos était trop faible pour en faire une classe à part entière et qui ne pouvaient pas être regroupées avec d'autres espèces ont été englobées dans la classe « Autre ». Cette dernière pourrait aider le modèle à performer dans des zones géographiques légèrement différentes. En effet, des espèces qui seraient spécifiques à une zone d'étude et qui n'auraient encore jamais été vues par le modèle, devraient, par hypothèse, être attribuées à cette classe.

Le tableau 4 reprend les classes créées et les espèces ou groupes d'espèces qu'elles comprennent. Ce tableau renseigne aussi la représentation de chaque classe dans les 2 bases de données. Les groupes d'espèces marqués d'un astérix signifient que les vidéos leur correspondant ont été reclassées parce que ceux-ci reprenaient des individus de plusieurs classes. C'est pour cette raison qu'ils se retrouvent à plusieurs endroits dans la table 4. Certaines espèces ont été regroupées. Cette décision peut avoir été prise pour plusieurs raisons. Soit parce que ces espèces n'apparaissent que dans trop peu de vidéos, soit car elles sont trop ressemblantes, soit parce qu'il y a eu des confusions lors de l'identification des données ou soit car elles n'avaient pas de réels intérêts à être séparées. Les classes minoritaires sont les grands félins, les buffles de forêt et les « Autre ». Ces classes pourraient poser des problèmes aux modèles à cause de ce manque d'effectifs.

Table 4 – Classes et effectifs de classes.

Espèces ou groupes d'espèces définis par le DFT de GxABT et le Cirad	**Classes**	**Jeu de données n°1**	**Jeu de données n°2**
Athérure	Athérure	1742	776
Antilope de Bates, Autre rongeur, Céphalophe à front noir, Daman, Galago, Grand aulacode, Humain, Hylochère, Oryctérope, Perodicticus edwardsi, Poiane, Ratel, Reptile, Sitatunga, Tragelaphus spp.*, Varan	Autre	155	77
Bongo, Tragelaphus spp.*	Bongo	266	/
Buffle de forêt	Buffle de forêt	79	/
Céphalophe à dos jaune, Céphalophe spp.*	Céphalophe à dos jaune	965	463
Céphalophe bleu, Céphalophe spp.*	Céphalophe bleu	10946	2800
Céphalophe spp.*, Céphalophe à bande dorsale noire, Céphalophe à pattes blanches, Céphalophe à ventre blanc, Céphalophe de Peters	Céphalophe rouge	10997	5751
Cercocèbe agile, Cercopithecus spp., Cercopithèque nez-blanc, Cercopithèque pogonias, Colobe guéréza, Hocheur, Mangabey à joues blanches, Miopithèque, Moustac, Petit singe	Cercopithecidae	1744	18
Chevrotain aquatique	Chevrotain aquatique	319	107
Chimpanzé, Grand singe*	Chimpanzé	517	209
Civette, Genette, Nandinie	Civette/Genette/Nandinie	261	94
Écureuil, Écureuil géant de Stanger, Écureuil spp., Funisciure à pattes rousses, Funisciure isabelle	Écureuil	1784	1881
Éléphant de forêt	Eléphant de forêt	1874	256
Gorille, Grand singe*	Gorille	325	44
Chat doré, Léopard, Panthère	Grand félin	68	45
Mandrill	Mandrill	782	272
Autre mangouste, Cusimanse, Mangouste à pattes noires, Mangouste à tête plate	Mangouste	555	251
No_sp	No_sp	7231	6664
Calao à huppe blanche, Francolin de Latham, oiseau, Pintade huppée, Pintade noire, Râle à pied rouge, Tourtelette demoiselle	Oiseau	1039	962
Pangolin géant, Petit pangolin	Pangolin	149	77
Potamochère	Potamochère	1248	286
Rat géant, Souris	Rat géant/Souris	818	932

* Groupe d'espèces qui a été reclassifié dans plusieurs classes.

2.4 Création de différents jeux de données

De la base de données numéro 1 ont été créés trois jeux de données : un jeu de données d'entraînement, un jeu de données de validation et un jeu de données de test. Ceux-ci ont servi à développer les différents modèles d'apprentissage profond. Le jeu de données d'entraînement a servi à entraîner les modèles. Sur le jeu de données de validation était calculé l'exactitude globale après chaque époque. Les paramètres des modèles ont été sauvegardés uniquement si l'exactitude globale sur le jeu de validation était supérieure au maximum des exactitudes globales calculées aux époques antérieures. Le jeu de test a été utilisé afin de calculer les performances des modèles. Ce dernier permet de juger les performances des modèles sur un jeu de données qui n'a pas influencé leur développement. En effet, le jeu de d'entraînement a servi pour entraîner les modèles et le jeu de validation a servi pour sauvegarder les paramètres des modèles qui maximisent leurs performances sur ce même jeu de données.

Ces 3 jeux de données ont été rendu aussi indépendants que possible, pour éviter le surapprentissage (ou *overfitting*) du modèle. En effet, cela permet de vérifier que le modèle garde des performances correctes sur des jeux de données qui sont bien différents de celui d'entraînement. Pour assurer cette indépendance, toutes les vidéos d'une même caméra ont été versées dans le même jeu de données comme l'ont fait Whytock *et al.* (2021).

Comme expliqué dans la section 1.2.2, à la page 18, les modèles d'apprentissage profond rencontrent des difficultés à faire de bonne prédiction sur des données dont ils n'ont jamais vu l'arrière-plan. Pour essayer de pallier à ce problème, les vidéos d'une même caméra et donc qui possèdent le même arrière-plan ont été placées dans le même jeu de données. De plus, le nombre de caméras et donc d'arrière-plans utilisés a été maximisé. L'entièreté des PP sont représentés le plus équitablement possible dans chaque classe de chaque jeu de données.

Les 3 jeux de données ont été définis pour la classification multi-espèces et binaire, d'images et de vidéos. Pour passer des vidéos aux images, une image par seconde a été échantillonnée sur chaque vidéo et placée dans le jeu de données équivalant. Ce nombre d'images par seconde a été choisi pour s'assurer de retrouver l'animal de chaque vidéo sur les images. En effet, l'animal n'est pas toujours présent sur toute la vidéo, parfois il n'apparaît qu'un très court instant. Ce nombre d'images par seconde permet également de ne pas sélectionner trop d'images qui sont fort semblables, d'une même vidéo. Il est préférable d'extraire moins d'images par seconde d'une vidéo mais d'échantillonner plus de vidéos. Cela permet de reprendre une plus grande variabilité de la base de données de départ. Les images ont dû être annotées. Cette annotation sera détaillée au point suivant.

Lors de la constitution des jeux de données, les vidéos reprenant plusieurs espèces ont été mises de côté. La raison est qu'elles ne représentent que 424 vidéos dans la base de données numéro 1 (donc moins de 1 % des données) et 0 vidéo dans la deuxième. L'utilisation d'un modèle d'apprentissage profond qui peut prédire plusieurs classes par vidéo, n'est pas pertinente car il serait probablement moins performant à cause de ce manque de variabilité de cas. Il est d'ailleurs très rarement utilisé dans la littérature pour classifier des données de PP.

L'objectif était de rassembler 70% des données sélectionnées dans le jeu d'entraînement, 20% dans le jeu de test et 10% dans le jeu de validation. Ces proportions devaient également être respectées au maximum que possible à l'échelle des classes. Pour chaque classe, un maximum de caméras ont été sélectionnées et chaque caméra a été représentée aussi équitablement que possible. Le nombre de caméras et d'arrière-plans qui sont disponibles par classe dans la base de données numéro 1 est repris dans la table 5. Pour rappel, il faut également que toutes les vidéos d'une même caméra se retrouvent dans le même jeu de données.

Afin de respecter toutes ces conditions, la division des données sélectionnées en 3 jeux de données a été opérée grâce à une méthode empirique. Un très grand nombre de configurations, qui respectent les conditions liées aux caméras, ont été testées et celle qui donnait la plus petite erreur a été retenue. Pour chaque classe de

chaque jeu de données, on calcule la différence entre la proportion de données qui devraient y être présentes et la proportion de données qui y sont réellement. L'erreur est la somme de ces différences et son calcul est décrit par l'équation suivante :

$$Erreur = \sum_{i=classes} ((|0,7 - P_{e,i}|) + (|0,2 - P_{t,i}|) + (|0,1 - P_{v,i}|)) \quad (1)$$

avec $P_{e,i}$ qui est la proportion de données de la classe i dans le jeu d'entraînement, $P_{t,i}$ qui est la proportion de données de la classe i dans le jeu de test et $P_{v,i}$ qui est la proportion de données de la classe i dans le jeu de validation.

Table 5 – Nombre de caméras par classe dans la base de données brutes numéro 1.

Classe	**Nombre de caméras**
Athérure	213
Autre	72
Bongo	26
Buffle de forêt	9
Céphalophe à dos jaune	185
Céphalophe bleu	270
Céphalophe rouge	300
Cercopithecidae	68
Chevrotain aquatique	56
Chimpanzé	111
Civette/Genette/Nandinie	118
Écureuil	192
Eléphant de forêt	100
Gorille	78
Grand félin	48
Mandrill	32
Mangouste	158
No_sp	294
Oiseau	179
Pangolin	88
Potamochère	132
Rat géant/Souris	152

Les tables 6 et 8 reprennent la composition des jeux de données qui ont servi à développer les différents modèles pour la CM, pour respectivement les images et les vidéos. Les tables 7 et 9 reprennent quant à elles la composition de ces mêmes jeux de données mais pour la CB.

Un deuxième jeu de données de test a été constitué grâce à la base de données numéro 2. Pour ce faire, 100 vidéos de chaque classe ont été sélectionnées tout en maximisant le nombre de caméras et donc d'arrière-plans. Si moins de 100 vidéos étaient disponibles pour une classe, toutes les vidéos de celle-ci ont été extraites. Ce dernier jeu de données a été utilisé à la fois pour la CM et pour la CB. Il va servir pour la comparaison entre les modèles développés dans ce travail et différents outils disponibles en libre accès. Les tables 10 et 11 reprennent la composition de ces deuxièmes jeux de données de test pour la CM et la CB.

Table 6 – Composition des jeux d'images utilisés pour développer les modèles, pour la classification multi-espèces.

Label	**Entraînement**		**Validation**		**Test**		**Total**	**% jeu de**
	Effectif	**% classe**	**Effectif**	**% classe**	**Effectif**	**% classe**		**données**
Athérure	1457	74,34	115	5,87	388	19,80	1960	2,70
Autre	876	73,24	180	15,05	140	11,71	1196	1,65
Bongo	2312	82,51	235	8,39	255	9,10	2802	3,86
Buffle de forêt	276	69,00	39	9,75	85	21,25	400	0,55
Céphalophe à dos jaune	1716	68,69	261	10,45	521	20,86	2498	3,44
Céphalophe bleu	983	65,01	183	12,10	346	22,88	1512	2,08
Céphalophe rouge	2626	72,20	271	7,45	740	20,35	3637	5,01
Cercopithecidae	3558	67,18	558	10,54	1180	22,28	5296	7,29
Chevrotain aquatique	1138	69,52	60	3,67	439	26,82	1637	2,25
Chimpanzé	1330	65,65	113	5,58	583	28,78	2026	2,79
Civette/Genette/ Nandinie	830	72,30	122	10,63	196	17,07	1148	1,58
Écureuil	1722	71,42	238	9,87	451	18,71	2411	3,32
Eléphant de forêt	1871	59,68	613	19,55	651	20,77	3135	4,32
Gorille	2064	68,48	232	7,70	718	23,82	3014	4,15
Grand félin	306	76,31	33	8,23	62	15,46	401	0,55
Mandrill	619	68,85	112	12,46	168	18,69	899	1,24
Mangouste	1802	77,91	149	6,44	362	15,65	2313	3,18
No_sp	17967	67,51	2961	11,13	5687	21,37	26615	36,63
Oiseau	2913	72,48	250	6,22	856	21,30	4019	5,53
Pangolin	576	64,57	132	14,80	184	20,63	892	1,23
Potamochère	1733	65,17	179	6,73	747	28,09	2659	3,66
Rat géant/Souris	1465	67,11	304	13,93	414	18,96	2183	3,00
Total	50140	69,01	7340	10,10	15173	20,88	72653	100,00

Table 7 – Composition des jeux d'images utilisés pour développer les modèles, pour la classification binaire.

Label	Entraînement		Validation		Test		Total	% jeu de données
	Effectif	% classe	Effectif	% classe	Effectif	% classe		
No_sp	17967	67,51	2961	11,13	5687	21,37	26615	36,63
Animal	32173	69,88	4379	9,51	9486	20,60	46038	63,37
Total	50140	69,01	7340	10,10	15173	20,88	72653	100,00

Table 8 – Composition des jeux de vidéos utilisés pour développer les modèles, pour la classification multi-espèces.

Label	Entraînement		Validation		Test		Total	% jeu de données
	Effectif	% classe	Effectif	% classe	Effectif	% classe		
Athérure	251	71,71	32	9,14	67	19,14	350	5,29
Autre	109	70,32	17	10,97	29	18,71	155	2,34
Bongo	193	72,56	50	18,80	23	8,65	266	4,02
Buffle de forêt	57	72,15	4	5,06	18	22,78	79	1,19
Céphalophe à dos jaune	273	69,11	50	12,66	72	18,23	395	5,97
Céphalophe bleu	214	64,85	41	12,42	75	22,73	330	4,99
Céphalophe rouge	244	67,22	37	10,19	82	22,59	363	5,49
Cercopithecidae	237	68,50	34	9,83	75	21,68	346	5,23
Chevrotain aquatique	175	68,90	13	5,12	66	25,98	254	3,84
Chimpanzé	255	65,72	23	5,93	110	28,35	388	5,87
Civette/Genette/ Nandinie	182	71,37	24	9,41	49	19,22	255	3,85
Écureuil	241	70,26	33	9,62	69	20,12	343	5,19
Eléphant de forêt	208	58,59	62	17,46	85	23,94	355	5,37
Gorille	180	67,42	38	14,23	49	18,35	267	4,04
Grand félin	50	73,53	5	7,35	13	19,12	68	1,03
Mandrill	133	68,56	25	12,89	36	18,56	194	2,93
Mangouste	247	72,86	33	9,73	59	17,40	339	5,12
No_sp	217	66,36	38	11,62	72	22,02	327	4,94
Oiseau	379	66,37	43	7,53	149	26,09	571	8,63
Pangolin	107	73,29	14	9,59	25	17,12	146	2,21
Potamochère	218	65,66	38	11,45	76	22,89	332	5,02
Rat géant/Souris	317	64,43	71	14,43	104	21,14	492	7,44
Total	4487	67,83	725	10,96	1403	21,21	6615	100,00

Table 9 – Composition des jeux de vidéos utilisés pour développer les modèles, pour la classification binaire.

Label	Entraînement		Validation		Test		Total	% jeu de données
	Effectif	% classe	Effectif	% classe	Effectif	% classe		
No_sp	217	66,36	38	11,62	72	22,02	327	4,94
Animal	4270	67,91	687	10,93	1331	21,17	6288	95,06
Total	4487	67,83	725	10,96	1403	21,21	6615	100,00

Table 10 – Composition du jeu de vidéos qui est utilisé pour comparer les différents outils et les modèles développés dans ce travail, pour la classification multi-espèces.

Label	Test	
	Effectif	% jeu de données
Athérure	100	5,70
Autre	77	4,39
Bongo	0	0,00
Buffle de forêt	0	0,00
Céphalophe à dos jaune	100	5,70
Céphalophe bleu	100	5,70
Céphalophe rouge	100	5,70
Cercopithecidae	18	1,03
Chevrotain aquatique	100	5,70
Chimpanzé	100	5,70
Civette/Genette/Nandinie	94	5,36
Écureuil	100	5,70
Eléphant de forêt	100	5,70
Gorille	44	2,51
Grand félin	45	2,56
Mandrill	100	5,70
Mangouste	100	5,70
No_sp	100	5,70
Oiseau	100	5,70
Pangolin	77	4,39
Potamochère	100	5,70
Rat géant/Souris	100	5,70
Total	1755	100,00

Table 11 – Composition du jeu de vidéos qui est utilisé pour comparer les différents outils et les modèles développés dans ce travail, pour la classification binaire.

Label	**Test**	
	Effectif	**% jeu de données**
No_sp	100	5,70
Animal	1655	94,30
Total	1755	100,00

2.5 Annotation des données

Comme expliqué dans la section précédente, les images ont dû être annotées car un animal présent dans une vidéo, ne l'est pas forcément sur toute la durée de celle-ci. L'annotation des images a été réalisée en 2 étapes. Premièrement, le logiciel libre d'accès TimeLapse [4] a été utilisé afin de définir si un animal était présent ou non sur chaque image. Dès qu'un bout de l'animal était visible sur l'image, il était considéré comme présent. Deuxièmement, si une image ne contenait aucun animal, elle a été attribuée la classe « No_sp ». Dans le cas contraire, elle a été versée dans la classe de la vidéo dont elle provient.

2.6 Difficultés liées aux données de pièges photographiques

Les difficultés que peut connaître un modèle d'AP liées aux espèces animales, aux paysages et aux arrière-plans qui ne se trouvent pas dans le jeu de données d'entraînement ont déjà été explicitées au point 1.2.2 de la page 18. Cependant, d'autres difficultés peuvent également intervenir. Les pièges photographiques étant installés dans la nature dans le but de surveiller la faune, la qualité des images et des vidéos va dépendre des conditions météorologiques du milieu et du comportement des animaux.

À cause des conditions météorologiques, certaines vidéos peuvent être très sombres ou très illuminées. Il peut y avoir de la brume (figure 10a) ou de la pluie. De plus, une goutte d'eau peut aussi rester sur l'objectif de la caméra et obstruer le champ de vision de celle-ci. Toutes ces possibilités rendent les animaux moins visibles et peuvent nuire à la bonne classification des vidéos.

En ce qui concerne les animaux, ils peuvent être assez petits (figure 10b), partiellement cachés par la végétation (figure 10c) ou cachés dans l'obscurité (figure 10d). Dans certain cas, seule une partie de l'animal est visible (figure 10e), il peut aussi se trouver loin dans le champ de la caméra (figure 10f) ou se trouver très proche à tel point qu'on ne puisse plus reconnaître l'animal (figure 10g). Les animaux qui bougent très vite peuvent également créer du flou. Contrairement aux difficultés dûes aux conditions météorologiques, celles engendrées par le comportement des animaux vont impacter la classification à l'échelle de l'image sans forcément perturber la classification à l'échelle de la vidéo. En effet, ces perturbations peuvent se produire que sur une partie de la vidéo.

Une dernière difficulté qui peut altérer les performances d'un modèle d'AP est que certaines classes se ressemblent. C'est notamment le cas des gorilles avec les chimpanzés, des classes de céphalophes entre elles, de certaines classes de céphalophes avec les bongos ou des rats géants/souris avec les athérures.

Afin d'atténuer ces problèmes, des solutions pourraient être d'augmenter la taille du jeu de données d'entraînement ou de faire de l'augmentation de données afin d'accroître le nombre de cas difficiles dans le jeu de données.

4. `https://saul.cpsc.ucalgary.ca/timelapse/` (consulté le 04/05/2022)

(a) Oiseaux peu visibles à cause de la buée

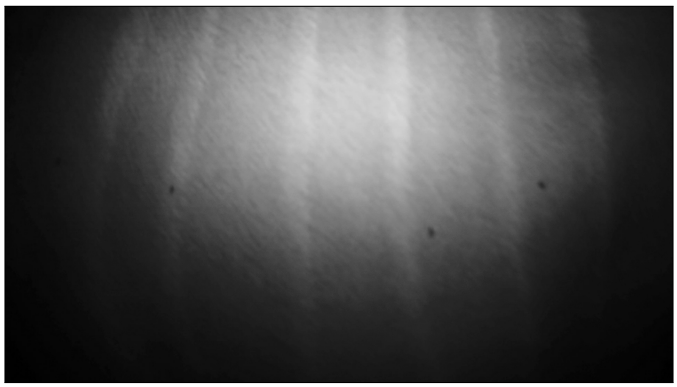
(b) Petite souris

(c) Cercopithecidae cachés par la végétation

(d) Mangouste peu visible dans l'obscurité

(e) Seule une partie du chimpanzé est visible

(f) Mangouste loin de la caméra



(g) Bongo proche de la caméra

Figure 10 – Exemples de cas compliqués rencontrés dans les différents jeux de données constitués.

2.7 Architectures des modèles d'apprentissage profond

Dans le but de classifier des vidéos de PP installés en forêt tropicale, en Afrique centrale, grâce à l'apprentissage profond, plusieurs architectures ont été testées :

— architecture numéro 1 : ResNet2D + ConvLSTM ;
— architecture numéro 2 : ResNet2D + LSTM ;
— architecture numéro 3 : ResNet2D 1 ;
— architecture numéro 4 : ResNet2D 2 ;
— architecture numéro 5 : ResNet2D 3 ;
— architecture numéro 6 : ResNet2D 2+3 ;
— architecture numéro 7 : ResNet3D 1 ;
— architecture numéro 8 : ResNet3D 2 .

La première est une combinaison d'un réseau de neurones convolutifs avec un ConvLSTM. La seconde est une combinaison d'un CNN avec un LSTM. Quatre architectures de CNN, basées sur de la classification d'images ont également été utilisées. Enfin, des premiers tests ont été effectués avec des CNN à 3 dimensions. Ces tests sont présentés en annexe 2 car ils n'ont pas pu faire l'objet d'une étude approfondie. Dans toutes les architectures testées, les fonctions d'activation intermédiaires sont des fonctions ReLU *rectified linear unit* et les fonctions d'activation finales sont des fonctions *Softmax*. La fonction ReLu change les valeurs négatives en 0. La fonction *Softmax* transforme le score des classes en probabilité d'appartenir à ces classes.

2.7.1 *Convolutiuonnal Long Short-Term Memory (ConvLSTM)* et *Long Short-Term Memory (LSTM)*

La première architecture qui a été étudiée est composée d'un extracteur de caractéristiques, un ConvLSTM et un perceptron multicouche. L'extracteur de caractéristiques est celui du ResNet-18. Les ResNets sont des CNN qui ont déjà été fortement utilisés pour classer des images de PP (Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Nguyen *et al.*, 2017 ; Norouzzadeh *et al.*, 2018 ; Tabak *et al.*, 2019 ; Willi *et al.*, 2019 ; Zualkernan *et al.*, 2020 ; Whytock *et al.*, 2021 ; Yang *et al.*, 2021) et qui ont donné des résultats prometteurs. Le Resnet-18 est l'architecture ResNet la moins profonde et donc la moins complexe. Il a pour avantage de performer assez bien pour de la classification d'images et de nécessiter de moins de ressources informatiques. Il faut également moins de temps pour l'entraîner. Le ConvLSTM doit gérer la dimension temporelle. Il doit apprendre quelles caractéristiques, extraites par le ResNet-18, doivent être gardées ou oubliées au cours du temps. À la fin du traitement de l'entièreté des images d'une vidéo, une prédiction sera faite grâce aux informations accumulées par le ConVLSTM dans sa cellule mémoire. Après le passage de la dernière image d'une vidéo dans le ConvLSTM, sa sortie va passer dans une couche *batch normalization*, puis dans une couche *adaptive average pooling* et sera ensuite aplatie dans un vecteur. Celui-ci est ensuite traité par un perceptron multicouche qui est le classificateur. Ce classificateur est constitué d'une couche d'entrée avec 512 neurones, 3 couches cachées (ou *hidden layers* en anglais) qui possèdent respectivement 256, 128 et 64 neurones, et une couche de sortie qui possède 22 neurones correspondant aux 22 classes définies dans la table 4. Cette architecture est identique à celle utilisée par Sudhakaran et Lanz (2017b) qui a détecté si des vidéos étaient violentes ou non. Les seules différences sont qu'ils avaient utilisé AlexNet comme extracteur de caractéristiques et que la couche de sortie du perceptron multicouche ne possédait que 2 neurones.

La deuxième architecture testée est une variante de la première. Un ResNet-18 sert d'extracteur de caractéristiques. Les cartes de caractéristiques sont ensuite aplaties en un vecteur qui est traité par un LSTM. Ce dernier est suivi par un perceptron multicouche qui sert de classificateur. Celui-ci est constitué d'une couche d'entrée avec 300 neurones, de 2 couches cachées avec 256 et 128 neurones et d'une couche de sortie avec 22 neurones. Cette architecture a été inspirée par une architecture mise en libre accès sur GitHub [5].

5. `https://github.com/pranoyr/cnn-lstm#readme` (consulté le 22/04/2022)

Pour les architectures 1 et 2, l'entrée de ces modèles est un ensemble d'images échantillonnées sur une vidéo et la sortie est une classe prédite qui correspond directement à la vidéo.

2.7.2 Architectures de modèles basées sur des images

Les 3 prochaines architectures (numéro 3,4 et 5) sont basées sur des CNN afin de classifier des images. Comme dit plus haut, la classification d'images de PP grâce à des CNN a déjà été réalisée à de multiples reprises (Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Nguyen *et al.*, 2017 ; Norouzzadeh *et al.*, 2018 ; Tabak *et al.*, 2019 ; Willi *et al.*, 2019 ; Zualkernan *et al.*, 2020 ; Whytock *et al.*, 2021 ; Yang *et al.*, 2021). Pour les 3 architectures, des CNN ResNets ont été utilisés pour les mêmes raisons que celles citées précédemment. L'extracteur de caractéristiques du CNN est resté intact mais le classificateur a été modifié. Ce dernier est un perceptron multicouche qui est constitué d'une couche en entrée avec un nombre de neurones égal au nombre de cartes de caractéristiques extraites par la première partie du réseau, une couche cachée qui comprend 256 neurones et une couche de sortie qui compte autant de neurones que de classes pouvant être prédites. Du *dropout* avec une probabilité qui a varié au cours des entraînements, a été utilisé sur la couche d'entrée et la couche cachée. Sur les 3 architectures, la numéro 3 sert à la CM (22 classes)(objectif 1), la numéro 4 sert à la CB (2 classes) (objectif 2) et la numéro 5 fait de la CM sans la classe « No_sp » (21 classes).

L'architecture numéro 6 est une combinaison des architectures numéro 4 et 5 dont le but est de faire de la CM en 2 étapes. D'abord, les images sont classifiées soit comme des animaux, soit comme des arrière-plans. Les animaux sont ensuite triés parmi les 21 classes restantes. Ceci a déjà été réalisé avec succès par Norouzzadeh *et al.* (2018) et Willi *et al.* (2019). L'architecture 5 a été développée dans le seul but de faire partie de cette architecture numéro 6.

Un autre avantage du développement de ces architectures est que les modèles entraînés grâce à celles-ci pourront servir lors de l'entraînement de modèles pour les architectures 1 et 2 (ResNet2D + ConvLTSM et ResNet2D + LSTM) car toutes ces architectures possèdent le même extracteur de caractéristiques. Ceci sera explicité plus bas.

2.8 Détails d'implémentation

2.8.1 Gestion de l'aléatoire

L'aléatoire a été fixé pour pouvoir comparer les différents entraînements de modèles. En effet, si cela n'est pas fait, la différence de résultats obtenue entre deux entraînements pourrait n'être que la conséquence de comportements aléatoires comme par exemple, l'ordre d'envoi des vidéos ou des images au modèle lors des entraînements.

2.8.2 Les transformations de données

Les images ont subi des transformations. Un redimensionnement et une normalisation ont été opérés. Pour le redimensionnement, le rapport largeur sur hauteur a été maintenu afin de ne pas déformer les images. Si les dimensions demandées ne correspondent pas à ce rapport, du *padding* a été imposé. C'est-à-dire que des pixels qui possède une valeur de 0 vont être ajoutés à l'image pour atteindre les dimensions désirées. La taille des images a varié au cours des entraînements. La normalisation est une opération qui va donner une valeur entre 0 et 1 à chaque pixel de chaque canal de l'image. Cela permet de mettre les images à la même échelle, d'améliorer les performances du modèle et qu'il converge plus vite (Elgendy, 2020). La normalisation a été réalisée grâce à la moyenne et l'écart-type du jeu de données « ImageNet ».

2.8.3 L'augmentation de données

Comme défini plus haut, l'augmentation de données est une méthode qui sert à faire augmenter la taille du jeu de données d'entraînement de manière artificielle, grâce à des transformations qui sont appliquées

sur les données, de façon aléatoire, à chaque époque. Chaque transformation possède une probabilité d'être appliquée à une image à une certaine époque. L'augmentation de données peut permettre au modèle d'atteindre de meilleurs résultats car il pourra mieux généraliser l'information contenue dans un jeu de données et donc limiter le surapprentissage (Elgendy, 2020).

Dans le cadre de ce travail, un flip horizontal avec une probabilité d'être appliquée de 50 % a été implémenté pour tous les entraînements. En plus de cela, deux combinaisons de transformations ont également été testées. La première est composée d'une modification du contraste et/ou de la luminosité avec une probabilité de 25 % et un ajout de flou avec une probabilité de 25 %. La deuxième combinaison est semblable à la première mais avec de la distorsion avec une probabilité de 25 % en plus. Ces transformations ont notamment été utilisées par Whytock *et al.* (2021). Les fonctions et leurs arguments qui ont été utilisés sont détaillés dans l'annexe 1. Lors des entraînements de modèles qui classifient directement les vidéos, les transformations se sont faites à l'échelle de la vidéo. C'est-à-dire qu'elles ont été appliquées à toutes les images d'une vidéo de la même manière. Alors que pour les modèles basés sur les images, les images d'une même vidéo étaient traitées indépendamment les unes des autres.

2.8.4 L'optimiseur

L'optimiseur utilisé est le *Stochastic gradient descent*. C'est celui qui est le plus souvent utilisé en AP (LeCun *et al.*, 2015; Elgendy, 2020).

2.8.5 La fonction de perte et les poids des classes

La fonction de perte ou le calcul de l'erreur qui a été choisi est la *cross entropy loss* qui est calculée entre la distribution de probabilité de la cible et celle de la prédiction.

Un poids a été attribué à chaque classe comme l'ont fait Delplanque *et al.* (2022) pour de la détection faunique sur des images aériennes avec des effectifs de classes déséquilibrés. Ces poids sont inversement proportionnels aux effectifs des classes du jeu de données d'entraînement. Ils ont été calculés grâce à l'équation 2 :

$$P_i = \frac{max(\{n_1, ..., n_i, ..., n_k\})}{n_i} \tag{2}$$

Où P_i est le poids de la classe i, n_i est l'effectif de la classe i dans le jeu de données d'entraînement et k est le nombre de classes. Les poids des classes vont impacter le calcul de la *cross entropy loss*. Plus l'effectif d'une classe sera important plus l'erreur sera diminuée et plus les paramètres du modèle seront ajustés de manière moins importante. En effet, le calcul du gradient dépend de l'erreur calculée. En d'autres mots, une erreur de prédiction sur une donnée qui appartient à la classe majoritaire aura moins d'importance pour le modèle qu'une erreur de prédiction sur une donnée de la classe minoritaire. Cette technique permet de restreindre l'influence des classes majoritaires dans un jeu de données déséquilibré.

2.8.6 Les hyperparamètres des modèles

Les hyperparamètres d'un modèle sont ses paramètres qui sont ajustés manuellement par l'homme au cours des entraînements afin de définir la combinaison d'hyperparamètres qui maximise les performances du modèle. Par exemple, la taille de l'image, la probabilité des *dropouts*, le nombre de couches cachées et de neurones dans un perceptron multicouche, le *momentum*, la taille du *batch* et le taux d'apprentissage (ou *learning rate* (LR)) sont des hyperparamètres.

Le *momentum* se souvient des gradients calculés précédemment, si les gradients antérieurs pointent dans la même direction que le nouveau gradient, les paramètres du modèle seront modifiés de manière plus importante. Par contre, si le nouveau gradient est de direction différente, les poids seront modifiés de manière moins importante. Le *momentum* permet de converger plus rapidement vers l'erreur minimum et

d'éviter d'être bloqué dans un minimum local. Dans le cadre de ce travail, le *momentum* a été fixé à sa valeur par défaut, c'est-à-dire 0,9, pour tous les entraînements réalisés.

Comme expliqué dans la section 1.2.2 de la page 18, les paramètres (poids et biais) du modèle vont être ajustés au cours de l'entraînement, en fonction de l'erreur calculée. L'importance de ces ajustements va être dictée par le taux d'apprentissage. Plus il est grand, plus les paramètres seront ajustés de manière importante et inversement. Un LR très petit va nous garantir d'atteindre l'erreur minimum mais nécessitera un nombre d'époques infini. Au contraire, un LR trop grand ne permettra pas d'arriver à l'erreur minimum. C'est l'hyperparamètre le plus important lors du développement d'un modèle d'AP (Elgendy, 2020). Pour ce travail, le LR de départ n'a pas varié car un planificateur a été utilisé.

2.8.7 Le planificateur

Pour gérer le LR de manière optimale, un planificateur (ou *scheduler* en anglais) a été utilisé. Le planificateur choisi est le « ReduceLROnPlateau » avec une patience de 4 et un LR minimum de 10^6. Si après 5 époques, l'exactitude globale maximale sur le jeu de données de validation n'avait augmenté, le LR était divisé par 10.

2.8.8 La sortie des modèles

La sortie des modèles est un fichier Excel qui reprend le nom de chaque vidéo, la prédiction top1 (CB et CM) et la prédiction top3 (CM).

2.9 Entraînement des modèles

Lors des entraînements, l'erreur sur le jeu de données d'entraînement ainsi que l'erreur et l'exactitude sur le jeu de données de validation ont été calculées à chaque époque. Les poids et biais du modèle en cours d'entraînement n'étaient sauvegardés uniquement si l'exactitude globale sur le jeu de validation dépassait le maximum des exactitudes globales calculées aux époques précédentes. Les modèles étaient programmés pour s'entraîner pendant 50 époques. Néanmoins, quand l'exactitude globale sur le jeu de validation maximale n'augmentait plus pendant un nombre assez important d'époques (une dizaine), l'entraînement était arrêté.

En ce qui concerne les architectures ResNet2D + ConvLSTM et ResNet2D + LSTM, un grand nombre de tests d'entraînement de modèles ont été réalisés. Les classifications, binaire et multi-espèces, ont été abordées. Les images échantillonnées sur une vidéo ou la différence de ces images ont été utilisées comme entrée du modèle. Différents taux d'apprentissage, tailles de *batch*, tailles d'image, nombres d'images échantillonnées par seconde sur la vidéo et différentes profondeurs de ResNets2D ont également été testés. Quand l'animal n'est pas présent sur toute la durée de la vidéo, il est le plus souvent présent sur les premières secondes de la vidéo et puis disparaît. Des tests pour lesquels les vidéos ont été lues en sens inverse ont donc été réalisés pour que les dernières images analysées par le ConvLSTM ou le LSTM soient celles qui comprennent l'animal. Les caractéristiques qui lui correspondent doivent donc avoir plus de chance de se retrouver dans la mémoire interne du ConvLSTM et du LSTM lors de la décision finale.

Deux configurations ont également été étudiées. Dans la première, l'extracteur de caractéristiques est préentraîné sur le jeu de données « ImageNet » et les poids du reste du modèle ont été initialisés aléatoirement. Dans la deuxième configuration, du transfert d'apprentissage a également été réalisé. Les poids de l'extracteur de caractéristiques ont été remplacés par les poids de l'extracteur de caractéristiques du meilleur modèle entraîné grâce à l'architecture ResNet2D 1. Les poids de l'extracteur de caractéristiques ont ensuite été gelés et n'ont donc plus été ajustés pendant l'entraînement. Cela doit permettre au modèle de s'entraîner plus facilement car il y a moins de poids à ajuster. En effet, seuls les poids du ConvLSTM ou du LSTM et du classificateur sont à ajuster.

Pour l'architecture ResNet2D 1, de nombreux modèles ont été entraînés. Ces entraînements résultent de la variation de plusieurs hyperparamètres : le nombre d'images qui constituent les jeux de données, la taille de l'image, la taille du *batch*, la probabilité du *dropout* et la combinaison d'augmentations de données utilisée. Plusieurs profondeurs de ResNets ont également été testées : 18, 34, 50. Pour tous les entraînements, du transfert d'apprentissage a été effectué. Les poids et biais de l'extracteur de caractéristiques ont été préentraînés sur le jeu de données « ImageNet ». Une fois que la meilleure combinaison d'hyperparamètres a été définie, un modèle a été entraîné avec cette dernière pour les architectures ResNet2D 2 et ResNet2D 3. Les performances des modèles des architectures ResNet2D 1, ResNet2D 2, ResNet2D 3 et ResNet2D 2+3 ont ensuite pu être définies grâce au jeu de données de test.

2.10 Conversion d'un ensemble de prédictions (images) en une unique prédiction (vidéo)

Dans le but de convertir les prédictions faites à l'échelle des images en une prédiction à l'échelle de la vidéo, différentes méthodes empiriques ont été expérimentées. Les résultats obtenus sur le jeu de données de validation ont permis de comparer ces méthodes. Si le modèle est parfait, si toutes les images d'une vidéo sont prédites comme vides, alors la vidéo l'est aussi (classe « No_sp »), sinon la prédiction finale est l'espèce qui a été la plus prédite sur la vidéo.

2.10.1 Classification multi-espèces

Pour la classification multi-espèces, six méthodes ont été développées et comparées afin de définir la classe la plus probable à laquelle appartient chaque vidéo :

— méthode numéro 1 : majorité de prédictions ;

— méthode numéro 2 : majorité de 1,0 ;

— méthode numéro 3 : moyenne ;

— méthode numéro 4 : moyenne -1/5 des prédictions ;

— méthode numéro 5 : moyenne x fréquence ;

— méthode numéro 6 : addition des probabilités.

Pour chacune de ces méthodes, la prédiction finale est la classe « No_sp » seulement si toutes les prédictions faites sur les images extraites d'une vidéo correspondent à cette même classe.

Dans la méthode 1 : « majorité de prédictions », la prédiction et sa probabilité sont retenues pour chaque image échantillonnée sur une vidéo. Le nombre de prédictions pour chaque classe est calculé et les probabilités sont sommées par classe. La prédiction finale est la classe qui compte le plus de prédictions. S'il y a une égalité, la classe avec la plus grande somme de probabilités est choisie.

La méthode numéro 2 est une variante de la première. Les mêmes informations sont extraites des prédictions sur les images. Cependant, le nombre de fois, pour chaque classe, que la probabilité de la prédiction est égale à 1,0 est également calculé. La prédiction pour la vidéo correspond à la classe qui a le plus de fois été prédite avec une probabilité de 1,0 car cela veut dire que le modèle est sûr de sa prédiction. S'il y a égalité, c'est le nombre de prédictions par classe qui tranche. S'il y a toujours égalité, c'est la somme des probabilités qui détermine la décision finale.

Pour la troisième méthode, les mêmes informations que précédemment sont extraites et la moyenne des probabilités est calculée pour chaque classe. La prédiction finale est la classe qui possède la moyenne la plus haute.

La méthode 4 : « moyenne -1/5 des prédictions » est une variante de la méthode trois. Les moyennes sont également calculées mais si une classe a été prédite moins de fois que le nombre d'images échantillonnées sur la vidéo divisé par 5, elle n'est pas prise en compte. Si aucune classe n'a été prédite assez de fois, la méthode trois est utilisée. La méthode quatre permet d'éviter de définir comme prédiction finale, une classe qui n'aurait été prédite qu'une seule fois avec une grande probabilité.

La méthode 5 essaie de prendre en compte le nombre de prédictions faites pour une classe et les probabilités avec lesquelles elle est prédite. Pour ce faire un indice a été calculé pour chaque classe grâce à la formule suivante :

$$Indice_i = [\exp(\frac{1}{10} * \frac{nombre\ de\ prédictions_i}{Nombre\ d'images})] * moyenne_i \quad (3)$$

Où i représente les classes et le nombres d'images est le nombre d'images échantillonnées sur la vidéo. La prédiction finale est la classe qui possède le plus grand indice. La formule 3 a été créée dans le cadre de ce travail. Le but recherché est qu'une classe qui possède une plus petite moyenne mais qui est prédite plus de fois puisse être choisie. La moyenne d'une classe va être modifiée en fonction du nombre de prédiction de cette classe. L'utilisation d'une exponentielle permet de donner un plus grand avantage si le nombre de prédictions est plus important. Le facteur $\frac{1}{10}$ sert lui à atténuer la modification de la moyenne.

Pour la dernière méthode (6), la probabilité qu'une image appartienne à une classe a été enregistrée pour chacune d'entre elles, pour chaque image. Ces probabilités ont été sommées. La prédiction à l'échelle de la vidéo correspond à la classe qui possède la somme de probabilités la plus importante.

Les méthodes numéro 1 et 2 sont surtout influencées par le nombre de fois que les classes sont prédites. Les méthodes numéro 3, 4 et 6 se basent, elles, sur les probabilités des classes. Alors que la méthode numéro 5 essaie de prendre ces 2 aspects en considération.

Six autres méthodes ont également été créées afin de déterminer les 3 classes auxquelles chaque vidéo a le plus de chances d'appartenir :

— méthode numéro 1 : score ;
— méthode numéro 2 : score avec avantage pour 1,0 ;
— méthode numéro 3 : majorité de 1,0 ;
— méthode numéro 4 : somme top3 ;
— méthode numéro 5 : moyenne top3 ;
— méthode numéro 6 : addition des probabilités.

Pour la méthode 1 : « score », les trois classes les plus probables ainsi que leur probabilité sont extraites pour chaque image d'une vidéo. Un score est ensuite calculé pour chaque classe. Pour chaque image, la classe la plus probable reçoit 3 points, la seconde 2 points et la troisième 1 point. Dans le cas où la probabilité de la classe est égale à 0, cette dernière ne reçoit aucun point. Les trois classes qui possèdent les plus grands scores sont ensuite sélectionnées.

La méthode numéro 2 est une variante de la première. Les scores sont calculés de la même manière sauf que si la probabilité d'une classe est égale à 1,0 (prédiction sûr), celle-ci reçoit 4,5 points.

Pour la méthode 3 : « majorité de 1,0 », le nombre de fois qu'une classe est prédite avec une probabilité de 1,0 est comptabilisé. Les 3 classes les plus de fois prédites avec certitude sont choisies. Si cela n'est pas possible (car aucune classe n'a été prédite avec certitude par exemple), le reste ou l'entièreté des classes à sélectionner sont déterminées grâce au score décrit à la méthode 1.

Pour la méthode 4, les trois classes les plus probables ainsi que leur probabilité sont également extraites pour chaque image d'une vidéo. La somme des probabilités, par classe est ensuite calculée. Les 3 prédictions finales sont les classes qui possèdent les plus grandes sommes de probabilités.

La méthode 5 : « moyenne top3 » est une variante de la méthode 4, la différence est qu'au lieu de calculer la somme des probabilités pour chaque classe, c'est la moyenne qui est calculée.

Pour la dernière méthode (6), la probabilité de chaque classe pour chaque image de la vidéo a été extraite. Ces probabilités ont ensuite été sommées. Les 3 classes sélectionnées sont celles qui possèdent les sommes de probabilités les plus importantes.

Pour toutes les méthodes et pour la plupart des vidéos, la classe « No_sp » est sélectionnée dans les 3 classes les plus probables. Ceci est logique, beaucoup de vidéos qui n'appartiennent pas à cette classe comprennent des images vides. Pour minimiser ce problème, le score de la classe « No_sp » a été calculé pour chaque vidéo du jeu de données de validation, pour chaque méthode. Ce score a ensuite été divisé par le nombre d'images extraites des vidéos pour que ce score puisse être comparé entre les vidéos quel que soit le nombre d'images qui sont extraites de celles-ci. Un seuil a ensuite été déterminé pour chaque méthode. Si le score de la classe « No_sp » est inférieur au seuil, cette dernière n'est pas prise en compte. Le seuil a été défini sur le jeu de données de validation, pour garder environ 95 % des vidéos vides qui étaient classées comme vides, attribuées correctement.

2.10.2 Classification binaire

Pour la classification binaire, trois méthodes ont été testées afin de définir si un animal est présent ou non à l'échelle de la vidéo :

— méthode numéro 1 : classique ;

— méthode numéro 2 : moyenne ;

— méthode numéro 3 : classique avec un seuil.

Pour la méthode 1 : « classique », une vidéo est vide seulement si toutes les images échantillonnées dans cette dernière ont été prédites comme telles.

Pour la seconde méthode, la moyenne des probabilités de chaque image d'une vidéo, pour les 2 classes sont calculées. La prédiction finale est celle qui possède la plus grande moyenne.

La dernière méthode (3) est un mixte des 2 autres méthodes. Pour chaque vidéo, la moyenne des probabilités de chaque image, pour la classe « Animal » est calculée. Ensuite, la première méthode est appliquée. Si la prédiction est « Animal » et que la moyenne des probabilités pour cette classe est inférieure à un certain seuil, la prédiction change et la vidéo est considérée comme vide. Le seuil a été défini afin de maximiser l'exactitude moyenne sur le jeu de données de validation.

2.11 Métriques et performances des modèles

Dans le but d'évaluer et de comparer les performances des différents modèles, la matrice de confusion et certaines métriques (exactitude, précision et score F1) ont été calculées.

La matrice de confusion ou table de contingence est un résumé des résultats prédits par un modèle. Elle compare les vérités terrains et les prédictions. Cela permet notamment de détecter les confusions que peut avoir le modèle entre plusieurs classes.

L'exactitude ou le rappel ou encore la sensibilité représente le nombre d'observations correctement classées par rapport au nombre total d'observations. Il est possible de calculer l'exactitude globale (équation 4), ainsi que l'exactitude par classe (équation 5).

$$Exactitude = \frac{Nombre\ d'observations\ correctement\ classées}{Nombre\ d'observations\ total} \tag{4}$$

$$Exactitude\ de\ la\ classe\ i = \frac{Nombre\ d'observations\ correctement\ classées\ dans\ la\ classe\ i}{Nombre\ d'observations\ total\ de\ la\ classe\ i} \tag{5}$$

L'exactitude moyenne a également été définie. C'est la moyenne des rappels des classes. C'est une métrique moins biaisée que l'exactitude globale quand des jeux de données fortement déséquilibrés sont utilisés. L'exactitude moyenne permet de s'assurer que le modèle performe bien pour chaque classe. Tandis que l'exactitude globale permet d'évaluer si le modèle fait des prédictions correctes en général.

La précision d'une classe (équation 6) correspond au nombre d'observations correctement attribuées à une classe sur le nombre total d'observations attribuées à cette même classe. Cela décrit la probabilité qu'une observation soit correctement classée si elle est classée dans une certaine classe.

$$Précision\ de\ la\ classe\ i = \frac{Nombre\ d'observations\ correctement\ classées\ dans\ la\ classe\ i}{Nombre\ d'observations\ total\ classées\ dans\ la\ classe\ i} \tag{6}$$

Le score F1 d'une classe (équation 7) est la moyenne harmonique du rappel et de la précision de celle-ci. C'est donc une unique métrique qui caractérise à la fois l'exactitude et la précision.

$$Score\ F1 = \frac{2 * exactitude * précision}{exactitude + précision} \tag{7}$$

2.12 Comparaison avec différents outils

Des outils qui sont disponibles en libre accès, ainsi que les meilleurs modèles pour les architectures ResNet2D 1 et ResNet2D 2 ont été essayés sur le second jeu de données de test (table 10 et 10). Celui-ci doit être vu comme un nouveau jeu de données qui revient tout juste du terrain et qui doit être analysé afin de réaliser une identification complète des données. Le test de ces outils permettra de comparer leurs performances avec celles des modèles créés au travers de ce travail.

Trois outils ont été sélectionnés : Wildlife Insights, Mbaza AI et Zamba Cloud. Ils ont été choisis car ils peuvent être appliqués sur des vidéos ou des images de PP qui ont été installés en forêt tropicale en Afrique centrale. De plus, ces outils ont été entraînés, entre autres, pour les classes utilisées dans ce travail.

Si la prédiction faite par ces outils est réalisée à l'échelle de l'image, les méthodes qui ont été sélectionnées pour les modèles ResNet2D 1 (classification multi-espèces) et ResNet2D 2 (classification binaire) seront utilisées afin de déterminer la prédiction à l'échelle de la vidéo.

2.12.1 Wildlife Insights

Wildlife Insights [6] est une plateforme développée par Google pour identifier des images de PP. Elle utilise un modèle d'AP. Le but de cet outil est d'être capable de classifier correctement des images d'un très grand nombre d'espèces animales différentes dans tous les environnements naturels possibles et de n'importe quelle zone géographique. Leur base de données est composée de plus de 15 millions d'images pour 993 classes. Cet outil sera utilisé uniquement pour faire de la CB. La reconnaissance d'espèces fonctionne avec des seuils de confiance. Si la probabilité qu'une image appartienne à une certaine classe n'est pas assez grande, l'image n'est pas classée ou est classée dans un sous-ensemble (« mammifère », « animal », « pas de résultat », ...). C'est pourquoi la CM n'a pas été explorée avec cette application. La sortie de cette plateforme est un tableur Excel avec le nom des images, la présence d'un animal ou non et la classification des images.

6. `https://www.wildlifeinsights.org/` (consulté le 17/07/2022)

2.12.2 Mbaza AI

Mbaza AI[7] est un logiciel qui utilise un modèle d'AP dans le but de classifier des images de PP. Il a été développé par Whytock *et al.* (2021). Son domaine d'utilisation se restreint aux espèces de mammifères terrestres et d'oiseaux vivant dans les forêts tropicales situées en Afrique centrale. Le modèle a été développé grâce à 347 120 images qui correspondent à 27 classes. Ce modèle a servi à la comparaison par rapport à la CB et la CM. La sortie de ce logiciel est également un tableur Excel qui reprend le nom des images, les 3 classes prédites les plus probables ainsi que leur probabilité.

2.12.3 Zamba Cloud

Zamba Cloud[8] est une plateforme qui permet de classifier des vidéos capturées par des PP grâce à un modèle d'AP. Il est capable de classer des vidéos de 11 espèces d'Europe occidentale et de 32 espèces d'Afrique centrale et d'Afrique de l'est. Sa base de données est constituée de 27 000 vidéos. Pour être en adéquation avec les classes de Zamba Cloud, celles présentées dans ce travail ont légèrement été modifiées. Les classes « Céphalophe à dos jaune », « Céphalophe bleu » et « Céphalophe rouge » ont été rassemblées dans une classe « Céphalophe ». De même, les écureuils et les rats géant/souris ont été rassemblés dans une classe « Rongeur ». Enfin la classe « Cercopithecidae » et la classe « Mandrill » ont été fusionnées dans une classe « Cercopithecidae ». Zamba Cloud possède deux modèles d'AP pour les données d'Afrique de l'ouest : un modèle basé sur l'architecture SlowFast et un modèle basé sur des images. Le second modèle a été utilisé car ses performances sont légèrement meilleures dans le cadre de la CM. Ce modèle a donc été testé pour la CM et la CB. La sortie de cet outil est de nouveau un tableau Excel qui reprend le nom des vidéos, les 3 classes prédites les plus probables, leur probabilité et la probabilité de chaque classe.

7. `https://github.com/Appsilon/mbaza` (consulté le 17/07/2022)
8. `https://www.zambacloud.com/` (consulté le 17/07/2022)

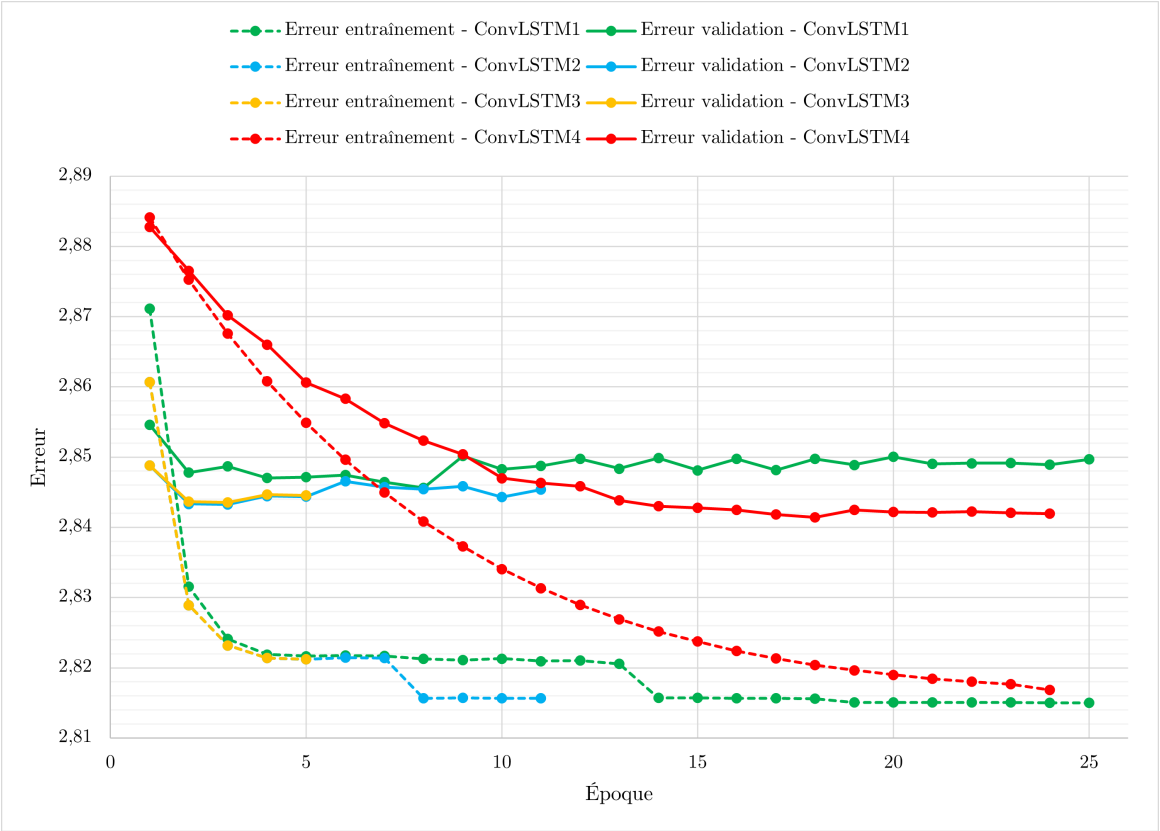
3 Résultats

3.1 *Convolutiuonnal Long Short-Term Memory (ConvLSTM)* et *Long Short-Term Memory (LSTM)*

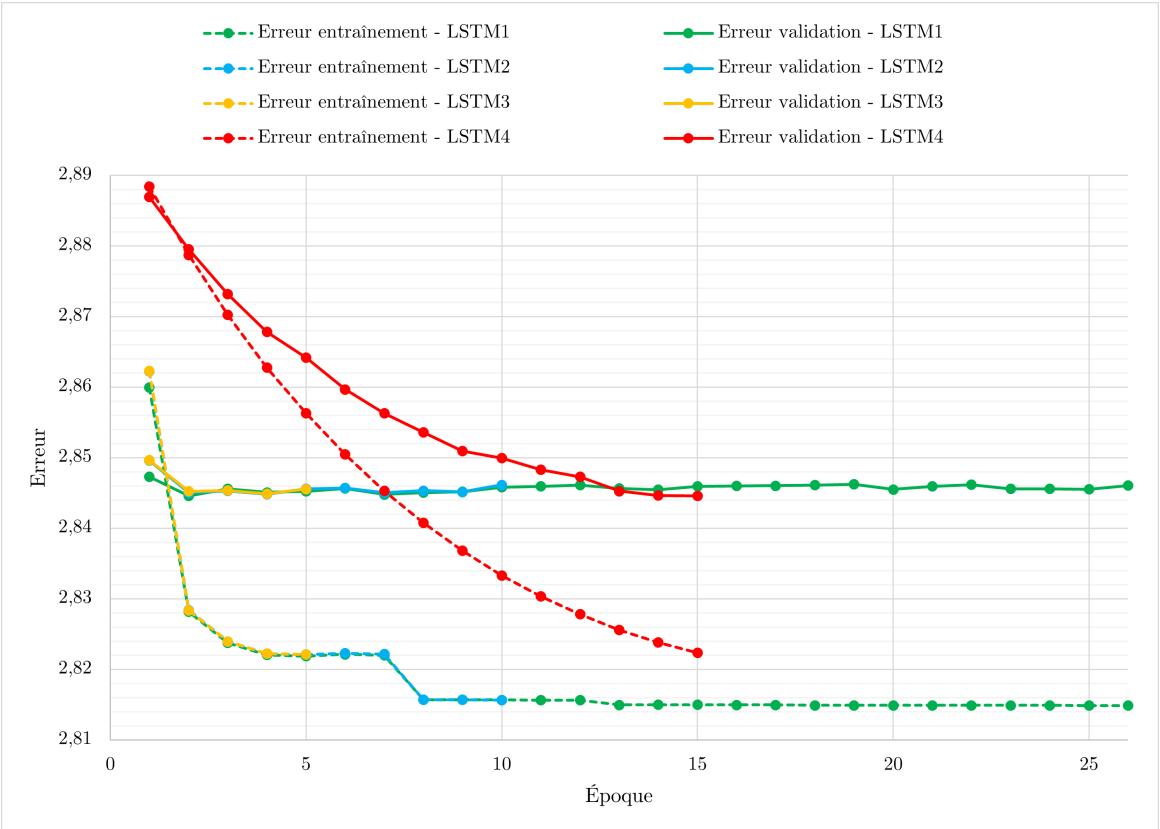
De nombreux tests d'entraînement de modèles ont été réalisés avec les architectures ResNet2D + ConvLSTM et ResNet2D + LSTM. L'évolution de l'erreur sur les jeux de données d'entraînement et de validation en fonction de l'époque est représentée sur la figure 11a pour l'architecture ResNet2D + ConvLSTM et sur la figure 11b pour l'architecture ResNet2D + LSTM. Seulement quatre entraînements par architecture sont représentés sur ces figures car ils sont représentatifs de tous les autres. Ces modèles ont été entraînés pour de la classification multi-espèces avec un ResNet-18 comme extracteur de caractéristiques et avec 2 images échantillonnées par seconde sur les vidéos. Les hyperparamètres qui caractérisent ses entraînements sont repris dans la table 12. Sur maximum 25 époques, l'erreur sur les jeux de données d'entraînement et de validation varie entre 2,81 et 2,89. En d'autres mots, les modèles ne parviennent pas à s'entraîner. Les modèles ne convergent pas vers une erreur minimale. Le fait d'utiliser les poids du meilleur modèle de l'architecture ResNet2D 1 pour l'extracteur de caractéristiques et de geler ces poids n'a pas réglé le problème. L'ajustement des hyperparamètres (LR, taille de l'image, sens de lecture de la vidéo, taille du *batch*, nombre d'images échantillonnées par seconde,...) n'a pas non plus été bénéfique.

Table 12 – Hyperparamètres des modèles entraînés pour les architectures ResNet2D + ConvLSTM et ResNet2D + LSTM.

Modèle ConvLSTM	**Modèle LSTM**	LR	**Sens de lecture de la vidéo**	**Taille de l'image**	**Extracteur de caractéristique**
ConvLSTM1	LSTM1	0,001	À l'endroit	512 x 1024	Entraîné
ConvLSTM2	LSTM2	0,001	À l'endroit	512 x 1024	Gelé
ConvLSTM3	LSTM3	0,001	À l'envers	512 x 1024	Gelé
ConvLSTM4	LSTM4	0,0001	À l'endroit	512 x 1024	Gelé



(a) ResNet2D + ConvLSTM



(b) ResNet2D + LSTM

Figure 11 – Évolution de l'erreur sur les jeux de données d'entraînement et de validation, des modèles entraînés grâce aux architectures ResNet2D + ConvLSTM (a) et ResNet2D + LSTM (b) pour de la classification multi-espèces.

3.2 Architectures de modèles basées sur des images

3.2.1 Résultats de la validation

Différents modèles ont été entraînés pour l'architecture ResNet2D 1. Les entraînements effectués sont décrits dans le tableau 14. Ce dernier reprend certains hyperparamètres des modèles tels que le nombre d'images que composent les jeux de données d'entraînement, de validation et de test, le nombre de classes qui peuvent être prédites, la taille de l'image en entrée (en pixels), la taille du *batch*, le LR, la probabilité qui caractérise le *dropout*, les augmentations de données appliquées, le ResNet2D utilisé et le nombre d'époques qu'a duré l'entraînement. En plus de ces hyperparamètres, l'exactitude globale maximale sur le jeu de données de validation est également contenue dans la table 14. La ligne écrite en rouge correspond au meilleur modèle.

Le modèle Resnet18_1_10 est celui qui a atteint la plus grande exactitude globale maximale sur le jeu de données de validation. Cette dernière vaut 71,81 %. C'est le modèle qui a été retenu pour l'architecture ResNet2D 1. C'est un ResNet-18 entraîné pendant 43 époques avec une taille d'images en entrée de 1080 pixels de haut et de 1920 pixels de large, un LR de 0,001, une taille de *batch* de 2, un *dropout* avec une probabilité de 0,5 et sans augmentation de données utilisées.

Les résultats de la validation à l'échelle de l'image pour les architectures ResNet2D 1, ResNet2D 2, ResNet2D 3 et ResNet2D 2+3 sont présentés dans la table 13. Ces résultats ont été calculés pour les deux types de classifications (multi-espèces et binaire). Le modèle de l'architecture ResNet2D 2+3 a atteint de moins bonnes performances. Celui de l'architecture ResNet2D 1 a été retenu pour la CM. Il possède une exactitude globale de 71,81 %. Pour la CB, c'est le modèle de l'architecture ResNet2D 2 qui a été choisi. Son exactitude globale équivaut à 85,76 %.

Table 13 – Résultats de la validation pour les architectures ResNet2D 1, ResNet2D 2, ResNet2D 3 et ResNet2D 2+3, pour les classifications multi-espèces et binaire, à l'échelle de l'image.

Classification	Métrique	ResNet2D 1	ResNet2D 2	ResNet2D 3	ResNet2D 2+3
multi-espèces	Exactitude globale	**71,81 %**	/	39,37 %	52,92 %
	Exactitude moyenne	**55,60 %**	/	36,20 %	37,60 %
Binaire	Exactitude globale	85,46 %	**85,76 %**	/	/
	Exactitude moyenne	87,01 %	**85,67 %**	/	/

Remarque : Les valeurs écrites en rouge correspondent aux architectures sélectionnées.

En ce qui concerne le passage des prédictions sur les images à une prédiction à l'échelle de la vidéo, 6 méthodes ont été testées afin de déterminer la classe la plus probable et 6 autres méthodes ont été testées afin de trouver les 3 classes les plus probables, pour la CM. La classification des images a été réalisée grâce au meilleur modèle de l'architecture ResNet2D 1. Pour ce qui est de la CB, 3 méthodes ont été testées afin de définir la classe la plus probable grâce à l'architecture ResNet2D 2. Les résultats de ces tests sont repris dans les tableaux 15 et 16. L'exactitude globale top1 et top3 ont été choisies comme métriques de comparaison pour la CM. Par contre, c'est l'exactitude moyenne qui a été choisie pour la CB car le jeu de données est fortement déséquilibré.

Table 14 – Hyperparamètres et exactitude globale maximale sur le jeu d'images de validation, des différents modèles entraînés avec l'architecture ResNet2D 1.

Modèle	**Nombre d'images**	**Nombre de classes**	**Taille de l'image**	**Taille du *batch***	**LR**	***Dropout***	**Augmentation de données**	**ResNet**	**Nombre d'époques**	**Exactitude validation**
Resnet18_1_1	58481	18	512 x 1024	8	0,001	0,0	/	ResNet-18	37	69,54 %
Resnet18_1_2	58481	18	512 x 1024	8	0,001	0,0	Combinaison 2	ResNet-18	27	67,32 %
Resnet18_1_3	58481	18	512 x 1024	8	0,001	0,0	Combinaison 1	ResNet-18	25	69,30 %
Resnet18_1_4	58481	18	512 x 1024	4	0,001	0,0	/	ResNet-18	27	64,33 %
Resnet18_1_5	58481	18	512 x 1024	8	0,001	0,0	/	ResNet-18	50	70,92 %
Resnet18_1_6	72653	22	512 x 1024	8	0,001	0,5	/	ResNet-18	25	64,55 %
Resnet18_1_7	72653	22	512 x 1024	8	0,001	0,6	/	ResNet-18	20	63,15 %
Resnet18_1_8	72653	22	512 x 1024	8	0,001	0,4	/	ResNet-18	41	64,65 %
Resnet18_1_9	72653	22	1080 x 1920	4	0,001	0,5	/	ResNet-18	32	66,79 %
Resnet18_1_10	**72653**	**22**	**1080 x 1920**	**2**	**0,001**	**0,5**	**/**	**ResNet-18**	**43**	**71,81 %**
Resnet18_1_11	72653	22	1080 x 1920	1	0,001	0,5	/	ResNet-18	11	20,49 %
Resnet34_1_12	72653	22	1080 x 1920	2	0,001	0,5	/	ResNet-34	43	61,75 %
Resnet50_1_13	72653	22	1080 x 1920	1	0,001	0,5	/	ResNet-50	7	15,37 %
Resnet50_1_14	72653	22	512 x 1024	2	0,001	0,5	/	ResNet-50	32	64,66 %
Resnet18_1_15	72653	22	512 x 1024	2	0,001	0,5	/	ResNet-18	32	62,06 %
Resnet18_1_16	72653	22	1080 x 1920	2	0,001	0,5	Combinaison 1	ResNet-18	42	64,93 %
Resnet18_1_17	72653	22	1080 x 1920	2	0,001	0,5	Combinaison 2	ResNet-18	50	69,99 %

La méthode numéro 5 (« moyenne x fréquence ») a été choisie afin de définir la classe la plus probable et la méthode numéro 6 (« addition des probabilités ») pour définir les 3 classes les plus probables, à l'échelle de la vidéo, pour la CM. Grâce à ces méthodes, une exactitude globale top1 de 65,38 % et une exactitude globale top3 de 80,97 % ont été obtenues. Pour la CB, c'est la méthode numéro 3 (« classique avec un seuil ») qui a permis d'atteindre les meilleurs résultats avec une exactitude globale de 75,91 %

Table 15 – Résultats sur le jeu de données de validation, du test des différentes méthodes qui servent à convertir les prédictions à l'échelle de l'image en une prédiction à l'échelle de la vidéo, dans le cadre de la classification multi-espèces.

Méthode top1	**N°1**	**N°2**	**N°3**	**N°4**	**N°5**	**N°6**
Exactitude globale top1	64,41 %	60,97 %	63,86 %	64,55 %	**65,38 %**	63,59 %
Méthode top3	**N°1**	**N°2**	**N°3**	**N°4**	**N°5**	**N°6**
Seuil	1,7	1,70	1,70	0,15	0,15	**0,10**
Exactitude globale top3	76,41 %	76,41 %	77,66 %	80,83 %	80,14 %	**80,97 %**

Remarque : Les valeurs écrites en rouge correspondent aux méthodes sélectionnées.

Table 16 – Résultats sur le jeu de données de validation, du test des différentes méthodes qui servent à convertir les prédictions à l'échelle de l'image en une prédiction à l'échelle de la vidéo, dans le cadre de la classification binaire.

Méthode	**N°1**	**N°2**	**N°3**
Rappel « No_sp »	44,74 %	57,89 %	**68,42 %**
Rappel « Animal »	95,05 %	89,67 %	**83,41 %**
Exactitude globale	92,41 %	88,00 %	**82,62 %**
Exactitude moyenne	69,89 %	73,78 %	**75,91 %**

Remarque : Les valeurs écrites en rouge correspondent à la méthode sélectionnée.

L'étude de l'impact du nombre d'images échantillonnées par seconde sur les vidéos a également été réalisée grâce au jeu de données de validation. La comparaison des résultats des meilleurs modèles et des meilleurs méthodes pour les deux types de classifications, en fonction du nombre d'images échantillonnées par seconde peut être faite à l'aide des tables 17 et 18.

Pour la CM, l'optimal est d'échantillonner 1 image par seconde des vidéos. Cela donne une exactitude top1 de 65,38 % et une exactitude top3 de 80,97 %. À 0,5 image par seconde, l'exactitude globale top1 et top3 sont inférieures (62,76 % et 80,00 %). Tandis qu'à 2 images par seconde, l'exactitude globale top1 est inférieure (64,97 %) et l'exactitude globale de top3 est légèrement supérieure (81,24 %). Cependant, le temps de traitement est presque doublé (x 1,889). Pour la CB, l'optimal a été défini à 2 images par échantillonnées seconde sur les vidéos. Ceci permet d'atteindre une exactitude moyenne de 77,15 %. Malgré que le temps de traitement soit plus long, cet optimum induit un gain de 2,19 % et de 0,71 % d'exactitude moyenne par rapport aux options à 0,5 et 1 image échantillonnée par seconde, respectivement.

Table 17 – Impact du nombre d'images échantillonnées par seconde dans les vidéos, sur les performances des modèles, pour la classification multi-espèces, sur le jeu de vidéos de validation.

Images par seconde	**0,5**	**1**	**2**
Exactitude globale top1	62,76 %	**65,38 %**	64,97 %
Exactitude globale top3	80,00 %	**80,97 %**	81,24 %
Temps [minutes]	12	**18**	34

Remarque : Les valeurs écrites en rouge correspondent au nombre d'images extraites par seconde des vidéos, sélectionné.

Table 18 – Impact du nombre d'images échantillonnées par seconde dans les vidéos, sur les performances des modèles, pour la classification binaire, sur le jeu de vidéos de validation.

Images par seconde	**0,5**	**1**	**2**
Exactitude moyenne	74,96 %	76,44 %	**77,15 %**
Seuil	0,8	0,9	**0,75**
Temps [minutes]	11	18	**34**

Remarque : Les valeurs écrites en rouge correspondent au nombre d'images extraites par seconde des vidéos, sélectionné.

3.2.2 Résultats du test

3.2.2.1 Classification multi-espèces

Les performances générales du modèle ResNet2D 1 sur le jeu de données de test (vidéos) sont reprises dans le tableau 19. Il possède une exactitude globale top1 de 67,93 % et top3 de 82,68 %. Son exactitude moyenne top1 et top3 sont respectivement de 61,75 % et de 78,10 %. Sa vitesse de traitement est d'environ 6 images par seconde.

Table 19 – Performances globales du modèle ResNet2D 1 sur le jeu de vidéos de test (classification multi-espèces).

Modèle	**Exactitude globale top1**	**Exactitude moyenne top1**	**Exactitude globale top3**	**Exactitude moyenne top3**	**Vitesse de traitement**
ResNet2D 1	67,93 %	61,75 %	82,68 %	78,10 %	6 images par seconde

La figure 12 présente la matrice de confusion obtenue grâce au modèle de l'architecture ResNet2D 1 appliqué sur le jeu de données de test (vidéos). Ce modèle a tendance à classer de manière erronée un bon nombre de vidéos dans la classe « No_sp ». Cela concerne surtout les vidéos d'athérures, d'écureuils, d'oiseaux et de rats géants/souris. Plusieurs vidéos de buffles de forêt (6 sur 18) ont été classées comme des potamochères. Il y a également une confusion évidente entre les gorilles et les chimpanzés.

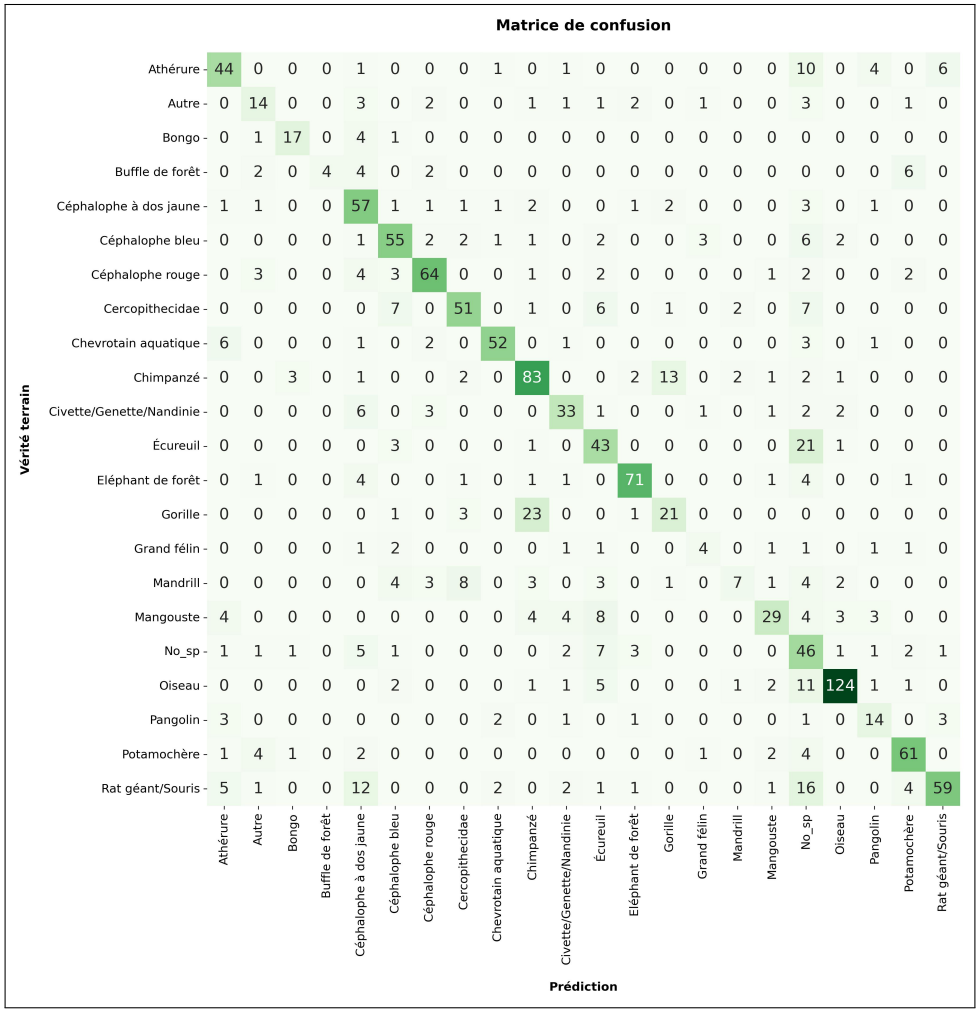


Figure 12 – Matrice de confusion obtenue grâce au modèle ResNet2D 1 appliqué sur le jeu de vidéos de test (classification multi-espèces).

Les performances par classe du modèle ResNet2D 1 sur le jeu de données de test (vidéos) se retrouvent dans la table 20. Les classes qui possèdent les plus hauts rappels sont les éléphants de forêt (83,53 %), les oiseaux (83,22 %) et les potamochères (80,26 %). À contrario, les mandrills, les buffles de forêt et les grands félins sont les classes avec les rappels les plus bas (respectivement 19,44 %, 22,22 % et 30,77 %). En terme de précision, ce sont les buffles de forêt (100,00 %), les oiseaux (91,18 %) et les chevrotains aquatiques (88,14 %) qui obtiennent les meilleurs résultats, à l'inverse des « No_sp » (30,67 %), grands félins (40,00 %) et « Autre » (50,00 %). Les scores F1 les plus grands sont ceux des Oiseaux (87,02 %), des éléphants de forêt (85,03 %) et des chevrotains aquatiques (83,20 %). Les mandrills (29,17 %), les grands félins (34,78 %) et les buffles de forêt (36,36 %) ont eux les scores F1 les plus petits. Les classes qui possèdent les plus petits rappels top3 sont les classes de mandrills (36,11 %), de grands félins (38,46 %) et « Autre » (62,07 %). Alors que les rappels top3 maximaux sont ceux des classes d'écureuils (94,20 %), d'oiseaux (93,29 %) et de céphalophes bleus (92,00 %).

Table 20 – Performances par classe du modèle ResNet2D 1 sur le jeu de vidéos de test (classification multi-espèces).

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]	**Rappel top3** [%]
Athérure	65,67	67,69	66,67	83,58
Autre	48,28	50,00	49,12	62,07
Bongo	73,91	77,27	75,56	91,30
Buffle de forêt	22,22	100,00	36,36	66,67
Céphalophe à dos jaune	79,17	53,77	64,04	84,72
Céphalophe bleu	73,33	68,75	70,97	92,00
Céphalophe rouge	78,05	81,01	79,50	85,37
Cercopithecidae	68,00	75,00	71,33	78,67
Chevrotain aquatique	78,79	88,14	83,20	86,36
Chimpanzé	75,45	68,03	71,55	87,27
Civette/Genette/Nandinie	67,35	68,75	68,04	71,43
Écureuil	62,32	53,75	57,72	94,20
Eléphant de forêt	83,53	86,59	85,03	88,24
Gorille	42,86	55,26	48,28	79,59
Grand félin	30,77	40,00	34,78	38,46
Mandrill	19,44	58,33	29,17	36,11
Mangouste	49,15	72,50	58,59	76,27
No_sp	63,89	30,67	41,44	79,17
Oiseau	83,22	91,18	87,02	93,29
Pangolin	56,00	53,85	54,90	76,00
Potamochère	80,26	77,22	78,71	89,47
Rat géant/Souris	56,73	85,51	68,21	77,88

3.2.2.2 Classification binaire

Pour la classification binaire, les performances globales du modèle ResNet2D 2 sur le jeu de données de test (vidéos) sont reprises dans la table 21. Ce modèle possède une exactitude globale de 84,89 % et une exactitude moyenne de 77,58 %. Il traite environ 6 images par seconde.

Table 21 – Performances globales du modèle ResNet2D 2 sur le jeu de vidéos de test (classification binaire).

Modèle	Exactitude globale top1	Exactitude moyenne top1	Vitesse de traitement
ResNet2D 2	84,89 %	77,58 %	6 images par seconde

La figure 13 correspond à la matrice de confusion obtenue grâce au modèle ResNet2D 2 appliqué sur le jeu de données de test (vidéos). La table 22 reprend les performances par classe du modèle ResNet2D 2 sur le jeu de données de test (vidéos). Le modèle obtient de bons résultats pour la classe « Animal ». Le rappel de cette classe est de 85,73 %, la précision est de 98,11 % et le score F1 est de 91,50 %. En revanche, le modèle est moins performant pour la classe « No_sp ». Si le rappel de cette classe est plutôt correct (69,44 %), la précision n'est que de 20,93 % et le score F1 de 32,05 %.

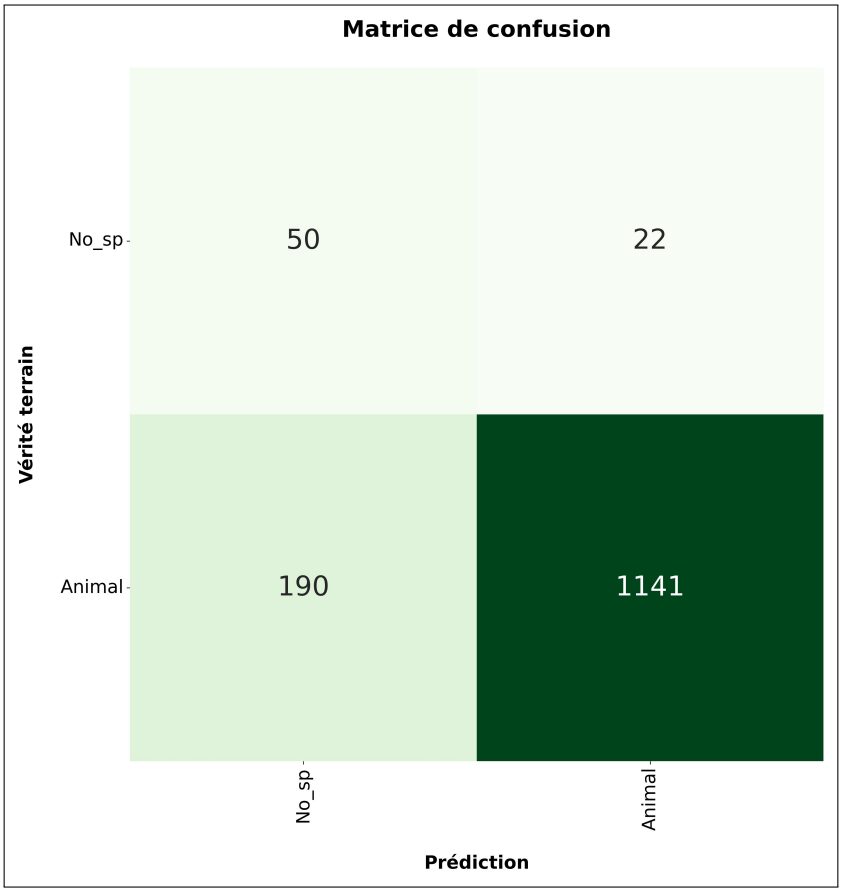


Figure 13 – Matrice de confusion obtenue grâce au modèle ResNet2D 2 appliqué sur le jeu de vidéos de test (classification binaire).

Table 22 – Performances par classe du modèle ResNet2D 2 sur le jeu de vidéos de test (classification binaire).

Classe	**Rappel [%]**	**Précision [%]**	**Score F1 [%]**
No_sp	69,44	20,83	32,05
Animal	85,73	98,11	91,50

3.3 Comparaison avec différents outils

3.3.1 Classification multi-espèces

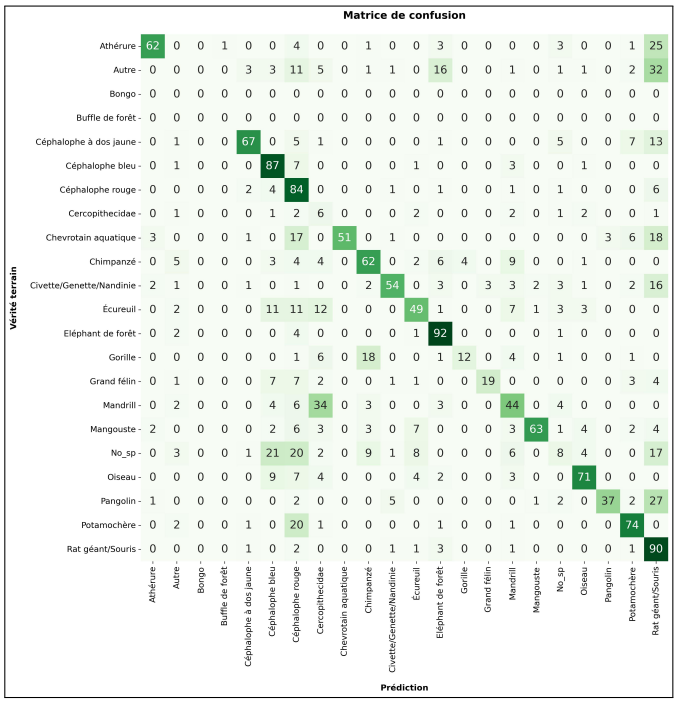
La table 23 reprend les performances globales des différents outils (Mbaza AI, Zamba Cloud) et du modèle ResNet2D 1, utilisés pour la classification multi-espèces. Le modèle Zamba Cloud est celui qui a le moins bien performé. Il possède les valeurs les plus basses pour toutes les métriques exposées dans le tableau 23. Le modèle ResNet2D 1 possède la plus grande exactitude globale top1 (60,97 %) et top3 (77,38 %) et la plus grande exactitude moyenne top3 (72,62 %). Le modèle Mbaza AI a lui la plus grande exactitude moyenne de top1 (55,62 %), celle-ci est légèrement supérieure à celle du modèle ResNet2D 1 (55,59 %).

Table 23 – Performances générales des différents outils utilisés pour la classification multi-espèces, sur le second jeu de test.

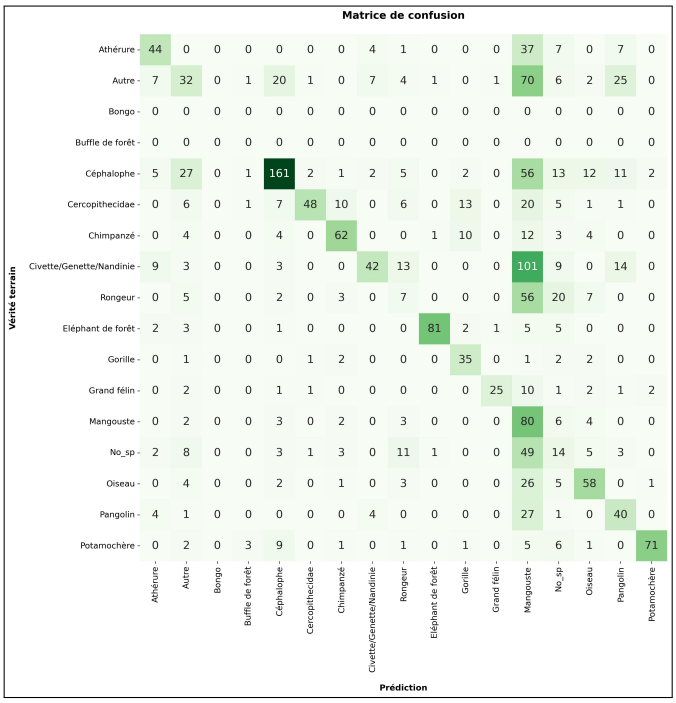
Modèle	**Exactitude globale top1**	**Exactitude moyenne top1**	**Exactitude globale top3**	**Exactitude moyenne top3**
Mbaza AI	58,80 %	55,62 %	72,14 %	69,51 %
Zamba	46,09 %	50,81 %	65,81 %	68,22 %
ResNet2D 1	60,97 %	55,59 %	77,38 %	72,62 %

Les matrices de confusion obtenues sur le second jeu de données de test (table 10) grâce aux modèles Mbaza AI, Zamba Cloud et ResNet2D 1, sont reprises dans la figure 14. Les performances (rappels, précisions, scores f1 et rappels top3) par classe de ces 3 modèles sont reprises dans les tables 24 (Mbaza AI), 25 (Zamba Cloud) et 26 (ResNet2D 1).

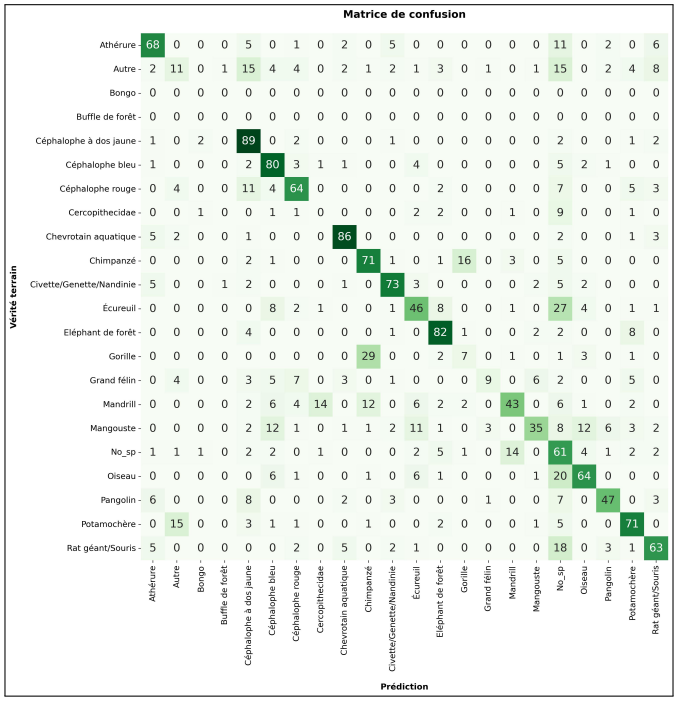
Comme il y a 0 vidéo de bongos et de buffles de forêt dans le second jeu de données de test, les performances des modèles pour ces classes seront nulles.



(a) Mbaza AI



(b) Zamba Cloud



(c) ResNet2D 1

Figure 14 – Matrices de confusion obtenues sur le second jeu de données de test, grâce au modèle Mbaza AI (a), au modèle Zamba Cloud (b) et au modèle de ResNet2D 1 (c) pour la classification multi-espèces.

En ce qui concerne le modèle Mbaza AI, si les vidéos des différentes classes de céphalophes sont plutôt bien classées, les vidéos des autres espèces (notamment les écureuils, les mandrills et les « Autre ») ainsi que les vidéos vides sont également souvent attribuées aux classes de céphalophes. Le même constat peut être tiré pour les Rats géants/Souris. Beaucoup de vidéos d'autres espèces (comme les athérures, les « Autre » et les pangolins) se retrouvent dans cette classe. Il y a également des confusions avec les gorilles qui sont souvent classés comme des chimpanzés et avec les mandrills qui sont parfois classés dans les cercopithecidae (figure 14a). Les classes qui possèdent le plus haut rappel sont les éléphants de forêt (92,00 %) et les rats

géants/souris (90,00 %). Les chevrotains aquatiques et les mangoustes possèdent les plus grandes précisions avec respectivement 100,00 % et 94,03 %. En terme de score F1, ce sont les éléphants de forêt (78,97 %), les céphalophes à dos jaune (75,71 %), les oiseaux (75,53 %) et les mangoustes (75,45 %) qui sortent du lot (table 24).

Table 24 – Performances par classe du modèle Mbaza AI sur le second jeu de test pour la classification multi-espèces.

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]	**Rappel top3** [%]
Athérure	62,00	88,57	72,94	76,00
Autre	0,00	0,00	0,00	0,00
Bongo	0,00	0,00	0,00	0,00
Buffle de forêt	0,00	0,00	0,00	0,00
Céphalophe à dos jaune	67,00	87,01	75,71	77,00
Céphalophe bleu	87,00	57,24	69,05	97,00
Céphalophe rouge	84,00	38,01	52,34	90,00
Cercopithecidae	33,33	7,50	12,24	50,00
Chevrotain aquatique	51,00	100,00	67,55	60,00
Chimpanzé	62,00	62,63	62,31	73,00
Civette/Genette/Nandinie	57,45	83,08	67,92	76,60
Écureuil	49,00	64,47	55,68	64,00
Eléphant de forêt	92,00	69,17	78,97	95,00
Gorille	27,27	75,00	40,00	70,45
Grand félin	42,22	86,36	56,72	46,67
Mandrill	44,00	50,00	46,81	79,00
Mangouste	63,00	94,03	75,45	78,00
No_sp	8,00	23,53	11,94	39,00
Oiseau	71,00	80,68	75,53	80,00
Pangolin	48,05	92,50	63,25	58,44
Potamochère	74,00	73,27	73,63	85,00
Rat géant/Souris	90,00	35,57	50,99	95,00

Le modèle Zamba Cloud a tendance à classer beaucoup de vidéos dans la classe des mangoustes. La même observation peut être faite pour la classe « No_sp » mais à moindre échelle. Il y a également une confusion entre les gorilles, les chimpanzés et les mandrills (table 25). Les plus grands rappels par classe correspondent aux éléphants de forêt (81,00 %) et aux mangoustes (80,00 %) alors que les rongeurs, les « No_sp » et les « Autre » possèdent des rappels de respectivement 8,00 %, 14,00 % et 18,08 %. Les classes qui sont prédites avec le plus de précision sont les éléphants de forêt (96,43 %), les potamochères (93,42 %) et les grands félins (92,59 %). Les éléphants de forêt (88,04 %) et les potamochères (80,68 %) ont également les plus haut scores F1 (table 25).

Table 25 – Performances par classe du modèle Zamba Cloud sur le second jeu de test pour la classification multi-espèces.

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]	**Rappel top3** [%]
Athérure	44,00	60,27	50,87	52,00
Autre	18,08	32,00	23,10	62,15
Bongo	0,00	0,00	0,00	0,00
Buffle de forêt	0,00	0,00	0,00	0,00
Céphalophe	53,67	74,54	62,40	76,33
Cercopithecidae	40,68	88,89	55,81	52,54
Chimpanzé	62,00	72,94	67,03	76,00
Civette/Genette/Nandinie	44,68	71,19	54,90	60,64
Rongeur	8,00	29,63	12,60	31,00
Eléphant de forêt	81,00	96,43	88,04	86,00
Gorille	79,55	55,56	65,42	84,09
Grand félin	55,56	92,59	69,44	68,89
Mangouste	80,00	14,41	24,43	89,00
No_sp	14,00	13,59	13,79	39,00
Oiseau	58,00	59,18	58,59	76,00
Pangolin	51,95	39,22	44,69	89,61
Potamochère	71,00	93,42	80,68	80,00

En ce qui concerne le modèle ResNet2D 1, beaucoup de vidéos de différentes classes sont prédites comme des vidéos vides. Il a du mal à classifier les vidéos de mangoustes qu'il classe dans les céphalophes bleus, les écureuils, les oiseaux ou les « No_sp ». Plusieurs vidéos de potamochères (15/100) ont été classées dans les « Autre » et certaines vidéos des « Autre » (15/77) ont été classées dans les céphalophes à dos jaune. Il y a également de la confusion au niveau des singes (gorilles, mandrills, chimpanzés et cercopithecidae) (table 26). Les classes qui possèdent le rappel le plus important sont les céphalophes à dos jaune (89,00 %), les chevrotains aquatiques (86,00 %) et les éléphants de forêt (82,00 %). Par contre, aucun cercopithecidae n'a été classé correctement. Les précisions les plus importantes sont pour les chevrotains aquatiques (83,50 %) et les civettes/genettes/nandinies (79,35 %). Les chevrotains aquatiques (84,73 %), les civettes/genettes/nandinies (78,49 %) et les éléphants de forêt (77,73 %) possèdent les plus hauts scores F1 (table 26).

Table 26 – Performances par classe du modèle ResNet2D 1 sur le second jeu de test pour la classification multi-espèces.

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]	**Rappel top3** [%]
Athérure	68,00	72,34	70,10	82,00
Autre	14,29	29,73	19,30	37,66
Bongo	0,00	0,00	0,00	0,00
Buffle de forêt	0,00	0,00	0,00	0,00
Céphalophe à dos jaune	89,00	58,94	70,92	93,00
Céphalophe bleu	80,00	61,54	69,57	87,00
Céphalophe rouge	64,00	68,82	66,32	79,00
Cercopithecidae	0,00	0,00	0,00	5,56
Chevrotain aquatique	86,00	83,50	84,73	91,00
Chimpanzé	71,00	61,21	65,74	85,00
Civette/Genette/Nandinie	77,66	79,35	78,49	87,23
Écureuil	46,00	56,10	50,55	85,00
Eléphant de forêt	82,00	73,87	77,73	90,00
Gorille	15,91	25,93	19,72	65,91
Grand félin	20,00	64,29	30,51	37,78
Mandrill	43,00	68,25	52,76	67,00
Mangouste	35,00	72,92	47,30	60,00
No_sp	61,00	27,98	38,36	71,00
Oiseau	64,00	69,57	66,67	77,00
Pangolin	61,04	75,81	67,63	79,22
Potamochère	71,00	66,98	68,93	86,00
Rat géant/Souris	63,00	67,74	65,28	86,00

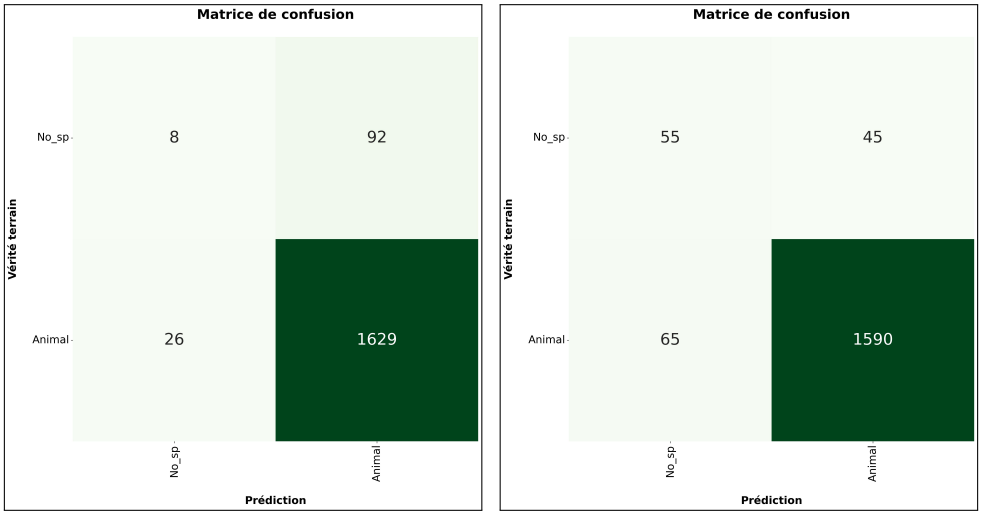
3.3.2 Classification binaire

Les performances globales de chaque outil utilisé (Mbaza AI, Wildlife Insights, Zamba Cloud et ResNet2D 2) pour la classification binaire sont reprises dans la table 27. Le modèle qui a le mieux performé est Wildlife Insights. C'est le modèle qui a la plus grande exactitude globale (93,73 %) et moyenne (75,54 %). Le modèle ResNet2D 2 possède la plus faible exactitude globale (84,73 %) mais il possède aussi la deuxième plus haute exactitude moyenne (68,88 %).

Table 27 – Performances générales des différents outils utilisés pour la classification binaire sur le second jeu de test.

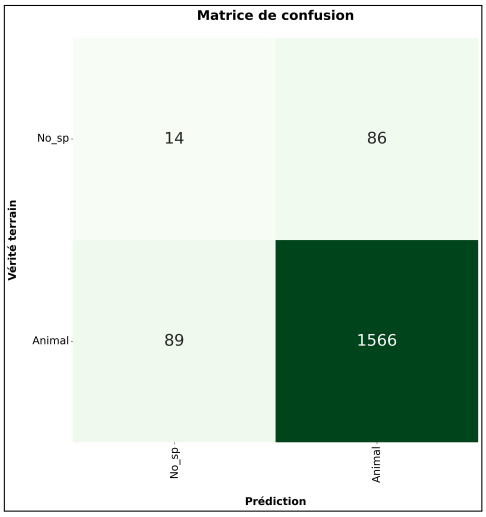
Modèle	Exactitude globale	Exactitude moyenne
Mbaza AI	93,28 %	53,21 %
Wildlife Insights	93,73 %	75,54 %
Zamba Cloud	90,03 %	54,31 %
ResNet2D 2	84,73 %	68,88 %

Les matrices de confusion obtenues grâce aux différents outils testés sur le second jeu de test sont présentées dans la figure 15. Le modèle Mbaza AI a tendance à classer la majorité des vidéos (1721 sur 1755) dans la classe « Animal » (figure 15c). Le modèle ResNet2D 2 est celui qui prédit le plus de vidéos vides (270 sur 1755) mais il se trompe assez souvent (219 fautes sur 270)(la figure 15d).

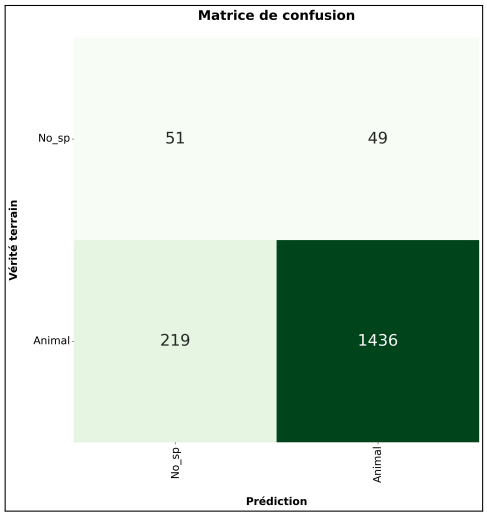


(a) Mbaza AI

(b) Wildlife Insights



(c) Zamba



(d) ResNet2D 2

Figure 15 – Matrices de confusions obtenues sur le second jeu de test grâce au modèle Mbaza AI (a), au modèle Wildlife Insights (b), au modèle Zamba Cloud (c) et au modèle ResNet2D 2 (d), pour la classification binaire.

La table 28 reprend les performances (rappels, précisions et scores F1) par classe de chaque outil testé. Le modèle Wildlife Insights possède le plus grand rappel (55,00 %), la plus grande précision (45,83 %) et

le plus grand score F1 (50,00 %) pour la classe « No_sp ». Il a également le deuxième plus grand rappel (96,07 %), la plus grande précision (97,25 %) et le plus grand score F1 (96,66 %) pour la classe « Animal ». Le rappel de classe « Animal » maximal (98,43 %) a été atteint grâce au modèle Mbaza AI. Le modèle ResNet2D 2 est caractérisé par les moins bons résultats (rappel de 86,77 %, précision de 96,70 % et score F1 de 91,46 %) pour la classe « Animal » mais les deuxièmes meilleurs résultats (rappel de 51,00 %, précision de 18,89 % et score F1 de 27,57 %) pour la classe « No_sp ».

Table 28 – Performances par classe des différents outils utilisés sur le second jeu de test pour la classification binaire.

Modèle	**Classe**	**Rappel [%]**	**Précision [%]**	**Score F1 [%]**
Mbaza AI	No_sp	8,00	23,53	11,94
	Animal	98,43	94,65	96,50
Wildlife Insights	No_sp	55,00	45,83	50,00
	Animal	96,07	97,25	96,66
Zamba Cloud	No_sp	14,00	13,59	13,79
	Animal	94,62	94,79	94,71
ResNet2D 2	No_sp	51,00	18,89	27,57
	Animal	86,77	96,70	91,46

4 Discussion

4.1 *Convolutiuonnal Long Short-Term Memory (ConvLSTM)* et *Long Short-Term Memory (LSTM)*

Malgré tous les tests effectués, les modèles entraînés avec les architectures ResNet2D + ConvLSTM et ResNet2D + LSTM n'ont jamais convergé vers une erreur minimale. Les modèles ne parviennent pas à s'entraîner. Le fait de reprendre les poids du meilleur modèle de l'architecture ResNet2D 1 et de geler ces poids pour l'extracteur de caractéristiques n'a pas arrangé le problème. Ces modèles sont probablement exposés au problème de la disparition du gradient. En effet, il a été remarqué que les poids de l'extracteur de caractéristiques et du ConvLSTM (ou du LSTM) ne variaient presque pas entre 2 ajustements de poids. Seuls ceux du perceptron multicouche subissent un changement significatif.

Pour expliquer cela, la première hypothèse est que le nombre de données utilisées pour entraîner les modèles étaient trop peu important. Cela semble tout de même peu probable. Car même avec peu de données, le modèle devrait s'entraîner un minimum sans arriver à de bons résultats. L'autre hypothèse est que les animaux ne sont pas présents sur assez d'images par vidéo ou qu'ils sont trop petits. Cela pourrait avoir pour conséquence que trop peu de caractéristiques d'animaux ne soient présentes dans la mémoire interne du ConvLSTM ou du LSTM lors de la prise de décision du modèle.

4.2 Architectures de modèles basée sur des images

4.2.1 Classification multi-espèces

Au regard des résultats des différents modèles entraînés grâce à l'architecture ResNet2D 1 (table 14), les hyperparamètres les plus influents sur les performances des modèles ont été la taille du *batch* et la taille de l'image. Un *batch* composé de moins d'images a été privilégié. Par contre, une taille de *batch* égale à une image a fait diminuer drastiquement l'exactitude globale maximale sur le jeu de données de validation. Utiliser des images plus grandes comme entrée du modèle a permis d'atteindre des meilleurs résultats. Le fait de prendre des tailles d'images plus grandes a pour conséquence que les animaux sont

représentés par plus de pixels dans l'image. Il est donc probablement plus facile pour le modèle d'identifier des caractéristiques d'animaux. Cela a dû favoriser la reconnaissance d'animaux petits, lointains, cachés par la végétation ou les animaux dont seule une partie de leur corps était visible. Le nombre d'images utilisées pour entraîner les modèles est également une caractéristique sensible. L'augmentation de la taille du jeu de données d'entraînement a tendance à améliorer les performances du modèle. La probabilité qui caractérise le *dropout* a eu un impact plus léger sur les résultats. Les augmentations de données testées n'ont pas été avantageuses dans le cadre de ce travail. Cependant, celles-ci pourraient peut-être s'avérer bénéfiques si elles étaient appliquées à une base de données plus fournie, avec plus de variabilité. L'utilisation de réseaux de neurones convolutifs plus profonds n'a pas non plus été bénéfique. Cela est en contradiction avec la littérature (Gomez Villa *et al.*, 2016 ; Gomez Villa *et al.*, 2017 ; Norouzzadeh *et al.*, 2018). Néanmoins, le ResNet-34 a rarement été utilisé pour de la classification de données de pièges photographiques et le ResNet-50 n'a pas pu être testé convenablement. Il nécessite plus de mémoire de calcul et celle-ci n'était pas disponible. De plus, les modèles plus profonds possèdent plus d'hyperparamètres à ajuster que les autres, ils sont donc plus vite sujets à du surapprentissage. Le jeu de données de taille restreinte et peu diversifié pourrait avoir favorisé ce surapprentissage.

À l'échelle de l'image, le modèle ResNet2D 1 a été choisi pour la Classification multi-espèces et le modèle ResNet2D 2 a été choisi pour la CB. Le modèle ResNet2D 2+3 n'a pas atteint de bons résultats. C'est contraire aux tests réalisés par Norouzzadeh *et al.* (2018) et Willi *et al.* (2019). Une nouvelle fois, l'utilisation d'un petit jeu de données d'entraînement peu diversifié pourrait en être la cause. Le fait de mettre de côté les images vides a fait passer la taille de ce jeu de données de 50140 à 32173 images. Le modèle ResNet2D 1 a du mal à différencier les animaux de petites tailles des images vides. Les exactitudes des classes de petits animaux ont donc augmenté pour le modèle ResNet2D 3 qui ne considère pas les images vides. Néanmoins, les autres classes ont subi une baisse de leur rappel avec ce dernier modèle. Il semble que le modèle apprend mieux quand il peut voir les mêmes images avec et sans l'animal. Si les images vides sont retirées du jeu de données de validation, le modèle ResNet2D 1 obtient une exactitude globale de 56,18 % alors qu'elle est de 39,37 % pour le ResNet2D 3.

Au sujet des méthodes qui permettent de passer d'un ensemble de prédictions pour les images d'une vidéo à une prédiction finale, ce sont les méthodes « moyenne x fréquence » (top1) et « addition de probabilités » (top3) qui ont été choisies pour la CM et la méthode « classique avec un seuil » pour la CB. D'autres méthodes pourraient être testées mais la plupart du temps si la prédiction finale est fausse, c'est parce que le modèle a mal classé toutes les images de la vidéo. Il serait donc plus judicieux d'essayer d'améliorer les performances des modèles à la place.

Une augmentation du nombre d'images extraites par vidéo provoque un accroissement du nombre de prédictions par vidéo. Cela peut être bénéfique pour les vidéos dans lesquelles l'animal n'est présent qu'un court instant, car plus d'images de l'animal seront extraites et participeront à la prise de décision. Les tableaux 17 et 18 démontrent qu'une augmentation du nombre d'images échantillonnées par vidéo induit une légère augmentation de l'exactitude. Le temps de traitement se voit aussi être augmenté fortement. Il faut trouver un compromis entre l'augmentation du temps de traitement et l'augmentation de l'exactitude. Cependant, le temps de traitement n'est pas le facteur le plus important car c'est l'ordinateur qui effectue la tâche. L'humain bénéficie quand même d'un gain de temps car il ne doit plus en prendre pour analyser manuellement les données.

Le modèle ResNet2D 1 a été favorisé pour la CM. Les exactitudes globales top 1 et top3 de ce modèle sont plus grandes que ses exactitudes moyennes top1 et top3 (table 19). Cela veut dire qu'il classe mieux les espèces qui sont plus représentées dans le jeu de données. Ceci confirme la constatation faite par Goodfellow *et al.* (2016) et Tabak *et al.* (2019). Comme dit précédemment, le modèle classe souvent les athérures, les écureuils, les oiseaux et les rats géants/souris dans la classe « No_sp ». Cela est compréhensible car ce sont les espèces les plus petites. Elles sont représentées par peu de pixels d'animaux sur l'image, par rapport aux autres espèces. Le modèle a également des difficultés avec les classes qui se ressemblent fortement.

C'est le cas des gorilles et des chimpanzés (figure 16a et 16b). Il apparaît que cette confusion intervient même si ces singes sont entièrement visibles sur la vidéo. Le modèle a également tendance à avoir plus de confusions entre certaines espèces qui sont actives la nuit (athérures, buffles de forêt, céphalophes à dos jaune, civettes/gennettes/nandinies, rats géant/souris et potamochères) et qui apparaissent sur les vidéos infrarouges. La raison est, par hypothèse, que le modèle ne peut pas profiter d'une large gamme de couleurs pour prendre sa décision.

Selon les tables 20 et 5, il y a une relation manifeste entre le rappel d'une classe et le nombre de caméras et donc d'arrière-plans différents disponibles par classe. Les espèces qui ont été capturées par plus de caméras possèdent presque systématiquement de meilleurs rappels. Lorsqu'une classe possède peu de diversité d'arrière-plans, le modèle ne parvient pas à généraliser l'information et n'arrive pas à classer correctement une vidéo quand l'arrière-plan change. Cela prouve, encore une fois, qu'il est difficile de développer des modèles qui sont performants sur des données qui possèdent des arrière-plans qui n'ont jamais été vus auparavant. Cela démontre également l'utilité de maximiser l'hétérogénéité et donc le nombre de caméras par classe dans le jeu de données et de ne pas mettre les données d'une même caméra dans des jeux de données différents. La même chose est observée avec le nombre d'images disponibles par classe dans le jeu de données (table 6). En effet, le modèle a atteint des moins bons résultats pour les classes qui étaient moins représentées dans le jeu de données d'entraînement.

Des rappels, précisions ou scores F1 assez élevés ont été atteints pour certaines classes (table 20). Toutefois, bien que ces résultats soient encourageants pour l'avenir, ce modèle ne pourrait malheureusement pas être utilisé afin d'automatiser entièrement l'analyse de données de PP, ou partiellement grâce aux prédictions top3. Ce modèle fait trop d'erreurs et possède des performances trop basses pour certaines espèces. Néanmoins, le modèle pourrait éventuellement être utile afin d'isoler les vidéos des classes qui possèdent les meilleures performances. Par exemple, les éléphants de forêt, les chevrotains aquatiques et les oiseaux ont des scores F1 assez élevés. L'utilisation d'un seuil de certitude comme l'ont fait Gomez Villa *et al.* (2016), Willi *et al.* (2019), Whytock *et al.* (2021) et Yang *et al.* (2021) pourrait également permettre d'améliorer les performances du modèle et d'élargir son domaine d'utilisation.

4.2.2 Classification binaire

Pour la classification binaire, c'est le modèle ResNet2D 2 qui a été retenu. De nouveau, l'exactitude globale est plus importante que l'exactitude moyenne. Cela veut dire que le modèle classe mieux les vidéos de la classe qui est la plus représentée dans le jeu de données. Dans ce cas-ci, c'est la classe « Animal ».

Ce genre de modèle peut être utile afin d'éliminer, au préalable, les vidéos vides de la base de données à analyser. Par conséquent, il ne restera plus qu'à classifier les vidéos d'animaux. Pour qu'un modèle de ce genre soit vraiment utilisable, il faut un rappel élevé pour la classe « Animal » de manière à ce que le modèle ne rate qu'un minimum de vidéos d'animaux. Il est aussi nécessaire que la précision de cette classe ne soit pas trop basse pour s'assurer que toutes les vidéos vides n'aient pas été classées dans les animaux. Le modèle ResNet2D 2 possède une haute précision pour la classe « Animal ». Néanmoins, le rappel de cette classe n'est que de 85,73 %. Cela veut dire que sur 100 images d'animaux quasiment 15 images seront perdues. Ce modèle n'est donc probablement pas directement utilisable. Mais peut-être que l'utilisation d'un seuil de certitude pourrait changer la donne.

Après l'étude des vidéos mal classées, il ressort que les modèles ResNet2D 1 et surtout ResNet2D 2 ont bien été perturbés par les difficultés citées au point 2.6 de la page 30. De fait, les vidéos mal classées concernent en grande partie les petits animaux, ceux dont seule une partie est visible, ceux qui sont cachés par la végétation ou ceux qui sont dans l'obscurité, pour la CB. Il a également été remarqué que des erreurs avaient été faites lors de l'identification des espèces sur les données. Cela a sûrement perturbé un minimum les modèles entraînés lors de ce travail.

4.3 Comparaison avec différents outils

Le modèle de classification multi-espèces développés dans le cadre de ce travail (ResNet2D 1) a obtenu une exactitude globale plus élevée que les autres modèles disponible en libres d'accès. Au-delà des performances mesurées plus basses, Mbaza AI et Zamba Cloud présentent plus de confusions entre les espèces. Ces résultats moins bons sont sûrement dûs au fait que ces outils ont été entraînés avec des données différentes en terme de caractéristiques (taille de l'image, résolution,...) ou en terme d'espèces et de paysages. Cela montre une nouvelle fois la complexité de faire des prédictions correctes sur des vidéos de PP qui ont été prises dans des endroits différents du jeu de données d'entraînement. Cependant, la CB ne semble pas être autant impactée par cette problématique. En effet, les outils testés ont mieux performé que le modèle ResNet2D 2.

Pour la CM, le modèle Mbaza AI attribue par erreur de nombreuses vidéos RGB dans les classes de céphalophes rouges et de céphalophes bleus, et de nombreuses vidéos infrarouges dans la classe des rats géants/souris. Le modèle Zamba Cloud attribue de manière erronée de nombreuses vidéos dans la classe des mangoustes. Et le modèle ResNet2D 1 classe lui par erreur de nombreuses vidéos dans la classe « No_sp » (figure 14). Ces classes qui sont plus souvent prédites par les modèles étaient sûrement dominantes dans les jeux de données d'entraînement. C'est d'ailleurs le cas pour le modèle ResNet2D 1 et la classe « No_sp ». Les modèles ont tous des difficultés pour différencier les gorilles des chimpanzés car ils se ressemblent fort (figure 16a et 16b). Le modèle Mbaza AI a également du mal avec les mandrills et les cercopithecidae (figure 16d et 16c).

(a) Gorille **(b)** Chimpanzés

(c) Cercopithecidae **(d)** Mandrill

Figure 16 – Images extraites d'une vidéo de gorille (a), de chimpanzé (b), de cercopithecidae (c) et de mandrill (d), capturée par un piège photographique.

Zamba Cloud est le modèle qui performe le moins bien sur le jeu de données utilisé malgré la réduction du nombre de classes, spécifique à ce modèle. Seulement 27 000 vidéos ont été utilisées pour entraîner ce modèle. Ce petit jeu de données doit être la cause de ces plus faibles performances.

À propos de la CB, l'outil Wildlife Insights possède le meilleur score F1 pour la classe « Animal » mais le modèle Mbaza possède la plus grande exactitude et haute précision pour cette même classe. Le modèle

Mbaza pourrait donc probablement être utilisé afin de faciliter l'analyse des données de PP. Un seuil de certitude pourrait sûrement minimiser le nombre d'animaux mal classés.

5 Recommandations et perspectives

Dans le futur, il est fort probable que l'homme utilisera l'apprentissage profond afin de l'aider à analyser les données capturées par des pièges photographiques. En ce qui concerne la reconnaissance d'espèces animales, l'automatisation pourrait être complète ou partielle. Il y a plusieurs possibilités. Un modèle pourrait classer les données de PP avec la même exactitude voire mieux, qu'un homme. Un seuil de certitude pourrait être utilisé afin de réduire le nombre de données qui doivent être analysées par l'homme. Un modèle pourrait également éliminer les données vides. Enfin, d'autres modèles pourraient identifier les 3 ou 5 classes (top3 ou top5) les plus probables dans le but de réduire le nombre de possibilités/choix lors l'identification de données de PP.

D'après les expériences réalisées dans ce travail, plusieurs conseils ont été établis pour entraîner des modèles afin de traiter des données de PP. Il faut maximiser la taille du jeu de données et son hétérogénéité afin de maximiser les performances d'un modèle d'apprentissage profond qui veut automatiser la reconnaissance d'espèces animales. Si ce modèle est destiné à traiter des données qui possèdent un arrière-plan différent de ceux des données d'entraînement, il est important de faire attention à 2 éléments. Premièrement, les vidéos d'une même caméra doivent se retrouver dans le même jeu de données. Ensuite, il est nécessaire d'utiliser les vidéos d'un maximum de caméras par espèce. Il est également primordial d'annoter correctement les données afin de ne pas perturber le modèle.

Différentes méthodes, ainsi que différentes architectures de réseaux de neurones profonds pourraient être utilisées. La détection d'objets et la classification d'images ou de vidéos semblent être les approches les plus prometteuses.

Suite à ce travail, il n'est pas recommandé d'utiliser des réseaux de neurones récurrents dans le but de classifier des vidéos de PP. Au contraire, il pourrait être intéressant d'approfondir l'approche basée sur la classification à l'échelle de l'image, notamment en appliquant les recommandations énoncées 2 paragraphes plus haut. Il pourrait aussi être possible de mettre en évidence les images vides grâce à des méthodes moins complexes que l'apprentissage profond comme Chen *et al.* (2019) l'ont fait avec le calcul de leur indice d'énergie. Une autre option à explorer sont les réseaux de neurones convolutifs à trois dimensions comme ceux qui ont commencé à être expérimentés dans l'annexe 2 ou comme l'architecture SlowFast (Feichtenhofer *et al.*, 2019) utilisée pour le modèle Zamba Cloud. Ceux-ci nécessitent cependant une mémoire de calcul importante et donc peut-être l'utilisation d'un super-ordinateur. Enfin, une approche basée sur la détection d'objets devrait également être étudiée. En détection d'objets, une boîte est tracée autour de chaque animal et le contenu de la boîte est classifié. Cela veut dire que moins de pixels d'arrière-plan et donc de bruit ne seront pris en considération lors de la prédiction. Cela pourrait éventuellement faciliter la classification de petits animaux et d'espèces ressemblantes. D'un côté, cette approche va demander un effort important d'annotation. D'un autre côté, il serait peut-être possible d'utiliser un modèle déjà entraîné afin de détecter les animaux et tracer des boites autour de ceux-ci (Norouzzadeh *et al.*, 2021).

Enfin, l'étape suivante est de développer des modèles qui sont capables de prédire plusieurs classes par vidéo (Yang *et al.*, 2016; Delplanque *et al.*, 2022) ou d'y compter le nombre d'animaux présents (Loos *et al.*, 2018) ou encore de prédire le comportement des animaux (Norouzzadeh *et al.*, 2018; Schindler et Steinhage, 2021). Les 2 premières options privilégieraient certainement de la détection d'objets alors que la troisième option serait probablement réalisée grâce à de la classification. Néanmoins, la détection d'objets nécessite un travail d'annotation plus important en amont.

6 Contribution personnelle de l'étudiant

Dans ce travail, la possibilité d'automatiser la reconnaissance d'espèces animales dans des vidéos capturées par des pièges photographiques installés dans des forêts tropicales d'Afrique centrale, a été étudiée. Ma contribution personnelle dans ce travail se résume à :

— La standardisation des données ;

— La définition des classes pour la classification multi-espèces, en collaboration avec un représentant du DFT de GxABT et du Cirad ;

— Le choix des architectures utilisées ;

— L'entraînement des différents modèles ;

— Le développement de méthodes qui permettent la conversion des prédictions faites sur les images échantillonnées dans une vidéo, en une prédiction finale pour cette dernière ;

— Le test de différents outils qui ont été développés dans le but de classifier des données de pièges photographiques ;

— Le calcul des résultats ;

— L'interprétation des résultats ;

— La rédaction de ce travail.

Ce travail pourrait servir de point de départ de nouvelles études afin d'avancer plus en profondeur sur le sujet.

7 Conclusion

Ce travail présentait pour objectifs de développer un modèle d'apprentissage profond dans le but de faire de la classification multi-espèces et de la classification binaire sur des vidéos prises par des pièges photographiques placés dans les forêts tropicales d'Afrique centrale. Un dernier objectif était la comparaison des modèles développés avec différents outils existants (Mbaza AI, Wildlife Insights et Zamba Cloud).

Plusieurs architectures de modèles composées de réseaux de neurones convolutifs et de réseaux de neurones récurrents, ont été abordées. Les modèles qui se sont montrés les plus performants sont des classificateurs d'images constitués d'un ResNet-18 à 2 dimensions. La prédiction à l'échelle de la vidéo est déterminée grâce à une méthode empirique appliquée sur les prédictions faites sur les images extraites de la vidéo. Les résultats sont prometteurs mais ne permettent pas d'automatiser entièrement la reconnaissance d'espèces animales sur des vidéos de PP car les modèles commettent encore trop d'erreurs. Par contre, ils pourraient être utilisés dans certains cas spécifiques comme par exemple, la classification d'espèces animales qui ont atteint de hautes performances. L'utilisation d'un seuil de certitude pourrait encore augmenter les résultats. Les vidéos dont la prédiction est faite avec une certitude inférieure à ce seuil devront être classifiées par l'homme, alors que les autres prédictions seront acceptées.

La taille de l'image, la taille du *batch* et le nombre d'images dans les jeux de données d'entraînement ont été les paramètres qui ont le plus influencé la qualité des modèles. Une augmentation du nombre d'images extraites par seconde d'une vidéo va provoquer une légère amélioration des résultats des modèles mais va fortement impacter le temps de traitement des vidéos. Il faut trouver le bon compromis entre ces 2 facteurs.

Les modèles développés classent les espèces les mieux représentées dans le jeu de données d'entraînement, avec une plus grande certitude. En contrepartie, plus de vidéos ont tendance à être classifiées de manière erronée dans ces espèces (baisse de la précision de la classe). Les modèles ont également du mal avec les espèces qui se ressemblent fortement. Les vidéos infrarouges peuvent aussi être plus compliquées à classer correctement pour les modèles. Le nombre de caméras différentes par espèce qui ont capturées des vidéos d'animaux a un impact sur les performances par classe du modèle. Un faible nombre de caméras pour une classe veut dire qu'il y a peu de variabilité de l'arrière-plan. Le modèle a alors du mal généraliser l'information pour cette classe et impacte défavorablement les métriques de cette dernière. En effet, le modèle ne parvient pas à classer correctement une vidéo de cette classe quand l'arrière-plan de celle-ci est différent de ceux des données utilisées pour développer le modèle. C'est un problème récurrent qui apparaît lors de l'utilisation de l'apprentissage profond pour traiter des données de PP (Beery *et al.*, 2019; Schneider *et al.*, 2020).

Le modèle développé pour la CM a atteint une exactitude globale de 67,93 % et une exactitude moyenne de 61,75 % sur le jeu de données de test. Pour la CB, le modèle possède une exactitude globale de 84,89 % et une exactitude moyenne de 77,58 %.

En ce qui concerne le dernier objectif, les outils testés se sont montrés moins performants pour la CM, que les modèles développés dans ce travail. Ces outils présentent également plus de confusions entre les classes. Au contraire, ces derniers se sont montrés plus performants pour la CB.

Des recommandations et des pistes d'explorations pour de futurs travaux ont également été rédigées. Il est fort probable que l'apprentissage profond sera utilisé à l'avenir afin d'analyser ou aider à l'analyse de données de PP. La classification et la détection d'objets sont les 2 options les plus envisageables à l'heure actuelle. Cependant, l'expertise des scientifiques sera toujours utile, soit pour valider les modèles avant leur utilisation, soit pour réaliser des tâches plus complexes comme de l'analyse comportementale.

Références

1. Almond, R. E., Grooten, M. et Peterson, T. (2020). *Living Planet Report 2020-Bending the curve of biodiversity loss*. World Wildlife Fund.

2. Apps, P. J. et McNutt, J. W. (2018). How camera traps work and how to work them. *African Journal of Ecology*, 56(4):702–709.

3. Assou, D., D'Cruze, N., Kirkland, H., Auliya, M., Macdonald, D. W. et Segniagbeto, G. H. (2021). Camera trap survey of mammals in the Fazao-Malfakassa National Park, Togo, West Africa. *African Journal of Ecology*, 59(3):583–596.

4. Babweteera, F. et Brown, N. (2010). Spatial patterns of tree recruitment in East African tropical forests that have lost their vertebrate seed dispersers. *Journal of Tropical Ecology*, 26(2):193–203.

5. Beery, S., Morris, D., Yang, S., Simon, M., Norouzzadeh, A. et Joshi, N. (2019). Efficient Pipeline for Automating Species ID in new Camera Trap Projects. *Biodiversity Information Science and Standards*, 3:e37222.

6. Beery, S., Van Horn, G. et Perona, P. (2018). Recognition in Terra Incognita. *In Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473.

7. Beery, S., Wu, G., Rathod, V., Votel, R. et Huang, J. (2020). Context R-CNN : Long Term Temporal Context for Per-Camera Object Detection. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13075–13085.

8. Blake, J. G., Guerra, J., Mosquera, D., Torres, R., Loiselle, B. A. et Romo, D. (2010). Use of Mineral Licks by White-Bellied Spider Monkeys (Ateles belzebuth) and Red Howler Monkeys (Alouatta seniculus) in Eastern Ecuador. *International Journal of Primatology*, 31(3):471–483.

9. Campos, R. C., Steiner, J. et Zillikens, A. (2012). Bird and mammal frugivores of Euterpe edulis at Santa Catarina island monitored by camera traps. *Studies on Neotropical Fauna and Environment*, 47(2):105–110.

10. Carl, C., Schönfeld, F., Profft, I., Klamm, A. et Landgraf, D. (2020). Automated detection of European wild mammal species in camera trap images with an existing and pre-trained computer vision model. *European Journal of Wildlife Research*, 66(4):62.

11. Carreira, J. et Zisserman, A. (2017). Quo Vadis, Action Recognition ? A New Model and the Kinetics Dataset. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308.

12. Chen, G., Han, T. X., He, Z., Kays, R. et Forrester, T. (2014). Deep convolutional neural network based species recognition for wild animal monitoring. *In 2014 IEEE International Conference on Image Processing (ICIP)*, pages 858–862.

13. Chen, R., Little, R., Mihaylova, L., Delahay, R. et Cox, R. (2019). Wildlife surveillance using deep learning methods. *Ecology and Evolution*, 9(17):9453–9466.

14. Cho, K., Merrienboer, B. v., Gülçehre, , Bougares, F., Schwenk, H. et Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR*, abs/1406.1078.

15. Chollet, F. (2021). *Deep Learning with Python, Second Edition*. Simon and Schuster.

16. de Wasseige, C., Hiol Hiol, F., Mayaux, P., Eba'a Atyi, R., Desclée, B., Defourny, P., Nasi, R., de Marcken, P., Billand, A. et Bayol, N. (2012). *Les forêts du bassin du Congo : état des forêts 2010*. Publications Office of the European Union.

17. Delplanque, A., Foucher, S., Lejeune, P., Linchant, J. et Théau, J. (2022). Multispecies detection and identification of African mammals in aerial imagery using convolutional neural networks. *Remote Sensing in Ecology and Conservation*, 8(2):166–179.

18. Diba, A., Fayyaz, M., Sharma, V., Karami, A. H., Arzani, M. M., Yousefzadeh, R. et Gool, L. V. (2017). Temporal 3D ConvNets : New Architecture and Transfer Learning for Video Classification. *CoRR*, abs/1711.08200.

19. Elgendy, M. (2020). *Deep Learning for Vision Systems*. Simon and Schuster.

20. Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science*, 14(2):179–211.

21. Feichtenhofer, C., Fan, H., Malik, J. et He, K. (2019). SlowFast Networks for Video Recognition. *In Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211.

22. Feichtenhofer, C., Pinz, A. et Zisserman, A. (2016). Convolutional Two-Stream Network Fusion for Video Action Recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941.

23. Garrote, G., Gil-Sánchez, J. M., McCain, E. B., de Lillo, S., Tellería, J. L. et Simón, M. (2012). The effect of attractant lures in camera trapping : a case study of population estimates for the Iberian lynx (Lynx pardinus). *European Journal of Wildlife Research*, 58(5):881–884.

24. Gil-Sánchez, J. M., Moral, M., Bueno, J., Rodríguez-Siles, J., Lillo, S., Pérez, J., Martín, J. M., Valenzuela, G., Garrote, G., Torralba, B. et Simón-Mata, M. (2011). The use of camera trapping for estimating Iberian lynx (Lynx pardinus) home ranges. *European Journal of Wildlife Research*, 57(6):1203–1211.

25. Gomez Villa, A., Diez, G., Salazar, A. et Diaz, A. (2016). Animal identification in low quality camera-trap images using very deep convolutional neural networks and confidence thresholds. *In International symposium on visual computing*, pages 747–756. Springer.

26. Gomez Villa, A., Salazar, A. et Vargas, F. (2017). Towards automatic wild animal monitoring : Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics*, 41:24–32.

27. Goodfellow, I., Bengio, Y. et Courville, A. (2016). *Deep learning*. MIT press.

28. Gray, T. et Channa, P. (2011). Habitat preferences and activity patterns of the larger mammal community in Phnom Prich Wildlife Sanctuary, Cambodia. *The Raffles Bulletin of Zoology*, 59:311–318.

29. He, K., Zhang, X., Ren, S. et Sun, J. (2016). Deep Residual Learning for Image Recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

30. Hochreiter, S. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116.

31. Hochreiter, S. et Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

32. Huang, G., Liu, Z., van der Maaten, L. et Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708.

33. Hénaut, Y. et Charruau, P. (2012). Nest attendance and hatchling care in wild American crocodiles (Crocodylus acutus) in Quintana Roo, Mexico. *Animal Biology*, 62(1):29–51.

34. Kalle, R., Ramesh, T., Qureshi, Q. et Sankar, K. (2011). Density of tiger and leopard in a tropical deciduous forest of Mudumalai Tiger Reserve, southern India, as estimated using photographic capture–recapture sampling. *Acta Theriologica*, 56(4):335–342.

35. Koike, S., Morimoto, H., Kozakai, C., Arimoto, I., Yamazaki, K., Iwaoka, M., Soga, M. et Koganezawa, M. (2012). Seed removal and survival in Asiatic black bear Ursus thibetanus faeces : effect of rodents as secondary seed dispersers. *Wildlife Biology*, 18(1):24–34.

36. Krizhevsky, A., Sutskever, I. et Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *In Advances in Neural Information Processing Systems*, pages 1106–1114.

37. Kutugata, M., Baumgardt, J., Goolsby, J. A. et Racelis, A. E. (2021). Automatic Camera-Trap Classification Using Wildlife-Specific Deep Learning in Nilgai Management. *Journal of Fish and Wildlife Management*, 12(2):412–421.

38. LeCun, Y., Bengio, Y. et Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

39. LeCun, Y., Bottou, L., Bengio, Y. et Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

40. Li, W., Wang, L., Li, W., Agustsson, E. et Gool, L. V. (2017). WebVision Database : Visual Learning and Understanding from Web Data. *CoRR*, abs/1708.02862.

41. Liu, X., Wu, P., Songer, M., Cai, Q., He, X., Zhu, Y. et Shao, X. (2013). Monitoring wildlife abundance and diversity with infra-red camera traps in Guanyinshan Nature Reserve of Shaanxi Province, China. *Ecological Indicators*, 33:121–128.

42. Loos, A., Weigel, C. et Koehler, M. (2018). Towards Automatic Detection of Animals in Camera-Trap Images. *In 2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1805–1809.

43. Lopucki, R. (2007). Social relationship in a bank vole Clethrionomys glareolus (Schreber, 1780) population : video monitoring under field conditions. *Polish Journal of Ecology*, 55(3):543–558.

44. Luo, G., Yang, C., Zhou, H., Seitz, M., Wu, Y. et Ran, J. (2019). Habitat use and diel activity pattern of the Tibetan Snowcock (Tetraogallus tibetanus) : a case study using camera traps for surveying high-elevation bird species. *Avian Research*, 10(1):1–9.

45. Meek, P. D., Ballard, G., Claridge, A., Kays, R., Moseby, K., O'Brien, T., O'Connell, A., Sanderson, J., Swann, D. E., Tobler, M. et Townsend, S. (2014). Recommended guiding principles for reporting on camera trapping research. *Biodiversity and Conservation*, 23(9):2321–2343.

46. Moruzzi, T. L., Fuller, T. K., DeGraaf, R. M., Brooks, R. T. et Li, W. (2002). Assessing Remotely Triggered Cameras for Surveying Carnivore Distribution. *Wildlife Society Bulletin (1973-2006)*, 30(2):380–386.

47. Mu, R. et Zeng, X. (2019). A Review of Deep Learning Research. *KSII Transactions on Internet and Information Systems (TIIS)*, 13(4):1738–1764.

48. Nguyen, H., Maclagan, S. J., Nguyen, T. D., Nguyen, T., Flemons, P., Andrews, K., Ritchie, E. G. et Phung, D. (2017). Animal Recognition and Identification with Deep Convolutional Neural Networks for Automated Wildlife Monitoring. *In 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 40–49.

49. Norouzzadeh, M. S., Morris, D., Beery, S., Joshi, N., Jojic, N. et Clune, J. (2021). A deep active learning system for species identification and counting in camera trap images. *Methods in Ecology and Evolution*, 12(1):150–161.

50. Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C. et Clune, J. (2018). 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.

51. Nyiramana, A., Mendoza, I., Kaplin, B. A. et Forget, P.-M. (2011). Evidence for Seed Dispersal by Rodents in Tropical Montane Forest in Africa. *Biotropica*, 43(6):654–657.

52. Oliveira-Santos, L. G. R., Graipel, M. E., Tortato, M. A., Zucco, C. A., Cáceres, N. C. et Goulart, F. V. B. (2012). Abundance changes and activity flexibility of the oncilla, Leopardus tigrinus (Carnivora : Felidae), appear to reflect avoidance of conflict. *Zoologia (Curitiba)*, 29:115–120.

53. O'Connell, A. F., Nichols, J. D. et Karanth, K. U., éditeurs (2011). *Camera Traps in Animal Ecology*. Springer Japan, Tokyo.

54. O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G. V., Krpalkova, L., Riordan, D. et Walsh, J. (2020). Deep Learning vs. Traditional Computer Vision. *In* Arai, K. et Kapoor, S., éditeurs : *Advances in Computer Vision*, Advances in Intelligent Systems and Computing, pages 128–144.

55. Pak, M. et Kim, S. (2017). A review of deep learning in image recognition. *In 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT)*, pages 1–3.

56. Pender, R. J., Shiels, A. B., Bialic-Murphy, L. et Mosher, S. M. (2013). Large-scale rodent control reduces pre- and post-dispersal seed predation of the endangered Hawaiian lobeliad, Cyanea superba subsp. superba (Campanulaceae). *Biological Invasions*, 15(1):213–223.

57. Petso, T., Jamisola, R. S. et Mpoeleng, D. (2021). Review on methods used for wildlife species and individual identification. *European Journal of Wildlife Research*, 68(1):1–18.

58. Rehman, A. et Belhaouari, S. B. (2021). Deep Learning for Video Classification : A Review. *TechRxiv*.

59. Schindler, F. et Steinhage, V. (2021). Identification of animals and recognition of their actions in wildlife videos using deep learning techniques. *Ecological Informatics*, 61:101215.

60. Schneider, S., Greenberg, S., Taylor, G. W. et Kremer, S. C. (2020). Three critical factors affecting automated image species recognition performance for camera traps. *Ecology and Evolution*, 10(7):3503–3517.

61. Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k. et WOO, W.-c. (2015). Convolutional LSTM Network : A Machine Learning Approach for Precipitation Nowcasting. *In Advances in Neural Information Processing Systems*, volume 28.

62. Simonyan, K. et Zisserman, A. (2014). Two-Stream Convolutional Networks for Action Recognition in Videos. *In Advances in Neural Information Processing Systems*, volume 27.

63. Soley, F. G. et Alvarado-Díaz, I. (2011). Prospective thinking in a mustelid ? Eira barbara (Carnivora) cache unripe fruits to consume them once ripened. *Naturwissenschaften*, 98(8):693–698.

64. Sollmann, R. (2018). A gentle introduction to camera-trap data analysis. *African Journal of Ecology*, 56(4):740–749.

65. Srbek-Araujo, A. C., Silveira, L. F. et Chiarello, A. G. (2012). The Red-Billed Curassow (Crax blumenbachii) : Social Organization, and Daily Activity Patterns. *The Wilson Journal of Ornithology*, 124(2):321–327.

66. Sudhakaran, S. et Lanz, O. (2017a). Convolutional Long Short-Term Memory Networks for Recognizing First Person Interactions. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2339–2346.

67. Sudhakaran, S. et Lanz, O. (2017b). Learning to detect violent videos using convolutional long short-term memory. *In 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6.

68. Szegedy, C., Ioffe, S., Vanhoucke, V. et Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. *In Thirty-first AAAI conference on artificial intelligence*.

69. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. et Rabinovich, A. (2015). Going Deeper With Convolutions. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

70. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. et Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826.

71. Tabak, M. A., Norouzzadeh, M. S., Wolfson, D. W., Sweeney, S. J., Vercauteren, K. C., Snow, N. P., Halseth, J. M., Di Salvo, P. A., Lewis, J. S., White, M. D., Teton, B., Beasley, J. C., Schlichting, P. E., Boughton, R. K., Wight, B., Newkirk, E. S., Ivan, J. S., Odell, E. A., Brook, R. K., Lukacs, P. M., Moeller, A. K., Mandeville, E. G., Clune, J. et Miller, R. S. (2019). Machine learning to classify animal species in camera trap images : Applications in ecology. *Methods in Ecology and Evolution*, 10(4):585–590.

72. Tran, D., Bourdev, L., Fergus, R., Torresani, L. et Paluri, M. (2015). Learning Spatiotemporal Features With 3D Convolutional Networks. *In Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497.

73. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y. et Paluri, M. (2018). A Closer Look at Spatiotemporal Convolutions for Action Recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6450–6459.

74. Trolliet, F., Huynen, M.-C., Vermeulen, C. et Hambuckers, A. (2014). Use of camera traps for wildlife studies. A review. *Biotechnol. Agron. Soc. Environ.*, 18(3):446–454.

75. Vecvanags, A., Aktas, K., Pavlovs, I., Avots, E., Filipovs, J., Brauns, A., Done, G., Jakovels, D. et Anbarjafari, G. (2022). Ungulate Detection and Species Classification from Camera Trap Images Using RetinaNet and Faster R-CNN. *Entropy*, 24(3):353.

76. Wearn, O. R. et Glover-Kapfer, P. (2017). *Camera trapping for conservation : a guide to best-practices*. World Wildlife Fund.

77. Weinstein, B. G. (2018). Scene-specific convolutional neural networks for video-based biodiversity detection. *Methods in Ecology and Evolution*, 9(6):1435–1441.

78. Whytock, R. C., Świeżewski, J., Zwerts, J. A., Bara-Słupski, T., Koumba Pambo, A. F., Rogala, M., Bahaa-el din, L., Boekee, K., Brittain, S., Cardoso, A. W., Henschel, P., Lehmann, D., Momboua, B., Kiebou Opepa, C., Orbell, C., Pitman, R. T., Robinson, H. S. et Abernethy, K. A. (2021). Robust ecological analysis of camera trap data labelled by a machine learning model. *Methods in Ecology and Evolution*, 12(6):1080–1092.

79. Willi, M., Pitman, R. T., Cardoso, A. W., Locke, C., Swanson, A., Boyer, A., Veldthuis, M. et Fortson, L. (2019). Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*, 10(1):80–91.

80. Wilson, E. O., éditeur (1988). *Biodiversity*. The National Academies Press, Washington, DC.

81. Wu, H., Liu, Q. et Liu, X. (2019). A Review on Deep Learning Approaches to Image Classification and Object Segmentation. *Computers, Materials & Continua*, 60(2):575–597.

82. Wu, Z., Wang, X., Jiang, Y.-G., Ye, H. et Xue, X. (2015). Modeling Spatial-Temporal Clues in a Hybrid Deep Learning Framework for Video Classification. *In Proceedings of the 23rd ACM international conference on Multimedia*, pages 461–470.

83. Yang, D.-Q., Tan, K., Huang, Z.-P., Li, X.-W., Chen, B.-H., Ren, G.-P. et Xiao, W. (2021). An automatic method for removing empty camera trap images using ensemble learning. *Ecology and Evolution*, 11(12):7591–7601.

84. Yang, H., Tianyi Zhou, J., Zhang, Y., Gao, B.-B., Wu, J. et Cai, J. (2016). Exploit Bounding Box Annotations for Multi-Label Object Recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 280–288.

85. Yin, W., Kann, K., Yu, M. et Schütze, H. (2017). Comparative Study of CNN and RNN for Natural Language Processing. *CoRR*, abs/1702.01923.

86. Yousif, H., Yuan, J., Kays, R. et He, Z. (2017). Fast human-animal detection from highly cluttered camera-trap images using joint background modeling and deep learning classification. *In 2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4.

87. Zha, S., Luisier, F., Andrews, W., Srivastava, N. et Salakhutdinov, R. (2015). Exploiting Image-trained CNN Architectures for Unconstrained Video Classification. *CoRR*, abs/1503.04144.

88. Zhang, Z., He, Z., Cao, G. et Cao, W. (2016). Animal Detection From Highly Cluttered Natural Scenes Using Spatiotemporal Object Region Proposals and Patch Verification. *IEEE Transactions on Multimedia*, 18(10):2079–2092.

89. Zualkernan, I. A., Dhou, S., Judas, J., Sajun, A. R., Gomez, B. R., Hussain, L. A. et Sakhnini, D. (2020). Towards an IoT-based Deep Learning Architecture for Camera Trap Image Classification. *In 2020 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)*, pages 1–6.

Annexes

Annexe 1 : Implémentation des combinaisons d'augmentations de données testées

La première combinaison de transformations qui a servi à faire de l'augmentation de données est composée d'un changement de luminosité et/ou de contraste et d'un ajout de flou. Cette combinaison a été implémentée grâce aux lignes de codes suivantes :

```
trans = [ColorJitter(always_apply=False, p=0.25, brightness=[0.5, 1.5],
         contrast=[0.5, 1.5], saturation=[1.0, 1.0], hue=[0.0, 0.0]),
         Blur(always_apply=False, p=0.25, blur_limit=(3, 11)))]
```

La deuxième combinaison est constituée d'une modification de la luminosité et/ou du contraste, d'un ajout de flou et d'une distorsion. Cela a été implémenté grâce aux lignes de codes suivantes :

```
trans = [ColorJitter(always_apply=False, p=0.25, brightness=[0.5, 1.5],
         contrast=[0.5, 1.5], saturation=[1.0, 1.0], hue=[0.0, 0.0]),
         OpticalDistortion(always_apply=False, p=0.25,
         distort_limit=(-0.2, 0.2), shift_limit=(-0.2, 0.2),
         interpolation=1, border_mode=4, value=None, mask_value=None),
         Blur(always_apply=False, p=0.25, blur_limit=(3, 11)))]
```

Annexe 2 : ResNet à 3 dimensions

Des modèles ont été entraînés grâce à des architectures composées de réseaux de neurones convolutifs à 3 dimensions. Un modèle a été entraîné pour la classification multi-espèces : ResNet3D 1, et un autre pour la classification binaire : ResNet3D 2. Ceux-ci demandent une mémoire de calcul plus importante. Le choix des hyperparamètres des modèles a donc été restreint par la capacité de calcul de l'ordinateur utilisé. Ces modèles sont des ResNet-50 à 3 dimensions. ResNet-50 a été choisi car seulement 2 profondeurs de réseaux (50, 101) sont proposées dans la bibliothèque utilisée (pytorchvideo). Le perceptron multicouche final est composé d'une couche d'entrée avec 2048 neurones et du *dropout* et d'une couche de sortie avec un nombre de neurones égal au nombre de classes qui peuvent être prédites.

Les hyperparamètres utilisés sont une taille de *batch* d'une vidéo, un taux d'apprentissage de 0,01, une taille d'images de 225 pixels de hauteur et de 450 pixels de largeur, un *dropout* avec une probabilité de 0,5 et un nombre d'images échantillonnées par seconde égal à 1. L'entraînement de ces modèles a été effectué grâce aux mêmes optimiseurs, fonction de perte et planificateur que ceux qui ont été utilisés pour entraîner les autres modèles développés à travers ce travail.

Les poids du modèle étaient sauvegardés lorsque l'exactitude globale surpassait l'exactitude globale maximale pour la CM et lorsque l'exactitude moyenne dépassait l'exactitude moyenne maximale pour la CB.

Ces modèles ne permettent pas d'atteindre de meilleurs résultats que ceux du ResNet2D 1 et du ResNet2D 2. Cependant, les premiers essais sont encourageants. Contrairement, aux modèles entraînés grâce aux architectures ResNet2D + ConvLSTM et ResNet2D + LSTM, les modèles ResNet3D 1 et ResNet3D 2 s'entraînent et convergent vers une erreur minimale. Les performances globales des modèles ResNet3D 1 et ResNet3D 2 sur le jeu de données de validation sont reprises dans la table 30.

Table 29 – Performances globales des modèles ResNet3D 1 et ResNet3D 2 sur le jeu de vidéos de validation.

Classification	Métrique	ResNet3D 1	ResNet3D 2
multi-espèces	Exactitude globale top1	18,48 %	/
	Exactitude moyenne top1	13,18 %	/
	Exactitude globale top3	41,10 %	/
	Exactitude moyenne top3	33,03 %	/
Binaire	Exactitude globale	93,52 %	94,76 %
	Exactitude moyenne	49,34 %	50,00 %

Pour la CM, les performances globales du modèle ResNet3D 1 sur le jeu de données de test sont présentées dans la table 30, alors que ses performances par classe sont reprises dans le tableau 31. La figure 17 est la matrice de confusion obtenue grâce à l'application du modèle ResNet3D 1 sur le jeu de données de test.

Table 30 – Performances globales du modèle ResNet3D 1 sur le jeu de vidéos de test, pour la classification multi-espèces.

Modèle	Exactitude globale top1	Exactitude moyenne top1	Exactitude globale top3	Exactitude moyenne top3	Temps de traitement
ResNet3D 1	15,68 %	14,15 %	39,42 %	38,19 %	Divisé par 8

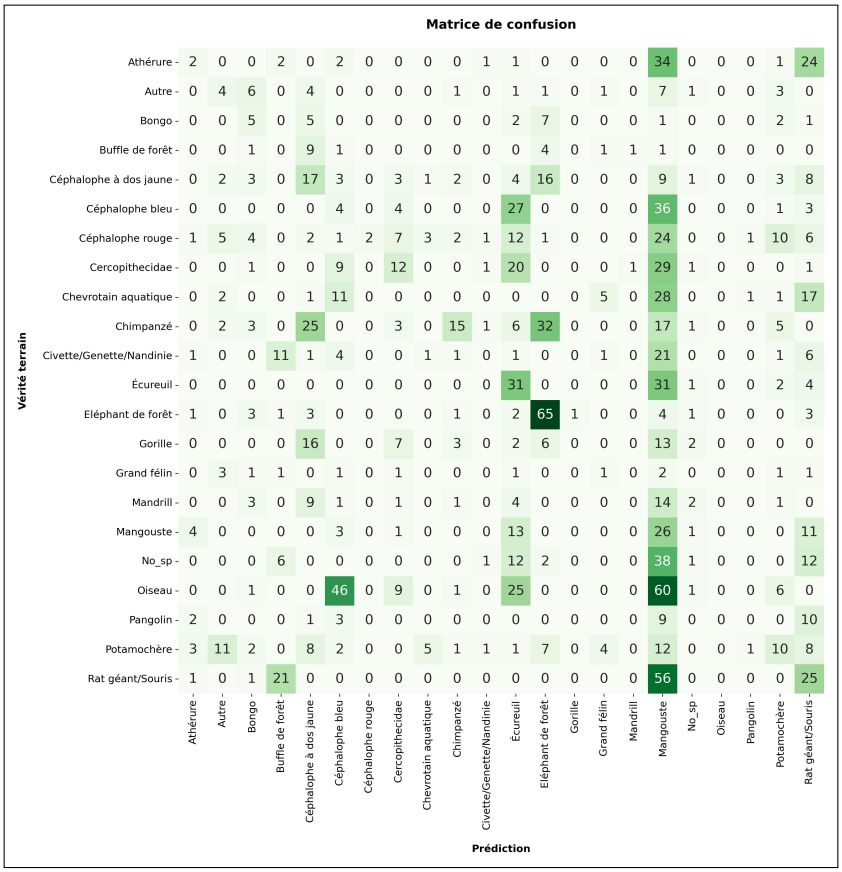


Figure 17 – Matrice de confusion obtenue grâce au modèle ResNet3D 1 appliqué sur le jeux de vidéos de test, pour la classification multi-espèces.

En ce qui concerne la CB, la table 32 reprend les performances globales du modèle ResNet3D 2 sur le jeu de données de test. La table 33 présente les performances par classe du modèle ResNet3D 2 sur le jeu de données de test. La figure 18 est la matrice de confusion obtenue grâce à l'application du modèle ResNet3D 2 sur le jeu de données de test.

Table 31 – Performances par classe du modèle ResNet3D 1 sur le jeu de vidéos de test, pour la classification multi-espèces.

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]	**Rappel top3** [%]
Athérure	2,99	13,33	4,88	77,61
Autre	13,79	13,79	13,79	44,83
Bongo	21,74	14,71	17,54	69,57
Buffle de forêt	0,00	0,00	0,00	0,00
Céphalophe à dos jaune	23,61	16,83	19,65	54,17
Céphalophe bleu	5,33	4,40	4,82	30,67
Céphalophe rouge	2,44	100,00	4,76	12,20
Cercopithecidae	16,00	25,00	19,51	48,00
Chevrotain aquatique	0,00	0,00	0,00	13,64
Chimpanzé	13,64	53,57	21,74	33,64
Civette/Genette/Nandinie	0,00	0,00	0,00	24,49
Écureuil	44,93	18,79	26,50	89,86
Eléphant de forêt	76,47	46,10	57,52	81,18
Gorille	0,00	0,00	0,00	0,00
Grand félin	7,69	7,69	7,69	23,08
Mandrill	0,00	0,00	0,00	0,00
Mangouste	44,07	5,51	9,79	86,44
No_sp	1,39	7,69	2,35	15,28
Oiseau	0,00	0,00	0,00	6,04
Pangolin	0,00	0,00	0,00	32,00
Potamochère	13,16	21,28	16,26	30,26
Rat géant/Souris	24,04	17,86	20,49	67,31

Table 32 – Performances globales du modèle ResNet3D 2 sur le jeu de vidéos de test, pour la classification binaire.

Modèle	**Exactitude globale**	**Exactitude moyenne**	**Time video**
ResNet3D 2	73,06 %	66,75 %	Divisé par 8

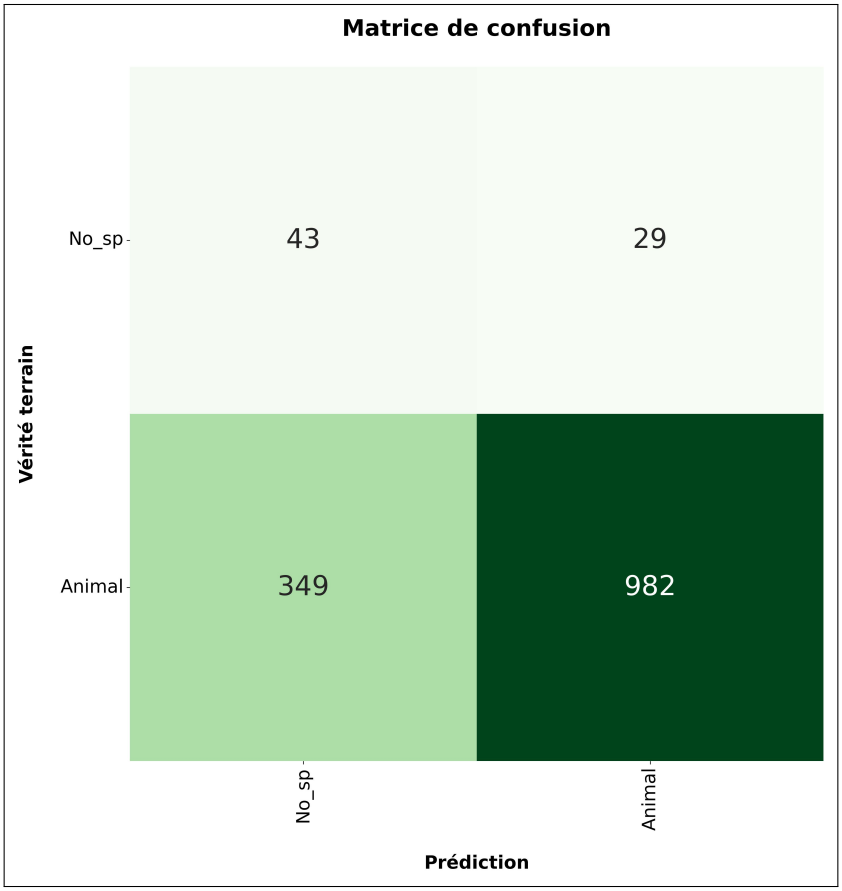


Figure 18 – Matrice de confusion obtenue grâce au modèle ResNet3D 2 appliqué sur le jeu de vidéos de test, pour la classification binaire.

Table 33 – Performances par classe du modèle ResNet3D 2 sur le jeu de vidéos de test, pour la classification binaire.

Classe	**Rappel** [%]	**Précision** [%]	**Score F1** [%]
No_sp	59,72	10,97	18,53
Animal	73,78	97,13	83,86

Il est probable que ces modèles puissent atteindre de meilleurs résultats et rivaliser avec les modèles ResNet2D 1 et ResNet2D 2 si la taille des jeux de données augmente. Implémenter un ResNet-18 à 3 dimensions à la place d'un ResNet-50 à 3 dimensions, ou utiliser un super ordinateur voire simplement utiliser un ordinateur qui possède plus de mémoire de calcul pourrait permettre de choisir une taille de *batch* et d'images plus grande ainsi que d'échantillonner plus d'images par seconde par vidéo. Ceci pourrait améliorer la qualité de ces modèles.