Article

Performance comparison of object detectors in detecting wildlife animals in camera-trap images vis-à-vis their performance on MS COCO

Frank G. Kilima[1], **Shubi Kaijage**[1], **Edith Luhanga**[1], **Colin Torney**[2]

[1]School of computational and Communications Sciences and Engineering (CoCSE), Nelson Mandela African Institution of Science and Technology (NM-AIST), Tanzania

[2]School of Mathematics and Statistics, University of Glasgow, UK

E-mail: kilimaf@nm-aist.ac.tz

Received 6 June 2023; *Accepted 10 July 2023*; *Published online 31 August 2023*; *Published 1 March 2024*

Abstract

Manual analysis of large amount of wildlife image data collected through camera-trapping is time consuming, expensive, prone to errors and bias, and laborious. It also constrains geographical area, size and duration of wildlife studies, and delays decision making. Recent advancements in deep learning (DL) have provided promising solutions for automating such analyses, and addressed many such drawbacks. However, the existence of many object detectors with varied performance on different datasets may complicate the choice of appropriate object detector for a particular task. This study deployed experimental approach to achieve two goals; firstly, to compare predictive performance and processing speed of Faster R-CNN and single shot multibox detector (SSD) across four feature extractors namely ResNet50, ResNet101, ResNet152 and Inception ResNet on MS COCO, vis-à-vis their performance on camera-trap image dataset (CTID) with 11,019 images. Secondly, to assess and compare performance of the same object detectors when trained on the same CTID. We found that object detectors demonstrate a smaller range in mean average precision (mAP) of 2.72% in our CTID than in MS COCO dataset (8.4%) and that detectors' performance on MS COCO and CTID does not match. We also found that all detectors attained similar predictive performance (accuracy) for each animal class on CTID, and that they all performed well in detecting hyena, giraffe, warthog, lion, and guineafowl, but poorly on baboon, buffalo and wildebeest. Lastly, detectors performed better on some small-bodied species like guineafowl and hartebeest than on large-bodied species like zebra and elephant, respectively, despite the large-bodied species having more training data than small-bodied ones. Similarly, all object detectors performed poorly on zebra with the largest training data (1351 images) than on hyena (931), giraffe (920), warthog (979), lion (862), guineafowl (931), hartebeest (479) and elephant (744). Similarly, elephant was outperformed on hartebeest by all detectors.

Keywords camera-trap image dataset; convolutional neural network; deep learning; MS COCO; object detection.

Computational Ecology and Software
ISSN 2220-721X
URL: http://www.iaees.org/publications/journals/ces/online-version.asp
RSS: http://www.iaees.org/publications/journals/ces/rss.xml
E-mail: ces@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

Camera-trapping is increasingly being deployed in many research projects studying populations, behaviours, ecological roles of wildlife animals in ecosystems as well as protection of endangered wildlife animals (Bowley et al., 2018; Chen et al., 2015; LeBien et al., 2020; Norouzzadeh et al., 2018, 2021; Schneider et al., 2018; Tabak et al., 2019; Torney et al., 2019; Valletta et al., 2017; Weinstein, 2017; Yousif et al., 2019). This is owing to its ability to generate large amount of image data, easy deployment and use, inexpensiveness, being unobtrusive (non-intrusive) to animals, detect many wildlife species and safer to researchers compared to other ground-based methods (Bowley et al., 2018; Chen et al., 2015; Weinstein, 2017; Yousif et al., 2019). Other advantages include its ability to produce high resolution images, can be deployed in large or hard-to-access areas, and high reliability. However, due to its widespread use, camera-trapping generates large amount of image data which is traditionally analysed and interpreted by manually checking images for various attributes such as presence of animal species, behaviours (sex, age, feeding, resting, standing, moving, interacting etc.) and animal abundance etc. Manual analyses of large amount of camera-trap image data are time consuming, expensive, prone to errors and biases, and constrains wildlife study in terms of geographical area, size, number, and duration. Such analyses do also not extract all information contained in images and cause delays in decision making due to longer time needed for analyses (Maire et al., 2015; Norouzzadeh et al., 2018, 2021; Schneider et al., 2020; Shepley et al., 2021; Tabak et al., 2019; Torney et al., 2019; Valletta et al., 2017; Yousif et al., 2019). These drawbacks have spurred efforts for effective automated methods for analyses and interpretations of collected camera-trap images using machine learning, specifically deep learning methods such as image classifiers and object detectors (Norouzzadeh et al., 2018, 2021; Schneider et al., 2018; Tabak et al., 2019; Valletta et al., 2017). Such automated methods have proven to reduce human labour, time, and cost in organizing and conducting wildlife surveys and subsequent analyses, produce unbiased analyses and interpretations, reduce dependence on domain expertise for interpretation, and eliminate bottleneck in conducting wildlife surveys (Norouzzadeh et al., 2018, 2021; Schneider et al., 2020; Torney et al., 2019; Weinstein, 2017). According to (Valletta et al., 2017), increasingly large, detailed and high-dimensional data such as images and sound recordings demonstrate a nonlinear relationship over multiple variables which does not conform to many assumptions commonly made in classical statistical methods. Such high dimensional data require the use of other advanced analysis methods such as machine learning methods.

However, the existence of many object detectors with different performance attributes makes choice for an object detector not a straight task. Different research works have published performance (predictive accuracy, processing speed, and other performance measures) of different object detectors on various datasets, including publicly available benchmark datasets such as MS COCO, PASCAL VOC, ImageNet datasets, MNIST (Bowley et al., 2018; Pathak et al., 2018) from which choice for object detectors for various object detection tasks can be made. However, (Bowley et al., 2018; Schneider et al., 2018, 2020) state that most of benchmark datasets contain small-sized images (such as 28×28 pixels for MNIST dataset), one object per image, objects that fill large area of image, clearly visible objects, balanced classes and large number of images (thousands or millions). On another hand, camera-trap images are characterized with limited labelled data, imbalanced classes (Schneider et al., 2020), multiple objects per image, high dimensions (such as 2048×1536 pixels), varied properties of objects in relation to occlusion, illumination conditions, viewpoints, sizes, and locations (Aggarwal, 2018; Jiao et al., 2019; Pathak et al., 2018; Schneider et al., 2018). In addition, it is stated by (Schneider et al., 2018) and (Schneider et al., 2020) that camera-trap images often contain cropped out images, animals which are partly obstructed, too close or too far from cameras, exhibiting different poses and affected by seasonal weather. Given these different sets of characteristics between benchmark datasets and camera-trap images, one question can be raised; *Does the performance of object*

detectors between benchmark datasets such as MS COCO, PASCAL VOC or ImageNet and camera-trap image dataset match? That is; will the object detector which performed better on a benchmark dataset (such as MS COCO) perform similarly better on a camera-trap image dataset than another detector which performed poorly on the same benchmark dataset? This question aims at investigating whether detectors' predictive performance (accuracy) on publicly available datasets such as MS COCO matches with their performance on camera-trap image datasets. According to (Schneider et al., 2018), it is important to assess performance of DL models on CTIDs rather than on benchmark or publicly available general datasets because such datasets may not be relevant to object detection task's niche. They also argue that deeper and practical understanding of predictive performance of DL models on image datasets specific to the task helps researchers to assess benefits (such as high processing speed and predictive accuracy) against associated cost for computational power, storage space, and training time of different DL models. Answering this question will equip researchers with knowledge on whether published performance of object detectors on benchmark datasets such as MS COCO matches with detectors' performance on CTIDs.

Studies such as (Chen et al., 2015; Norouzzadeh et al., 2018, 2021; Schneider et al., 2018, 2020; Tabak et al., 2019; Torney et al., 2019) have deployed deep learning methods to automate analyses of camera-trap image datasets with high accuracies (Norouzzadeh et al., 2018; Tabak et al., 2019; Yousif et al., 2019). However, several of them, such as (Norouzzadeh et al., 2018, 2021; Tabak et al., 2019) have used large datasets with hundred thousands or millions of training images which are often difficult and expensive to collect and annotate by many projects especially small ones (Schneider et al., 2020; Shin et al., 2016). Even when such large image dataset is collected and annotated, not all classes will contain even number of images, which often leads to class imbalance, that is one class having more training data than other classes in the dataset. This is one of the common problems affecting performance of many machine learning algorithms which are designed on the assumption of balanced classes in the dataset. This brings another question; *Does any class imbalance in camera-trap image dataset disadvantage in terms of predictive performance the underrepresented classes?* That is, do object detectors always perform better on classes with more training images than on classes with less training images in the dataset? For instance, will the object detectors perform poorly on an animal class with 500 images than on another animal class with more than 500 images in the same dataset?

Evaluation metrics provide a mechanism for evaluating and comparing models' performance on one class and among classes. However, studies such as (Norouzzadeh et al., 2018, 2021; Schneider et al., 2018; Yousif et al., 2019) which included multiple animal species in their datasets, produced a single performance score (accuracy, mAP, F1-score etc.) across all classes for each model without providing class-specific scores to demonstrate how detectors performed on each animal class in the dataset. This raises one more question; *Can different object detectors perform with significant difference in predictive performance (accuracy) on the same animal class (species) in the same camera-trap image dataset*? That is; can one object detector perform significantly better than another detector on the same animal class in the same dataset? Answering this question will equip researchers with knowledge on whether certain object detectors are better than others in detection of certain animal species in camera-trap images. One advantage of camera-trapping is its ability to capture both small-bodied as well as large-bodied animals. However, (Agarwal et al., 2019; Bagla et al., 2022; Bowley et al., 2018; Nguyen et al., 2020) found out that small-sized objects are more challenging to detect than large-sized objects. This raises another question; *Can object detectors perform significantly better on large-bodied animals such as elephant, giraffe or zebra than they do on small-bodied animals such as guineafowl or warthog?*

This study attempted to achieve two objectives. Firstly, we compared performance in terms of predictive

accuracy and processing speed of two object detectors namely Faster R-CNN and SSD fitted across four feature extractors (backbones), namely ResNet50, ResNet101, ResNet152 and Inception ResNet when trained on MS COCO dataset vis-à-vis their performance when trained on a camera-trap dataset of 11,019 images. Secondly, we compared the predictive performance (accuracy), processing speed and storage space requirement of these detectors when trained on the same CTID using transfer learning, TensorFlow 2 and TensorFlow 2 Object Detection API. Transfer learning is an approach in which a DL algorithm is first trained on a larger and general-purpose (usually a publicly available) dataset such as MS COCO, ImageNet, MNIST etc. before being trained on smaller and more specific dataset for the intended classification or object detection task. Transfer learning ensures that knowledge (initialization parameters such as weights) acquired by the model from the larger dataset is repurposed in training it on smaller and specific dataset (Norouzzadeh et al., 2018). It is a widely used approach in training deep learning algorithms because it requires less training data and time than training them from the scratch (LeBien et al., 2020; Maeda-Gutiérrez et al., 2020; Schneider et al., 2018). CTID used in this study contained eleven different animal species, and was extracted from Snapshot Serengeti (SS)(Swanson et al., 2015) dataset and Serengeti Biodiversity dataset (Saltane, 2020) collected from 2010 to 2013 and2015 to 2019 respectively from Serengeti National Park, in Tanzania. All detectors deployed in this study were pretrained on MS COCO 2017 dataset which contains about 2.5 million labeled instances from 328,000 images, 80 different object classes and 91 stuff categories of common objects such as person, car, handbag, umbrella, chair, laptop, bus, train, traffic light, bench etc. It also contains animal classes such as bird, cat, dog, horse, sheep, cow, elephant, zebra, bear and giraffe. The dataset is commonly used to train and evaluate machine learning models for tasks such as object detection, instance segmentation and captioning(Lin et al., 2014). In realizing these objectives, our study attempted to answer the following questions; (1) Does the performance of object detectors between benchmark datasets and CTIDs match? (2) Does any class imbalance in the CTID disadvantage underrepresented classes in terms of predictive performance? (3) Can different object detectors perform with significant difference in predictive performance on the same animal class (species) in the same camera-trap image dataset? (4) Can object detectors perform significantly better on large-bodied animal classes such as giraffe or elephant than they do on small-bodied classes in the dataset such as warthog or guineafowl?

The main contribution of this paper is to provide more knowledge to DL researchers on whether performance of object detectors on benchmark datasets (such as MS COCO dataset), and CTIDs (such as SS) matches, and on detectors' predictive performance on CTIDs with regard to class imbalance and animals' body size. It also provides understanding and knowledge on whether object detectors perform with significant difference in detecting animals of the same class in a given CTID. This knowledge is useful in improving choices of object detectors for object detection tasks on CTIDs. It will also help DL researchers to better understand how class imbalances, animal species, and body size may influence performance of object detectors. Lastly, it also contributes an annotated camera-trap image dataset of 11,019 images which can be used by other researchers for the aim of improving wildlife monitoring and management using CNN based object detection methods (algorithms).

2 Deep Learning

Deep learning represents a class of machine learning (ML) algorithms which learn features/associations/patterns automatically and progressively from raw data through sets of multiple layers of data representations (Chollet, 2018, p. 8). DL is a subcategory of artificial neural networks (ANNs), a branch of ML based computational algorithms which mimic some properties (functioning) of the human brain (Patterson and Gibson, 2017, p. 41). ANNs are made up of thousands of computational units called neurons (or

nodes)(Wang, 2016), which are connected to one another through numerical values called weights. Common DL architectures include recurrent neural networks (RNNs), recursive neural network, deep belief networks (DBNs), generative adversarial networks (GANs), and convolutional neural networks (CNNs), also known as convnets (Carrio et al., 2017; Chollet, 2018, p. 119). Of all DL architectures, CNNs have become increasingly popular in natural language processing (NLP) and computer vision applications. In computer vision, they are widely being used in object detection tasks such as pedestrian detection, traffic sign detection, face recognition, autonomous car driving, security monitoring, transportation surveillance, drone scene analysis, robotic vision, image interpretation, image classification, precision agriculture, law enforcement, asset inspection, wildlife assessment, drug discovery, genomics and gender classification (Aggarwal, 2018; Jiao et al., 2019; Maire et al., 2015; Schneider et al., 2020; Shanmugamani, 2018; Zhao et al., 2019). Object detection is a procedure of determining an instance of the class (object classification) to which an object belongs and estimating its location (object localization) by drawing bounding box around the object in the image (Pathak et al., 2018).

CNNs are composed of sequence of layers of neurons organized into three-dimensional structure, commonly referred to as spatial dimensionality which represent $(\mathrm{height}, \mathrm{width}, \mathrm{depth})$ (Chollet, 2018, p. 123). CNNs are best suited for computer vision and NLP tasks because of their spatial dimensionality shape which matches with that of input image i.e., $(width, height, colour\ channel)$ (Chollet, 2018, p. 123; Khan et al., 2020; Sakib et al., 2018). This shape enables them to process image data efficiently (Chollet, 2018, pp. 123-124; Khan et al., 2020; Patterson and Gibson, 2017, p. 128; Sakib et al., 2018), combine related tasks such as classification, localization and detection,and optimize their performance(Zhao et al., 2019). Other features include ability to learn translation invariant local features (Chollet, 2018, p. 123), locally connected patches of neurons, ability to exploit relationship between space and pixels of images based on assumption that nearby pixels of a given space are more related than distant ones which are randomly related (Khan et al., 2020) and ability to learn features in spatial hierarchies (Chollet, 2018, p. 123). Translation invariant local features allow any learned features (patterns) to be recognized when they appear in other parts of the image. Locally connected patches allow a neuron in the current convolutional layer to connect to a small patch (region) of the preceding layer, a feature which reduces number of connections and parameters to train while maintaining quality feature extraction (Patterson and Gibson, 2017, p. 130) as well as reducing computations and overfitting (Murphy, 2016). Learning features in spatial hierarchy allows CNNs to learn features in hierarchical manner in which the first convolutional layer learns from data small and local (general) patterns such as edges, second convolutional layer learns larger features out of those learned by the first convolutional layer (Chollet, 2018, p. 123) etc.

3 Materials and Methods

3.1 Data acquisition

This study used a learning set of 11,019 camera-trap images of eleven different wildlife animal species (classes) which included 10 mammals and 1 bird species. Of all images in thelearning set, 10,558 images were extracted from Season 1 of the SS dataset downloaded from http://lila.science/datasets/snapshot-serengeti. SS dataset is a collection of about 3.2 million camera-trap images captured from Serengeti National Park, Tanzania from 2010 to 2013 (Schneider et al., 2018, 2020; Tabak et al., 2019) with 48 different wildlife species. However, only about 25% of SS dataset contains images with animals, with the most abundant species being wildebeest, zebra, thomson's gazelle, buffalo, hartebeest, elephant, human, giraffe, impala, guineafowl, grant gazelle, and warthog (Swanson et al., 2015). The remaining 461 images were extracted from Serengeti Biodiversity Program's dataset, also captured from Serengeti National Park from 2015 to 2019. Serengeti Biodiversity

Program dataset contains about 989,321 camera-trap images of 24 different species, with the most abundant species being wildebeest, zebra, topi, impala, thomson's gazelle, cattle and buffalo (Saltane, 2020). Apart from increasing size of dataset for some classes, combining images from two different datasets increases image features (diversity) such as colour, background, animal poses etc. for DL algorithms to learn from which improves transferability of trained models to new locations. Table 1 presents description of study's learning set.

Table 1 Description of the learning set.

#	Species (classes)	Training set	Validation set	Total
1	Buffalo	642	160	802
2	Elephant	744	186	930
3	Giraffe	920	230	1150
4	Guineafowl	931	233	1164
5	Hyena	1081	270	1351
6	Lion	862	215	1077
7	Hartebeest	479	120	599
8	Warthog	979	246	1225
9	Wildebeest	575	144	719
10	Zebra	1351	338	1689
11	Baboon	251	62	313
Total		**8815**	**2204**	**11019**

The learning set was split with stratified random sampling techniques into 8,815 images (80%) and 2,204 images (20%) for model training and validation respectively, using Python script called partition_dataset.py. Unlike studies such as (Norouzzadeh et al., 2018; Tabak et al., 2019; Yousif et al., 2019) which used large learning sets with hundreds of thousands or millions of images which are practically difficult to collect and hand-label (annotate) by many, specifically small studies, this study deployed a fairly small learning set which can easily be collected and hand-labelled. The learning set described in Table 1 manifests class imbalance (a common feature in many CTIDs) as three classes namely baboon, hartebeest and wildebeest contain much less images than other animal classes specifically zebra, hyena, warthog, guineafowl, giraffe and lion.

3.2 Data processing and preprocessing

The learning set was hand-labelled (annotated) for bounding box coordinates into XML files (Pascal VOC format) using labelImg software. The resulting annotations were converted into .csv files by using a Python script called xml_to_csv.py (available on https://github.com/datitran/raccoon_dataset). The .csv files (for training and validation sets) were further converted into respective tfrecord format files using a Python script called generate_tfrecord.py (also available on https://github.com/datitran/raccoon_dataset).

3.3 Model selection

This study assessed and compared performance and behaviours of Faster R-CNN and SSD object detectors implemented on four ResNet feature extractors (backbones) namely ResNet50, ResNet101, ResNet152 and Inception ResNetV2.

3.4 Hardware platform and ML/DL frameworks

3.4.1 Hardware platform

Training an object detection algorithm is a resource intensive, and time-consuming activity. All object detectors were trained on Ubuntu server 18.04.3 LTS installed on MS virtual Azure server and mounted with a 12 GB memory NVIDIA Corporation GK210GL [Tesla K80] GPU card. The MS virtual Azure server was

remotely connected to using HP ProBook laptop installed with CORE i5-6200U CPU @2.30GHz, 64-bit Ubuntu 20.04.2 desktop operating system, 16GB of RAM and 1TB of solid-state drive (SSD).

3.4.2 Machine learning/deep learning frameworks and libraries

This study used several ML/DL frameworks and libraries such as TensorFlow 2.4, TensorFlow 2 Object Detection API, Keras and TensorBoard and several other Python and object detection packages/libraries. TensorFlow 2.4.1 and Keras 2.4.0 were installed on Python virtual environment created on Python 3.5.6 in MS Azure virtual server. TensorBoard 2.4.0 was used to graphically visualize performance metrics (average precision (AP) and mean average precision (mAP)) and training time of trained models on HP ProBook laptop. Scikit-learn library provided necessary machine learning algorithms and functions for performing linear regression in assessing correlation of various variables (size of training data and predictive accuracy).

3.5 Hyperparameter tuning and model training

Object detectors were trained one by one by fine tuning various hyperparameters such as epochs, warmup steps, batch size, image size, weight value, warmup learning rate, learning rate, momentum value, stddev and data augmentation features. Tuning the hyperparameters and rerunning training experiments were repeated several times and the hyperparameter space of an experiment with the highest mAP was recorded. Each training experiment comprised of 15 epochs, with each epoch having 2,204 steps (33,060 steps in total). Owing to memory limitation, a batch size of 4 images per step and image dimension of $640 \text{ x } 640$ pixels were maintained throughout training phase.

3.6 Model evaluation and evaluation metrics

Object detection evaluation metrics such as AP, mAP, average recall (AR), mean average recall (mAR), intersection over union (IoU) and F1-score quantify performance of object detectors in drawing bounding boxes around detected objects in relation to ground-truth bounding boxes. This study used two PASCAL VOC standard-based evaluation metrics called AP, and mAP. In order to infer some behaviours of trained detectors, the study did observe training time (in minutes) and storage space (in gigabyte (GB)) consumed by each detector when trained on our CTID. The evaluation of each object detector was invoked and performed only once on the checkpoint produced at the end of each training epoch, at which performance values for all evaluation metrics under consideration were recorded. In every training experiment, detector's performance measures for all metrics under consideration were recorded from the epoch with the highest mAP. The batch size at evaluation phase was maintained at 1 image per step.

3.6.1 Average precision

Average precision is an accuracy measure that expresses percentage of correct predictions of an object detector in detecting objects of a particular class in the dataset obtained by computing area under curve (AUC) of $\text{precision x recall curve}$ (Padilla et al., 2020). It is obtained by averaging precision across all recall values from 0.0 and 1.0, at one or various intersection over union (IoU) values (Padilla et al., 2020). IoU is the ratio between intersection (overlapping) and union of predicted bounding box (B_p) and ground truth box (B_{gt}). It is mathematically depicted as;

$$IoU = \frac{area\,(B_p \cap B_{gt})}{area\,(B_p \cup B_{gt})}$$

Precision is the fraction (percentage) of all correct (true) positive predictions over all positive predictions made by the model (Padilla et al., 2020).

$$\text{Precision} = \frac{True\,Positives\ (TP)}{True\,Positives\ (TP) + False\,Positives\ (FP)}$$

Recall is the fraction of all correct (true) positive predictions among all positive ground truths (Padilla et al., 2020).

$$\text{Recall} = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Negatives\ (FN)}$$

Unlike MS COCO evaluation standard which computes mAP by averaging APs at ten different intersection over union values ($IOU \in [0.5 : 0.05 : 0.95]$), PASCAL VOC standard computes AP at only one IOU value i.e., $IoU = 0.50$. Generating performance evaluation (AP) for each class allows for comparison of detectors' performance on the same class and different classes and further investigation when needed.

3.6.2 Mean average precision (mAP)

Mean average precision is a commonly used overall score evaluation metric (measure) for object detection models. It is an evaluation measure obtained by averaging all average precisions of all classes in a dataset in order to produce an single numerical performance score across all classes for each detector (Padilla et al., 2020). The averaging of AP is done over one or multiple IoU thresholds such as mAP@0.5for PASCAL VOC evaluation metrics standard ormAP@[0.5:0.05:0.95] for MS COCO evaluation metrics standard. It is mathematically defined as $\text{mAP} = \frac{1}{C}\sum_{i=1}^{C} AP_i$ where C is the number of classes in the dataset and AP_i is the AP of the ith class (Padilla et al., 2021). Since PASCAL VOC evaluation metrics standard was used, the mAP was computed at only $\text{IoU} = 0.5$, i.e., mAP@0.5.

4 Results

4.1 Object detectors' predictive performance on MS COCO and CTID

Table 2 presents processing speed and mAP of object detectors trained on MS COCO dataset and in our study's CTID. The detectors' performance on MS COCO dataset was extracted from https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md.

Although the MS COCO evaluation measured (expressed) processing speed in milliseconds (ms) which is different from our study's measure for processing speed (minutes), it is sufficient enough to produce a mapping on whether a detector is faster or slower than others in each case (MS COCO and CTID). Despite low mAP values, which are ascribed to the large number of object classes/categories (80) in MS COCO dataset, they are good enough to provide performance comparison among detectors. Results from Table 2 show that detectors have demonstrated smaller mAP range of 2.82% (between the highest mAP of 82.86% (Faster R-CNN ResNet101) and lowest mAP of 80.04% (Faster R-CNN Inception ResNet)) in our CTID, compared to mAP range of 8.4% demonstrated in MS COCO (between Faster R-CNN Inception ResNet at 37.7% and Faster R-CNN ResNet50 at 29.3%). Table 2 further shows that Faster R-CNN Inception ResNet (37.7%) and Faster R-CNN ResNet101 (82.86%) have performed the best on MS COCO dataset and on our study's CTID respectively, while Faster R-CNN ResNet50 (29.3%) and Faster R-CNN Inception ResNet (80.04%) have performed the least on MS COCO and on our CTID respectively. Results indicate that none of the top three object detectors on MS COCO evaluation is in the top three detectors in the CTID evaluation, with the best detector on MS COCO (Faster R-CNN Inception ResNet) being the least performing detector on CTID. Results also show that two best performing object detectors in the CTID (Faster R-CNN ResNet101 and Faster R-CNN ResNet152) are among three least performing detectors on MS COCO evaluation. Given the MS COCO evaluation, it is shown that Faster R-CNN (two-stage object detector) is generally less accurate than SSD (one-stage object detector) on all ResNet backbones (i.e., ResNet50, ResNet101 and ResNet152) except Inception ResNet. This means that SSD has generally performed better than Faster R-CNN on MS COCO

dataset. Given our study's CTID, Faster R-CNN has attained higher predictive accuracy than SSD with ResNet101 and ResNet152 backbones, but lower on ResNet50.

Table 2 mAP and processing time of trained object detectors on MS COCO and CTID.

Object detector	MS COCO		Camera-trap dataset		
	Speed (ms)	mAP (%)	mAP (%)	Training Time (Minutes)	Disk space (GB)
Faster R-CNN Inception ResNet	206	37.7	80.04	1330	6.9
SSD ResNet101 (RetinaNet101)	57	35.6	81.76	972	5.9
SSD ResNet152 (RetinaNet152)	80	35.4	81.04	1489	7.2
SSD ResNet50 (RetinaNet50)	46	34.3	82.11	713	3.7
Faster R-CNN ResNet152	64	32.4	82.33	1668	7.3
Faster R-CNN ResNet101	55	31.8	82.86	1048	5.5
Faster R-CNN ResNet50	53	29.3	80.95	604	3.3

Four object detectors namely Faster R-CNN ResNet101, Faster R-CNN ResNet152, SSD ResNet50 (RetinaNet50) and Faster R-CNN ResNet50 (Table 2) moved from lower ranks in MS COCO evaluation to higher ranks in CTID in terms of mAP, and therefore considered to have performed better on CTID than on MS COCO. Three other object detectors namely SSD ResNet101 (RetinaNet101), SSD ResNet152 (RetinaNet152) and Faster R-CNN Inception ResNet moved from higher ranks in MS COCO evaluation to lower ranks on CTID in terms of mAP and therefore considered to have performed poorly on our CTI dataset than on MS COCO. These results indicate that Faster R-CNN has performed generally better than SSD on CTID.

4.2 Detectors' predictive ability on animals of the same class (species)

Table 3 and Fig. 1 present average precision (AP) of each object detector for every animal class in our study's CTID. Results from Table 3 show that the lowest AP of 57.74% was attained by Faster R-CNN Inception ResNet on baboon, while 98.28% is the highest AP attained by Faster R-CNN ResNet152 on hyena. The Table further shows that all object detectors attained similar AP for each animal class on our CTI dataset. For instance, the APs for all object detectors are between 60.68% - 67.34% and 92.93% - 94.95% for buffalo and warthog respectively. Additionally, all object detectors have performed very well (with APs of at least 87.63%) in detecting guineafowl, lion, warthog, giraffe and hyena, but performed poorly (with APs between 57.74% - 67.52%) in detecting baboon, buffalo and wildebeest. Because APs of all object detectors for each animal class in our CTID are similar (i.e., in close range), their mAPs are similarly in close range as shown in Table 2.

4.3 Effects of body size and training data size on models' predictive accuracy

Based on number of images of each animal class in the CTID, Table 2 and Fig. 1 show that an object detector can attain lower predictive accuracy (AP) for a class with larger training data size than another class with less training data in the same dataset. For instance, while zebra has the largest size of training data (1,351 images), it has been outperformed by all object detectors on hyena (1,081 images), giraffe (920 images), warthog (979 images), lion (862 images), guineafowl (931 images), hartebeest (479 images) and elephant with 744 images. Results from Table 2 and Fig. 1 show that all detectors have performed poorly on elephant with 744 images than on hartebeest with 479 images. Fig. 3 shows the relationship between size of training data per animal class and mean of average precision (AP) of each class across all object detectors. We considered mean of average precision because all object detectors have attained similar APs for each animal class in the learning

set. Similarly, results show that zebra which has much larger body size (and the largest training data), has attained smaller APs by all object detectors than guineafowl, warthog and hyena which have comparatively smaller body size (and less training data). Similarly, all seven models have performed poorly on elephant which has much larger body size and more training images (744 images) than hartebeest which has smaller body size and less training images (479 images).

Table 3 Training data size for each class, Average Precision (AP) and mean AP.

Class	Training data	Faster R-CNN ResNet50	Faster R-CNN ResNet101	Faster R-CNN ResNet152	Faster R-CNN Inception ResNet	SSD ResNet50 (RetinaNet 50)	SSD ResNet101 (RetinaNet101)	SSD ResNet152 (RetinaNet152)	Mean AP (%)
Buffalo	642	63.27	67.34	65.35	60.88	66.72	66.53	63.35	64.75
Elephant	744	77.55	80.06	79.20	78.04	80.33	79.33	78.73	79.03
Giraffe	920	93.07	95.48	95.97	94.40	93.11	93.90	93.18	94.16
Guineafowl	931	89.05	89.92	87.63	88.58	89.40	89.80	90.00	89.20
Hyena	1081	98.14	98.14	98.28	98.15	96.49	97.67	96.51	97.63
Lion	862	94.00	93.73	93.85	91.48	92.22	91.72	91.98	92.71
Hartebeest	479	81.40	84.82	83.65	82.39	83.19	81.67	81.51	82.66
Warthog	979	93.59	93.86	94.95	92.93	93.83	93.19	93.22	93.65
Wildebeest	575	65.42	67.52	65.63	62.84	66.40	66.65	66.64	65.87
Zebra	1351	74.47	77.39	76.42	74.31	77.23	76.92	75.40	76.02
Baboon	251	60.50	63.15	64.65	57.74	64.31	61.95	60.95	61.89

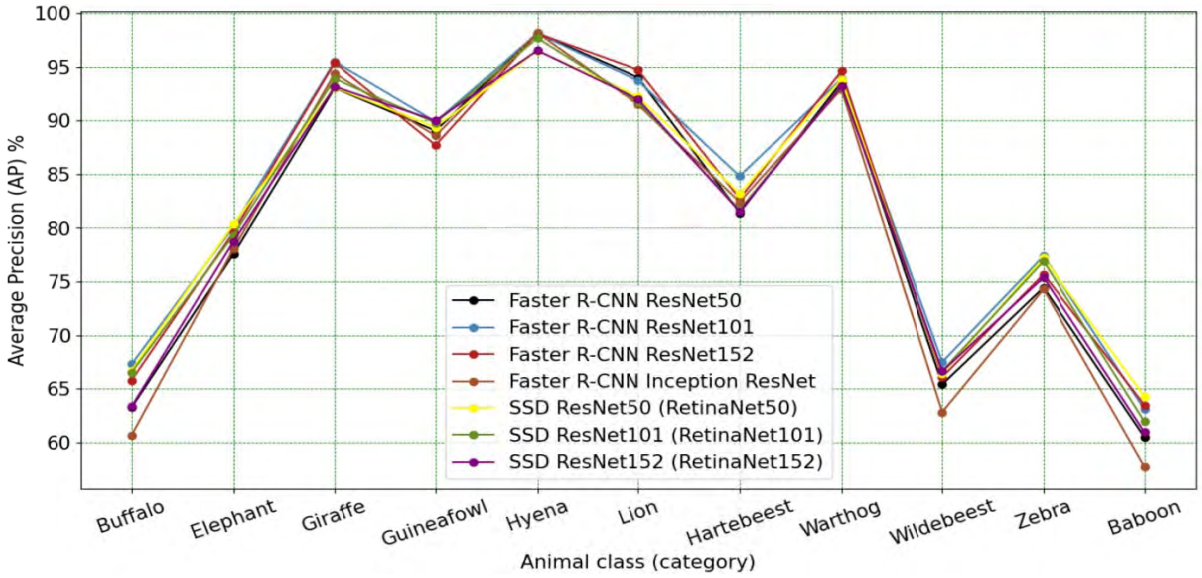

Fig. 1 Average Precision for each object detector.

4.4 Training time and disk space consumed

Training time is an important factor to consider when choosing an object detector for a particular object detection task. Table 2, apart from other information, includes total training time (in minutes) and disk space (in GB) consumed in training each object detector on our CTID. From the Table, it can generally be deduced that given a particular ResNet backbone (such as ResNet101), the more accurate the object detector, the slower it is. Results from Table 4 demonstrate that training time is direct proportional (increases) with the depth (number of layers) of the backbone, that is the more layers the backbone has, the more training time it takes to

train it. For instance, object detectors integrated with ResNet101 generally consume more training time than those integrated with ResNet50. Similarly, detectors integrated with ResNet152 take longer training time than those integrated with ResNet101. Results (Table 4) also show that, generally, given a particular backbone such as ResNet101, SSD is faster than Faster R-CNN. From Table 2, it can be observed that given MS COCO dataset evaluation, Faster R-CNN is faster (has higher processing speed) than SSD in ResNet101 and ResNet152 backbones, but it is slower than SSD in ResNet50 backbone. Given CTID, Faster R-CNN is slower than SSD in ResNet101 and ResNet152 backbones, but faster than SSD in ResNet50.

Table 4 Total training time and disk space consumed by each object detector.

Backbone	Training Time (Minutes)		Disk space (GB)	
	SSD	Faster R-CNN	SSD	Faster R-CNN
ResNet50	713	604	3.7	3.3
ResNet101	972	1048	5.9	5.5
ResNet152	1489	1668	7.2	7.3

Object detectors need adequate storage space for purposes such as storage of network parameters, parameter gradients, network activation, network activation gradients, updater state, training data, temporary arrays, checkpoints, evaluation events, working memory (Patterson and Gibson, 2017, p. 250). Table 4 shows that storage space consumed is directly proportional to the depth of the ResNet backbone and training time, as further demonstrated in Fig. 2. Table 4 shows that detector integrated with a backbone with more layer (such as ResNet101) consumes more training time, and also storage space than when integrated with a less layer backbone (such as ResNet50).

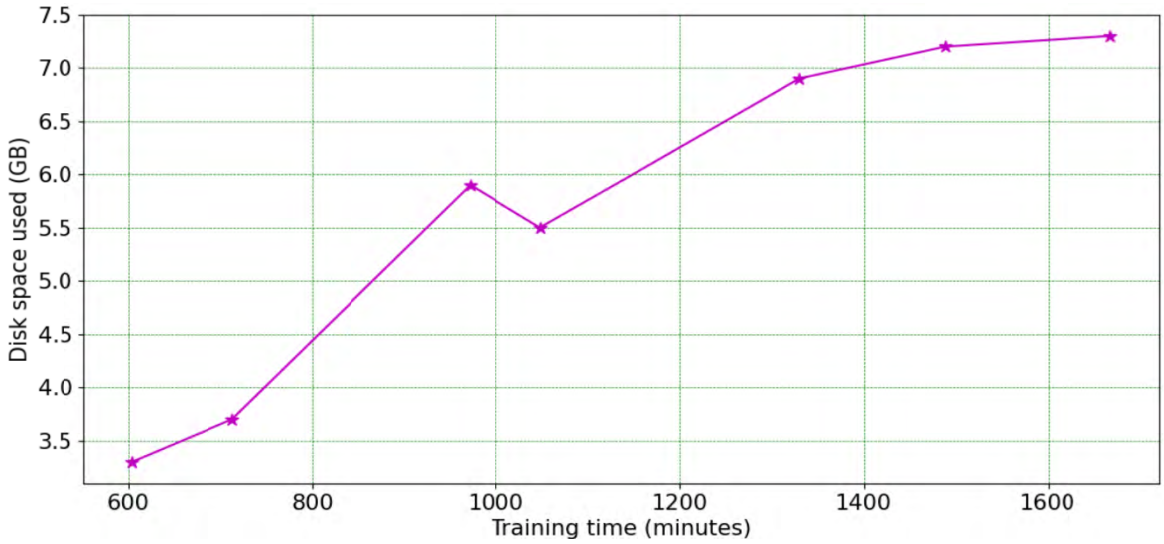

Fig. 2 Relationship between training time and disk space for each object detector.

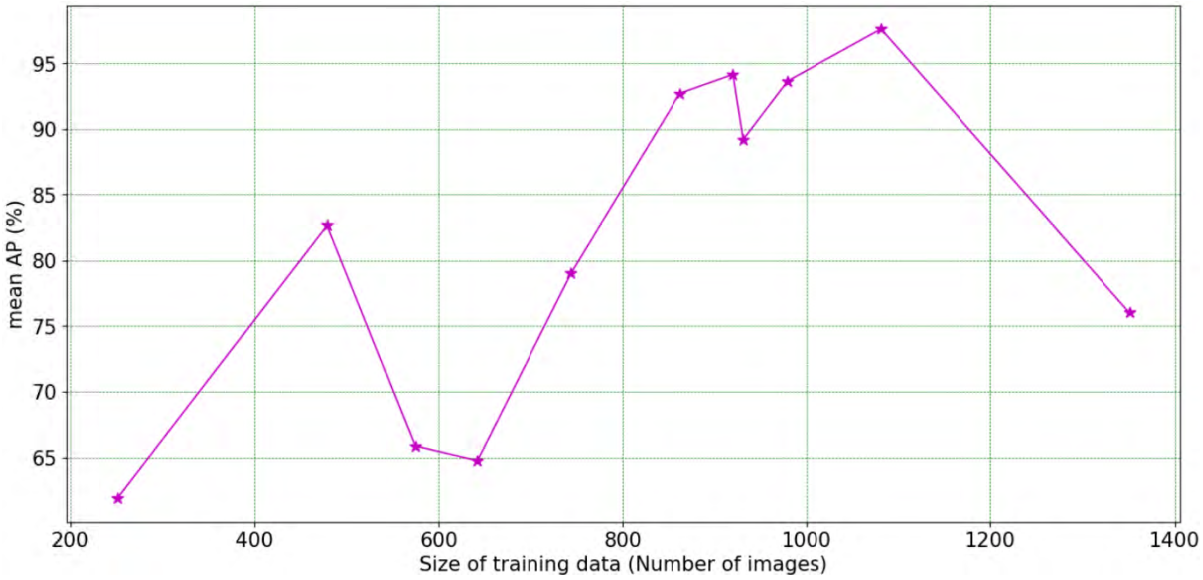

Fig. 3 Relationship between size of training data and mean AP of each class.

5 Discussion

This study attempted to address two issues. First to compare performance in terms of predictive accuracy and processing speed of two object detectors (Faster R-CNN and SSD) when integrated on four different feature extractors (backbones) namely ResNet50, ResNet101, ResNet152 and Inception ResNet, and then trained on MS COCO dataset vis-à-vis their performance on a learning set with 11,019 CTIs of eleven wildlife species. Second, we assessed performance (predictive accuracy, processing speed and storage space) of each of these object detectors and compared with others, when trained on the same CTID. We trained all object detectors on MS Azure server mounted with a 12 GB memory NVIDIA Corporation GK210GL [Tesla K80] GPU card using transfer learning, TensorFlow 2, TensorFlow 2 Object Detection API and Keras.

Our study's results demonstrate that detectors' performance on MS COCO does not match with their performance on CTID in two ways. First, the range in mAP between detectors is smaller in CTID than it is on MS COCO. Second, performance (mAP and processing speed) of individual detectors on MS COCO does not match with their performance on CTID. These variations suggest that object detectors do not behave uniformly (the same) across all types of datasets (e.g., camera-trap image dataset vs non-camera-trap image dataset), and that a detector that performs well on one type of image dataset (such as MS COCO) may not necessarily perform well on another type of dataset (such as camera-trap image dataset), and vice versa. Such variations in detectors' performance across datasets may be caused by factors such as differences in types of objects contained in each dataset, quality of images (size of objects, blurriness, occlusion, light intensity, colour of objects, background, image dimension etc.), training environment, model optimization techniques, class imbalances etc. These variations and their implication in models' performance demonstrate that detectors' performance on benchmark datasets such as MS COCO should not be likened to their performance on a characteristically different dataset such as camera-trap image dataset. Instead, choice of an object detector (on predictive accuracy basis) should rely on detectors' predictive accuracy on characteristically similar datasets. As explained, a detector that attained better predictive accuracy and/or processing speed on benchmark dataset may be outperformed on a different image dataset by another detector with less predictive accuracy and/or processing speed on the same benchmark dataset.

Our results also show that all object detectors attained similar predictive accuracy (AP) for each animal

class in the CTID as they attained comparatively similar average precision on each animal class and performed well on hyena, giraffe, warthog, lion and guineafowl, but poorly on baboon, buffalo and wildebeest. This demonstrates that predictive ability (computational difficult) of all object detectors in detecting animals of a particular animal class in a dataset is comparatively similar, and that no detector performs distinctively better on one animal class than others. This means that given marginal difference in predictive accuracy among object detectors, any detector can be chosen for detection task of any animal class in the CTID, keeping constant other factors such as processing speed, training time etc.

Although (Norouzzadeh et al., 2018, 2021; Schneider et al., 2018; Tabak et al., 2019; Yousif et al., 2019) found that DL models perform better on classes with larger training data than on classes with less training data, our study has shown that some classes with less training data may outperform others with larger training data. Fig. 3 shows that, given detectors' performance on our CTID, there is weak positive correlation between data size (amount of training images) per class and accuracy (mean of APs) with $R = 0.592675$, and that only 35.1% of detectors' accuracy has been contributed (explained) by size of training data. While detectors' poor performance on baboon may be ascribed to class imbalance (having the smallest size of training data (251 images) in the CTID), all detectors have also performed similarly poor on wildebeest and buffalo which have 2.3 and 2.6 times more training data than baboon respectively. Interestingly, all detectors have performed better (than wildebeest and buffalo) on hartebeest which is the second smallest training data size (479 images) after baboon. If class imbalance (small training data size) was the main factor for detectors' poor performance on baboon, then all detectors would have also performed similarly poor on hartebeest than on wildebeest and buffalo. Although small training data size has contributed to some degree to detectors' poor performance, further observation of CTID revealed that a considerable number of images from Serengeti Biodiversity Program dataset contained very small-sized animals (especially for baboon), blurred (fainted) and darkened images. These images were also captured from high angle and long distance (making them difficult to visually see even with human eyes) compared to Snapshot Serengeti images which are largely characterized by large-sized animals, clearly recognizable (visible) and laterally captured and from short distance. Of 461 images from Serengeti Biodiversity Program dataset used in this study, 372 were part of training data, of which 85 were baboon images, 231 wildebeest images, 13 elephant images, 25 hartebeest images and 18 giraffe images. All these images contained about 45 occurrences (not images) of zebra. The presence of such large number of low-quality images in these two animal classes (baboon and wildebeest) must have significantly contributed to detectors' poor performance than their small training data sizes may have contributed. This finding matches with (Schneider et al., 2018) in which Faster R-CNN ResNet101 performed better (93% in accuracy) on a 946-image RCT dataset compared to 76.7% accuracy attained on GSSS dataset with 4096 images. Similarly, (Schneider et al., 2018) cite class imbalance and messier data in GSSS dataset as main reasons for detector's poor performance. This means that in spite of this class imbalance (small training data), detectors may have performed better on baboon and wildebeest if their images were as of good quality as images of other classes which constituted most of SS images. However, all detectors' poor performance on zebra despite it having largely high quality and largest training data still raises question. Our study results further indicate that with transfer learning, wildlife studies may require much less (several hundreds to a few thousands) high quality training image data to attain higher predictive accuracies than many wildlife studies have done before. This will reduce labour, cost, time, computational power and storage space required to collect, annotate large amount of images and eventually train object detectors on such datasets.

Several research such as (Agarwal et al., 2019; Jiao et al., 2019)have shown that object detectors perform poorly on small objects than on medium and large sized objects, owing to the fact that small objects have less information associated with them for detection, can easily be confused with image backgrounds, large image

sizes, and higher precision requirements for localization. However, our research results demonstrate that large body size of wildlife animals (given similar or larger training data) may not automatically lead to better predictive accuracy than animal classes with smaller body sizes. As shown in Table 3 and Fig. 1, guineafowl, warthog, and hyena have outperformed zebra, while hartebeest has outperformed elephant by all detectors despite zebra and elephant having larger body sizes than other classes. This finding demonstrates that with high quality images in which animals are fairly visible, size of animals may not positively influence predictive accuracy of object detectors. It also demonstrates that DL object detectors may effectively be applied to study wildlife animals with less training data such as rare or elusive species and small-bodied animals (like rodents, birds, snakes, small mammals etc.) and produce high predictive accuracy similar to or higher than classes with larger training size and body sizes. However, it also calls for investigation on effects of factors such as skin colours, image background etc., on predictive accuracy of object detectors on animal detection in camera-trap images.

Our study results show that although single performance scores across all classes in the dataset such as mAP, precision, F1-score etc. provide useful information on each model's average performance across the dataset (i.e., all classes), they do not provide information on how each detector performed on individual classes in the dataset. This calls for studies to also consider the use of class-specific performance score such as AP in addition to performance score across all classes. It has been stated by (Schneider et al., 2018) that while object detector's performance (mAP, F1-score, recall, precision etc.) across all species (classes) may be high, it may be significantly low on some classes. For instance, while Faster R-CNN Inception ResNet has attained mAP of 80.04%, the score does not indicate in anyway the detector's poor performance on baboon (57.57%), buffalo (60.68%) and wildebeest (62.84%) nor its excellent performance on warthog (92.93%), giraffe (94.4%) and hyena (98.15%). Generating class-specific performance metrics ensures that predictive accuracy of each detector on each class is known and compared with other classes by the same or different detectors. This will ensure that classes with low APs are investigated to ascertain causes (for such poor performance) such as image background, colour of animals, class imbalance, quality of images (occlusion, varying light intensity, animals' distance from camera, cropped images) etc., and appropriate actions be taken for improvement.

Strong positive correlation between training time and disk space with $R = 0.956634$ means that disk space increases as training time increases, and that about 91.51% of disk space consumed is influenced by training time. Such strong correlation, despite small difference in predictive performance (mAP) between models as shown in Table 2 and Table 4, means that training time and disk space are important factors that may significantly influence the choice of object detector for a particular object detection task. For instance, while Faster R-CNN ResNet101 outperforms SSD ResNet50 (RetinaNet50) by only 0.75% in mAP, it is 1.47 times slower and requires 1.49 times more disk space than SSD ResNet50 (RetinaNet50). Faster R-CNN ResNet152 which outperforms SSD ResNet50 (RetinaNet50) by only 0.22%, is 2.34 times slower and requires 1.97 times more disk space than SSDResNet50 (RetinaNet50). Similarly, Faster R-CNN ResNet101 outperforms Faster R-CNN ResNet50 by only 1.91%, but it is 1.74 times slower and requires 1.67 times more disk space than Faster R-CNN ResNet50. Faster R-CNN ResNet152 which outperforms Faster R-CNN ResNet50 by only 1.38%, is 2.76 times slower and requires 2.2 times more disk space than Faster R-CNN ResNet50. Training time of the object detectors is affected by factors such as size of training dataset, available computational power (CPU, GPU or TPU), hyperparameters settings, image dimension (number of pixels), internal structure (number of layers) of the feature extractor (backbone), structure of detector used etc. For instance, one-stage object detectors (e.g., SSD) are faster than two-stage object detectors (e.g., Faster R-CNN) (Jiao et al., 2019; Schneider et al., 2018; Shanmugamani, 2018). These results evidently mean that for applications in which processing speed is a top priority than predictive accuracy (and given small differences

in their mAP), Faster R-CNN ResNet50 or SSD ResNet50 (RetinaNet50) may be appropriate choices. This is because they attain marginally less predictive performance (mAPs) while incurring less cost (training time and disk space) than Faster R-CNN ResNet101 and Faster R-CNN ResNet152. Such compute (resource) intensive detectors (Faster R-CNN ResNet101 and Faster R-CNN ResNet152) should be deployed for object detection tasks where predictive accuracy is a top priority, and in computing systems with adequate computational power (GPUs and TPUs) and storage space. Our results also show that mAP of object detectors such as Faster R-CNN or SSD integrated on the same ResNet backbone such as ResNet50 increases with training time, that is the higher the mAP, the longer the training time. These results are supported by (Jiao et al., 2019; Schneider et al., 2018) that higher detectors' predictive accuracy comes at greater cost of training time.

6 Conclusion and Future Works

In this study, we first compared the predictive performance in terms of mAP and processing speed of Faster R-CNN ResNet50, SSD ResNet50, Faster R-CNN ResNet101, SSD ResNet101, Faster R-CNN ResNet152, SSD ResNet152 and Faster R-CNN Inception ResNet on MS COCO dataset vis-à-vis their performance on a 11,019 camera-trap image dataset. Secondly, we compared mAP, processing speed and storage space consumption of the same object detectors when trained on a dataset with 11,019 CTIs using transfer learning, TensorFlow 2, Keras and TensorFlow 2 Object Detection API. Our study attempted to find answers on whether performance of object detectors on benchmark datasets matches with their performance on camera-trap image dataset; is positively influenced by large training data size and body size; and whether object detectors can perform differently on the same animal class (species) in the dataset.

First, there has been smaller difference in mAP (2.82%) among detectors in CTID than it is on MS COCO evaluation (8.4%). We also found that mAP and processing speed of individual object detectors on MS COCO do not match on camera-trap image dataset as four detectors which performed better on MS COCO performed poorly on CTID and three detectors which performed poorly on MS COCO performed better on CTID. Due to small difference in their predictive performance (mAP), other factors such as amount of training time, storage space etc. may take into account when choosing suitable detector for a particular object detection task. Second, larger size of training data for a particular animal class may not necessarily result to performance advantage over classes with smaller training data. Our study has demonstrated that zebra which had the largest training data size was outperformed by hyena, giraffe, warthog, lion, guineafowl which have less training data, just as elephant was outperformed by hartebeest despite having more training data. Third, larger body size may not necessarily result to performance advantage over animal classes with smaller body sizes. Our study has demonstrated that zebra which has larger body size (and the largest training data) has been outperformed by all detectors on guineafowl, warthog and hyena which have smaller body size (and less training data), just as elephant which has larger body (and larger training data) has been outperformed by hartebeest with smaller body and less training data. Fourth, all object detectors demonstrate similar predictive ability (computational difficult) for a particular class in a dataset as they attained similar APs for each animal class, performed well on the same set of animal classes (hyena, giraffe, warthog, lion and guineafowl), and performed poorly on the same set of animal classes (baboon, buffalo and wildebeest).

We recommend four research areas for future work. First, further research on causes of object detectors' poor performance on classes with larger training data size such as zebra over classes with less training data like hyena, giraffe, warthog, lion, guineafowl, hartebeest and elephant, as well as buffalo over hartebeest given that the images are of the same quality. Second, to investigate causes of detectors' poor performance on animal classes with larger body size (and larger data sizes) over classes with smaller body size (and smaller training data sizes) given that images are of the same quality. For instance, why zebra has been outperformed by hyena,

warthog and guineafowl just as elephant has been outperformed by hartebeest. The third research area we recommend is to investigate the effects (influence) of untrained image backgrounds (locations) on detectors' predictive performance; that is how images from new locations not seen during training will affect predictive performance of object detectors. Fourth, we recommend further investigation on the threshold of training data for a class below which its predictive performance by a detector degrades. Many studies spend huge amount of time and resources in collecting and annotating large amount of training data, training models over many hours on resource intensive computing systems to attain high predictive accuracy. Establishing threshold for minimum amount of training data in certain computing environment while guaranteeing reasonable predictive accuracy will be of great importance to many studies with small datasets, inadequate computational power, fund and other resources.

In addition to areas for further research, we also emphasize the use of class-specific performance metrics such as AP in addition to overall-class performance metrics such as mAP in order to determine object detectors' performance on each class in the dataset. This will allow researchers to make adjustments (improvements) such as repartitioning or increasing training data, changing hyperparameters etc. in order to attain desired performance for poor performing classes.

Data availability

The dataset used for this study will be available online.

Conflict of interest

Authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgement

We thank Snapshot Serengeti community and Serengeti Biodiversity Program projects for providing access to their camera-trap image datasets which have been an important component of this study. We also extend our gratitude to university of Glasgow for providing access to GPU capable MS Azure server used for training the object detectors.

References

Agarwal S, Ogier J, Terrail D, Jurie F, Agarwal S, Ogier J, Terrail D, Jurie F. 2019. Recent advances in object detection in the age of deep convolutional neural networks. 2019. ffhal-01869779v2f . https://doi.org/HAL Id: hal-01869779

Aggarwal CC. 2018. Neural Networks and Deep Learning. Springer. https://doi.org/10.1007/978-3-319-94463-0

Bagla K, Diwan AD, Agarwal K. 2022. DarthYOLO: Using YOLO for real-time image segmentation. Advances in Transdisciplinary Engineering, 27: 551-558. https://doi.org/10.3233/ATDE220794

Bowley C, Mattingly M, Barnas A, Ellis-felege S, Desell T. 2018. Detecting Wildlife in Unmanned Aerial Systems Imagery using Convolutional Neural Networks Trained with an Automated Feedback Loop. ICCS 2018. https://doi.org/10.1007/978-3-319-93698-7

Carrio A, Sampedro C, Rodriguez-ramos A, Campoy P. 2017. A review of deep learning methods and applications for unmanned aerial vehicles. Journal of Senseors, 2017: 3296874

Chen G, Han TX, He Z, Kays R, Forrester T. 2015. Deep convolutional neural network based species recognition for wild animal monitoring. International Conference on Image Processing (ICIP), November 2016. https://doi.org/10.1109/ICIP.2014.7025172

Chollet F. 2018. Deep learning with Python. Manning Publications Co. https://livebook.manning.com/book/deep-learning-with-python/

Jiao L, Zhang F, Liu F, Yang S, Li L, Feng Z, Qu R. 2019. A survey of deep learning-based object detection. IEEE Access, 7(3): 128837-128868. https://doi.org/10.1109/ACCESS.2019.2939201

Khan A, Sohail A, Zahoora U, Qureshi AS. 2020. A survey of the recent architectures of deep convolutional neural networks. Artificial Intelligence Review, 53(8): 5455-5516. https://doi.org/10.1007/s10462-020-09825-6

LeBien J, Zhong M, Campos-Cerqueira M, Velev JP, Dodhia R, Ferres JL, Aide TM. 2020. A pipeline for identification of bird and frog species in tropical soundscape recordings using a convolutional neural network. Ecological Informatics, 59: 101113. https://doi.org/10.1016/j.ecoinf.2020.101113

Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. 2014. Microsoft COCO: Common objects in context. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8693 LNCS(PART 5): 740-755. https://doi.org/10.1007/978-3-319-10602-1_48

Maeda-Gutiérrez V, Galván-Tejada CE, Zanella-Calzada LA, Celaya-Padilla JM, Galván-Tejada JI, et al. 2020. Comparison of convolutional neural network architectures for classification of tomato plant diseases. Applied Sciences, 10(4). https://doi.org/10.3390/app10041245

Maire F, Alvarez LM, Hodgson A. 2015. Automating marine mammal detection in aerial images captured during wildlife surveys: A deep learning approach. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9457(December): 379-385. https://doi.org/10.1007/978-3-319-26350-2_33

Murphy J. 2016. An Overview of Convolutional Neural Network Architectures for Deep Learning. 1-22

Nguyen ND, Do T, Ngo TD, Le DD. 2020. An evaluation of deep learning methods for small object detection. Journal of Electrical and Computer Engineering, 2020. https://doi.org/10.1155/2020/3189691

Norouzzadeh MS, Morris D, Beery S, Joshi N, Jojic N, 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. https://doi.org/10.1111/2041-210X.13504

Norouzzadeh MS, Nguyen A, Kosmala M, Swanson A, Palmer MS, Packer C, Clune J. 2018. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. Proceedings of the National Academy of Sciences of the United States of America, 115(25): E5716-E5725. https://doi.org/10.1073/pnas.1719367115

Padilla R, Netto SL, Da Silva EAB. 2020. A Survey on Performance Metrics for Object-Detection Algorithms. International Conference on Systems, Signals, and Image Processing, 2020-July(July): 237-242. https://doi.org/10.1109/IWSSIP48289.2020.9145130

Padilla R, Passos WL, Dias TLB, Netto SL, Da Silva EAB. 2021. A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics, 10(3): 1-28. https://doi.org/10.3390/electronics10030279

Pathak AR, Pandey M, Rautaray S. 2018. Application of Deep Learning for object detection. Procedia Computer Science, 132(Iccids): 1706-1717. https://doi.org/10.1016/j.procs.2018.05.144

Patterson J, Gibson A. 2017. Deep Learning A Practitioner's Approach (1st Ed). O'Reilly Media Inc, USA

Sakib S, Ahmed Jawad A, Kabir J, Ahmed H. 2018. An Overview of Convolutional Neural Network: Its Architecture and Applications. ResearchGate, November. https://doi.org/10.20944/preprints201811.0546.v1

Schneider S, Greenberg S, Taylor GW, Kremer SC. 2020. Three critical factors affecting automated image

species recognition performance for camera traps. Ecology and Evolution, 10(7): 3503-3517. https://doi.org/10.1002/ece3.6147

Schneider S, Taylor GW, Kremer S. 2018. Deep learning object detection methods for ecological camera trap data. Proceedings - 2018 15th Conference on Computer and Robot Vision, CRV 2018, 321-328. https://doi.org/10.1109/CRV.2018.00052

Shanmugamani R. 2018. Deep Learning for Computer Vision Expert techniques to train advanced neural networks using TensorFlow and Keras. Packt Publishing Ltd, USA

Shepley A, Falzon G, Meek P, Kwan P. 2021. Automated location invariant animal detection in camera trap images using publicly available data sources. Ecology and Evolution, 11(9): 4494-4506. https://doi.org/10.1002/ece3.7344

Shin HC, Roth HR, Gao M, Lu L, Xu Z, Nogues I, Yao J, Mollura D, Summers RM. 2016. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE Transactions on Medical Imaging, 35(5): 1285-1298

Swanson A, Kosmala M, Lintott C, Simpson R, Smith, A, Packer C. 2015. Snapshot Serengeti , high-frequency annotated camera trap images of 40 mammalian species in an African savanna. Scientific Data, 2: 150026. https://doi.org/10.1038/sdata.2015.26

Tabak MA, Norouzzadeh MS, Wolfson DW, Sweeney SJ, Vercauteren KC, Snow NP, Halseth JM, Di Salvo, PA, et al. 2019. Machine learning to classify animal species in camera trap images: Applications in ecology. Methods in Ecology and Evolution, 10(4): 585-590. https://doi.org/10.1111/2041-210X.13120

Torney CJ, Lloyd-Jones DJ, Chevallier M, Moyer DC, Maliti HT, Mwita M, Kohi EM, Hopcraft GC. 2019. A comparison of deep learning and citizen science techniques for counting wildlife in aerial survey images. Methods in Ecology and Evolution, 10(6): 779-787. https://doi.org/10.1111/2041-210X.13165

Valletta JJ, Torney C, Kings M, Thornton A, Madden J. 2017. Applications of machine learning in animal behaviour studies. Animal Behaviour, 124: 203-220. https://doi.org/10.1016/j.anbehav.2016.12.005

Wang X. 2016. Deep Learning in object recognition, detection, and segmentation. Foundations and Trends in Signal Processing, 8(4). https://doi.org/http://dx.doi.org/10.1561/2000000071

Weinstein BG. 2017. A computer vision for animal ecology. Journal of Animal Ecology, 533-545. https://doi.org/10.1111/1365-2656.12780

Yousif H, Yuan J, Kays R, He Z. 2019. Animal Scanner: Software for classifying humans, animals, and empty frames in camera trap images. Ecology and Evolution, 9(4): 1578-1589. https://doi.org/10.1002/ece3.4747

Zhao ZQ, Zheng P, Xu ST, Wu X. 2019. Object detection with deep learning: A review. IEEE Transactions on Neural Networks and Learning Systems, 30(11): 3212–3232. https://doi.org/10.1109/TNNLS.2018.2876865